Тодор Димов

ИНФОРМАТИКА

за I (прву) годину гимназијског образовања

Одлуком о одобравању и употреби уџбеника за наставни предмет Информатика за I (први) разред средњег гимназијског образовања, бр. 26-420/3 од 07.04.2026. године, коју је донела Национална комисија за уџбенике.

Скопље, 2026.

ИНФОРМАТИКА

I (прву) годину гимназијског образовања

Назив оригинала:
Информатика
за I (прва) година гимназиско образование

Издавач:
ДПТУ „СТУДЕНТСКИ СЕРВИС“ ДОО – Скопље

Аутор:
Тодор Димов

Рецензенти:
проф. др. Газмен Џафери
проф. др. Атанас Христов
Жаклина Милошева

Уредник:
Симона Ристовска

Лектор:
Нена Ристић Костовска
Превод са македонског на српски језик:
Нена Ристић Костовска

Дизајн и техничка припрема:
ДПТУ „СТУДЕНТСКИ СЕРВИС“ ДОО – Скопље

Штампа:
ДПТУ „СТУДЕНТСКИ СЕРВИС“ ДОО
ул. „Перо Наков“ бр. 112 – Скопље

Тираж:
50 примерака

CIP - Каталогизација во публикација
Национална и универзитетска библиотека «Св. Климент Охридски», Скопје

004(075.3)=163.41

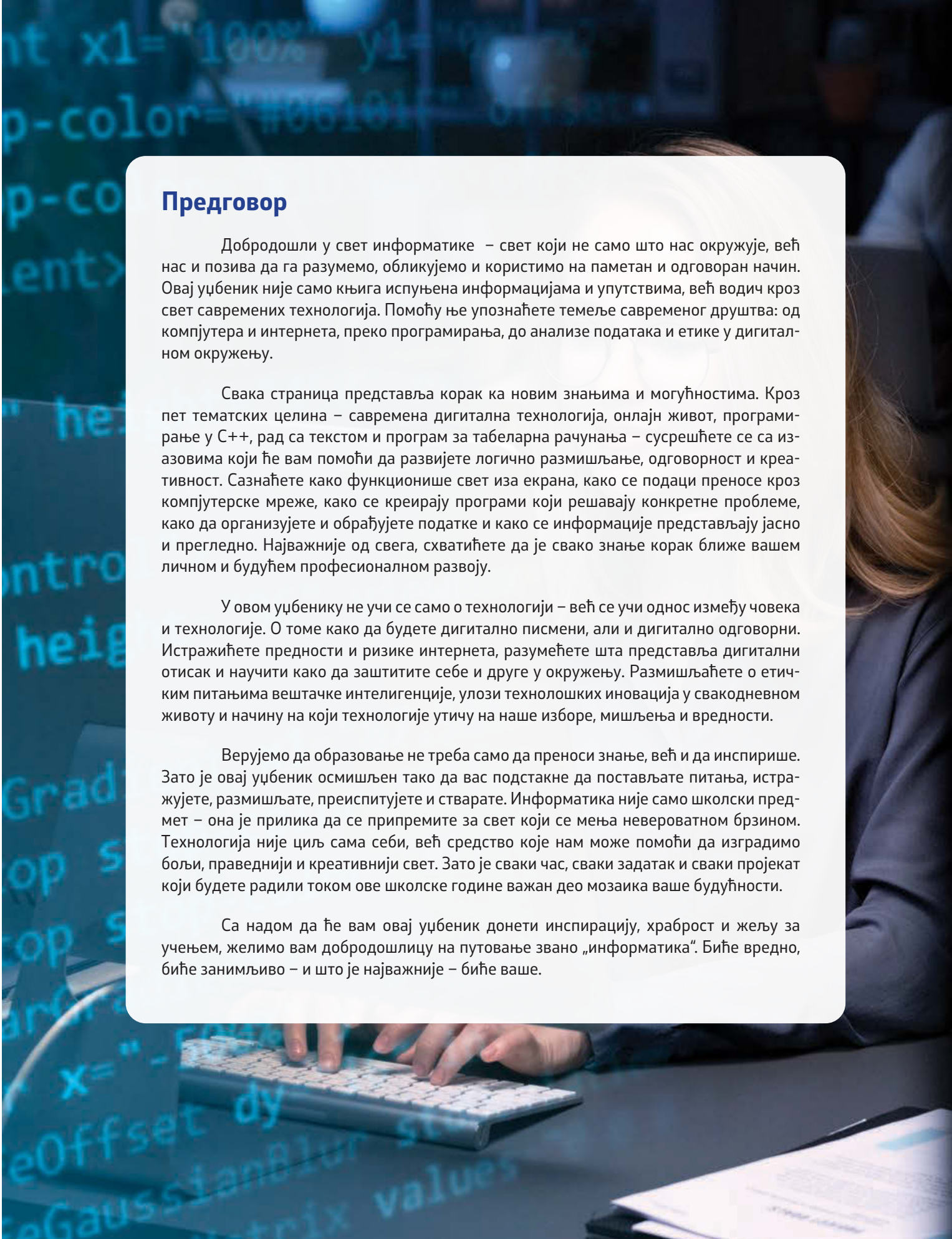
ДИМОВ, Тодор
Информатика за I (прву) годину гимназијског образовања / Тодор Димов ; [превод са македонског на српски језик Нена Ристић Костовска]. - Скопје : Студентски сервис, 2026. - 263 стр. : илустр. ; 26 см

Превод на делото: Информатика за I (прва) година гимназиско образование

ISBN 978-608-4777-54-0

COBISS.MK-ID 69471237

САДРЖАЈ



Предговор

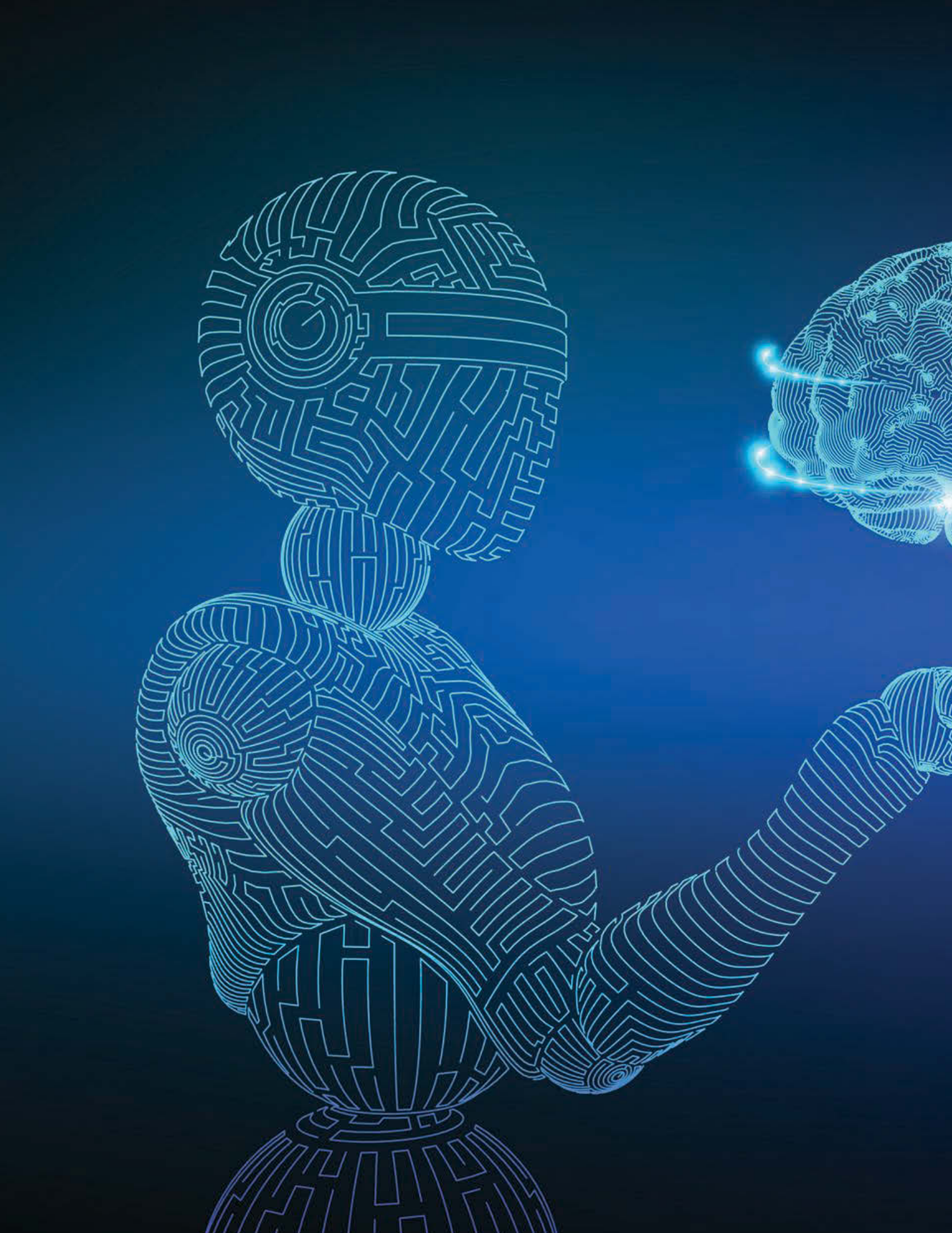
Добродошли у свет информатике – свет који не само што нас окружује, већ нас и позива да га разумемо, обликујемо и користимо на паметан и одговоран начин. Овај уџбеник није само књига испуњена информацијама и упутствима, већ водич кроз свет савремених технологија. Помоћу ње упознаћете темеље савременог друштва: од компјутера и интернета, преко програмирања, до анализе података и етике у дигиталном окружењу.

Свака страница представља корак ка новим знањима и могућностима. Кроз пет тематских целина – савремена дигитална технологија, онлајн живот, програмирање у C++, рад са текстом и програм за табеларна рачунања – сусрешћете се са изазовима који ће вам помоћи да развијете логично размишљање, одговорност и креативност. Сазнаћете како функционише свет иза екрана, како се подаци преносе кроз компјутерске мреже, како се креирају програми који решавају конкретне проблеме, како да организујете и обрађујете податке и како се информације представљају јасно и прегледно. Најважније од свега, схватићете да је свако знање корак ближе вашем личном и будућем професионалном развоју.

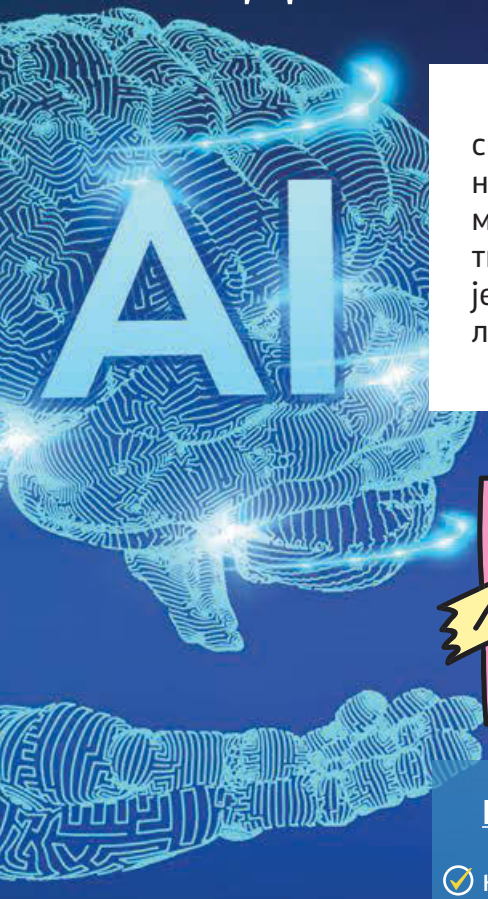
У овом уџбенику не учи се само о технологији – већ се учи однос између човека и технологије. О томе како да будете дигитално писмени, али и дигитално одговорни. Истражићете предности и ризике интернета, разумећете шта представља дигитални отисак и научити како да заштитите себе и друге у окружењу. Размишљаћете о етичким питањима вештачке интелигенције, улози технолошких иновација у свакодневном животу и начину на који технологије утичу на наше изборе, мишљења и вредности.

Верујемо да образовање не треба само да преноси знање, већ и да инспирише. Зато је овај уџбеник осмишљен тако да вас подстакне да постављате питања, истражујете, размишљате, преиспитујете и стварате. Информатика није само школски предмет – она је прилика да се припремите за свет који се мења невероватном брзином. Технологија није циљ сама себи, већ средство које нам може помоћи да изградимо бољи, праведнији и креативнији свет. Зато је сваки час, сваки задатак и сваки пројекат који будете радили током ове школске године важан део мозаика ваше будућности.

Са надом да ће вам овај уџбеник донети инспирацију, храброст и жељу за учењем, желимо вам добродошлицу на путовање звано „информатика“. Биће вредно, биће занимљиво – и што је најважније – биће ваше.



ТЕМА 1: Савремена дигитална технологија



Савремена дигитална технологија обухвата широк спектар алатки и система који користе електронске сигнале и податке за обраду, складирање и пренос информација. Она је основа савремене комуникације, аутоматизације и иновације у различитим аспектима живота, од једноставних алатки као што су компјутери и паметни телефони до напредних дигиталних система.

Нови појмови

- Creative Commons лиценце,

општа вештачка интелигенција,

генеративна вештачка интелигенција.

Већ знаш да...

- набрајаш и описујеш основне делове компјутерских система и наводиш њихове основне функције;
- објашњаваш улогу меморије и процесора у компјутеру;
- набрајаш разне облике меморија;
- објашњаваш својим речима функције хардверских уређаја;
- објашњаваш разлике између разних уређаја, деск-топ компјутера, лаптопа, таблета, смартфона;
- образложиш појмове вирус и антивирусни програм;
- поштујеш основна правила етичког коришћења компјутера.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Истражи свој компјутер или лаптоп или користи слику уколико немаш приступ правом уређају. Затим, креирај табелу у програму за обраду текста као што је приказана наниже и попуни пратећи пример у првом реду!

Део компјутера	Опис (својим речима)	Функција
CPU (процесор)	Извршава све наредбе и „мисли“ шта треба да уради компјутер.	Извршава све инструкције из програма и управља радом других делова.
RAM (меморија)		
Хард диск (HDD/SSD)		
...		

2. Искористи исту табелу, додај још две колоне и измени наслове у колонама као пример наниже да би упоредио разне уређаје! Допуни табелу другим уређајима које знаш!

Уређај	Коришћење (где?)	Преносивост Да/Не	Величина екрана	Батерија (радни сати)
Десктоп				
Лаптоп				
Таблет				
Смартфон				
...				

3. Направи постер у електронском облику на тему – „Како да се заштитимо од вируса“ са следећим елементима:

- Кратко објашњење компјутерског вируса.
- Три савети за заштиту.
- Име једног антивирусног програма.

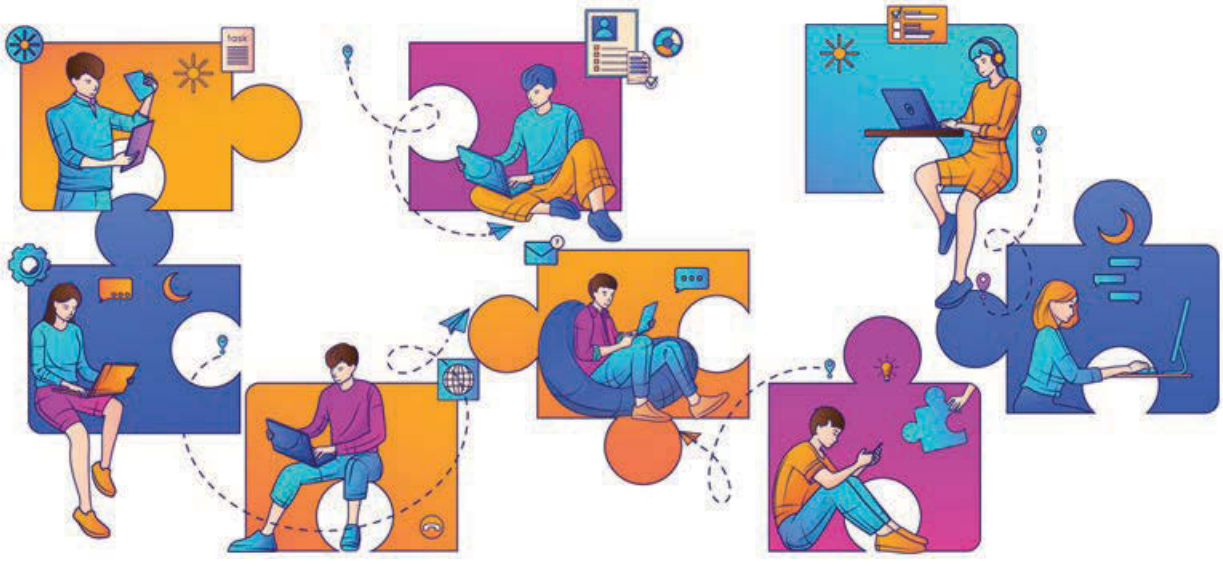
4. Напиши пет правила за етичко коришћење компјутера и интернета! Можеш да их напишеш као „Не треба да...“ или „Увек треба да...“

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- идентификујеш модел компјутерског система у зависности од намене;
- наводиш и описујеш савремене дигиталне технологије;
- објашњаваш значај и улогу дигиталних технологија у савременом друштву;
- користиш одговарајући софтвер за различите намене;
- прилагођаваш ставке оперативног система;
- објашњаваш вештачку интелигенцију преко њених карактеристика.

1.1 Улога дигиталне технологије у свакодневном животу

У савременом друштву дигитална технологија је присутна готово у сваком аспекту живота. Омогућава аутоматизацију, повезивање и обраду података на начине који су до недавно били незамисливи. Код куће, на радном месту, у образовању и здравству, дигитална технологија обезбеђује средства и системе која олакшавају свакодневицу и повећавају квалитет живота. Уз њену помоћ информације се добијају брже, комуникација се одвија у реалном времену, а задаци који су захтевали сати рада сада се завршавају за неколико секунди па чак и милисекунди. Разумевање њене улоге је важан део дигиталне писмености и предуслов за активно учешће у савременом друштву.



Слика 1 *Дигитална технологија у свакодневном животу*

У домаћинствима дигитална технологија се користи преко уређаја као што су паметни телефони, телевизори, гласовни асистенти, сензори за температуру и системи за безбедност. Тако помоћу апликација, корисници могу да контролишу осветљење, грејање или уређајима за забаву у дому. Паметни уређаји не само што повећавају удобност, већ и штеде енергију, време и ресурсе преко аутоматизованих сценарија и управљања на даљину.

У образовању ученици и наставници користе платформе за е-учење, дигиталне табле, симулације и алатке за креирање и прослеђивање образовних садржаја. Дигитална технологија проширује приступ знању, омогућава учење од куће, а мултимедијални садржаји чине наставу динамичнијом и интерактивнијом тако да су ученици активније укључени у процес учења. Преко електронских дневника родитељи могу да прате напредак учења своје деце, а помоћу видеоконференција одржавају се састанци и консултације без физичког присуства.

Слика 2 *Дигитална технологија у образовању*

У здравству медицинске установе користе електронске здравствене картоне пацијената, дигиталне дијагностичке уређаје и телеконсултације. Пацијенти заказују термине онлајн, добијају резултате преко интернета и користе мобилне апликације за праћење здравственог стања. Вештачка интелигенција се примењује у анализи медицинских података, док се роботизовани системи користе у хирургији или нези.

На готово сваком сваком радном месту дигиталне алатке омогућавају комуникацију, тимски рад, организацију задатака и аутоматизацију процеса. Рад на даљину, прослеђивање докумената путем услуга у облаку и видеоконференцијских система свè чешће се користе у институцијама и компанијама. Дигитализација омогућава повећану ефикасност, али захтева и нове вештине од радника, као што су сајбер-безбедност, анализа података и дигитална комуникација.

У транспорту дигитална технологија се користи за навигацију, управљање саобраћајем, праћење возила и аутоматизованих плаћања. Паметне апликације за јавни превоз омогућавају праћење возног реда у реалном времену, док навигациони системи оптимизују маршруту кретања и смањују време путовања. Аутомобили са аутономним управљањем, који користе сензоре, камере и алгоритме за ВИ, већ се тестирају на путевима и представљају будућност мобилности.

У трговини је куповање и плаћање преко интернета у сталном расту. Електронска трговина, мобилно банкарство и дигитални новчаници омогућавају трансакције без готовине, са сваког места и у свако време. Путем анализе података, продавнице добијају увид у навике потрошача и понуду прилагођавају према њиховим интересима. Истовремено, алгоритме за препоруке, који се базирају на вештачкој интелигенцији, утичу на то шта и како потрошач купује, отварајући нова питања етике и приватности.

У јавној администрацији, услуге се чешће одвијају у дигиталној форми. Електронско заказивање, онлајн плаћање рачуна и приступ документима преко интернета омогућавају бржу и ефикаснију комуникацију са институцијама. Дигитална технологија смањује потребу за физичким присуством, скраћује бирократске процедуре и повећава транспарентан приступ информацијама. Ипак, ове промене траже развој дигиталне писмености код грађана и увођење безбедносних мера за заштиту личних података.

Негативни утицаји дигиталне технологије такође су присутни и не смеју бити занемарени. Прекомерна употреба екрана доводи до зависности, поремећаја сна и социјалне изолације. Приватност и безбедност су изложене ризику због злоупотребе личних података и сајбер-напада. Поред тога, постоји опасност од дигиталне неједнакости, при чему они без приступа технологији остају ускраћене за информације, образовање и бројне дигиталне услуге. Из тих разлога, дигитална технологија мора да се користи одговорно и критички.

Дигитална технологија постаје саставни део свакодневног живота, омогућавајући бржу комуникацију, паметније управљање ресурсима, бољи приступ услугама и новим облицима учења и рада. Њена примена доноси бројне користи, али се истовремено јављају и нови изазови повезани са етиком, безбедношћу и здрављем. Разумевање ових аспеката омогућава да се технологија користи свесно, одговорно и ефикасно, у служби човека и друштва.

ПРИМЕР 1

Е-учење у пракси

У школи наставник користи дигиталну платформу за учење преко које ученици добијају материјале, решавају тестове и добијају повратне информације у реалном времену.

Ова платформа омогућава и родитељима да прате напредак свог детета, а наставник добија аутоматске извештаје о резултатима.

ПРИМЕР 2

Паметна кућа

У паметној кући преко мобилне апликације контролишу се осветљење, температура и безбедносни систем. Када нико није код куће, систем аутоматски смањује грејање, активира аларм и шаље обавештења када сензори покрета региструју неуобичајено кретање.

ПРИМЕР 3

Телемедицина

Пацијент са хроничном болешћу користи апликацију која мери крвни притисак и шаље резултат директно матичном лекару. При промени изнад дозвољених граница, систем аутоматски алармира лекара и заказује онлајн консултацију.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни три начина на која се дигитална технологија користи у домаћинству!
2. Наведи примере употребе дигиталне технологије у образовању!
3. Објасни **ка**ко дигитална технологија помаже у здравству!
4. Шта је телеме**ди**цина? Објасни на примеру!
5. Наведи пози**тив**не и негативне ефекте коришћења мобилних телефона!
6. Које опасности н**ас**тају услед прекомерне зависности од дигиталних уређаја?
7. Шта се подра**зу**мева под дигиталном неједнакошћу? Како се она може смањити?
8. Истражи **к**ако се дигитална технологија користи у јавној управи у нашој држави!

ПРАКТИЧАН ЗАДАТАК

Замисли да си ИТ администратор у некој школи. Имаш задатак да изабереш одговарајући компјутерски систем за сваку од следећих ситуација. За сваки случај, објасни зашто си избрао тај систем.

Сценарио 1: Канцеларијски посао у школској администрацији

▸ Треба ти систем за обраду докумената, е-пошта и управљање подацима о ученицима.

Сценарио 2: Видеомонтажа и графички дизајн за школски мултимедијални клуб

▸ Треба ти систем који ће моћи да обрађује мултимедију са високом резолуцијом.

Сценарио 3: Учионица за програмирање и учење основа вештачке интелигенције

▸ Треба ти систем који подржава развој софтвера, рад са великим подацима и моделима за машинско учење.

Сценарио 4: Информацијски киоск за посетиоце школе

▸ Треба ти систем који ради стално, једноставан је за коришћење и нема потребу за тастатуром или маусом.

За сваки сценарио изабери један од следећих типова компјутерских система: Десктоп компјутер, лаптоп, сервер, радна станица, уграђени систем (embedded system), таблет/интерактивни екран. Објасни свој избор за сваки сценарио са две до три реченице.

ЗАКЉУЧАК

Дигитална технологија представља неизбежни део свакодневног живота и значајно обликује начин на који учимо, радимо и комуницирамо. Омогућава бржи приступ информацијама, лакшу комуникацију и ефикасно извршавање разних активности. Ипак, поред бројних предности као што је повећана продуктивност, креативност и олакшано учење, постоје и одређени недостаци. Најчешће су повезани са ризицима о заштити приватности, могуће прекомерно коришћење, изложеност дезинформацијама и зависност од технологије. Зато је важно да развијеш критичко разумевање улоге дигиталне технологије, њених предности и потенцијалних опасности. Таква свест помаже људима да користе технологију на одговоран, безбедан и разборит начин, чиме ће јачати своје дигиталне вештине и компетенције за живот и рад у савременом дигиталном друштву.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

У савремене дигиталне технологије спада и виртуелна реалност (VR, Virtual Reality) и проширена реалност (AR, Augmented Reality) помоћу којих се стичу интерактивна дигитална искуства, али функционишу на различите начине. Циљ им је да потпуно „замене“ реални свет компјутерски генерисаном околином.

Виртуелна реалност (VR) је симулација различитих активности и за њу су потребне:

- **VR наочаре/шлем (headset)** – приказује 3D слику пред очима корисника.
- **Сензори за кретање** – који следе покрете главе и тела.
- **Контролери** – који омогућавају интеракцију са виртуелним светом.
- **Аудио-систем** – који обезбеђује 3D звук за реално искуство.

Функционише на тај начин да када ставиш наочаре, видиш дигитални свет који реагује на твоје покрете. Свако окретање главе или покрет рукама прати се и приказује у реалном времену. Програми користе графичке моторе (као Unity или Unreal Engine) да креирају реалистичне 3D околине.

Проширена реалност (Augmented Reality) надграђује реални свет са дигиталним елементима (слике, текст, 3D објекти). Користи информације у реалном времену као што је текст, слика, звук или други виртуелни елементи интегрисани у реалном свету и по томе се разликује од виртуелне реалности. AR интегрише и додаје вредност интеракцији корисника са реалним светом насупрот симулацијама. Потребни су камера и екран – најчешће на мобилном телефону, таблет или AR наочаре, сензори и GPS – за препознавање околине и позицију и софтвер за препознавање објеката/простора – да би се „залепили“ дигитални елементи преко реалног света. Функционише тако што камера снима реални свет, софтвер анализира простор и додаје дигиталне објекте (на пример, 3D модел новог радног стола у твојој соби). Корисник гледа комбинацију реалне и дигиталне слике на екрану.

Истражи како се дигитална технологија користи у „паметним градовима“. Које технологије се примењују за управљање у саобраћају, јавној безбедности, отпаду и енергији? Проучи како велики светски градови користе вештачку интелигенцију за побољшање урбаног живота. Како може VR и AR да помогну у побољшању квалитета живота у граду? Да ли могу да нађу примену и у руралним срединама?

1.2 Компјутерски систем и његове компоненте

Како се повезује дигитална технологија са компјутерским системом?

Дигитална технологија је зависна од компјутерских система. За функционисање ВИ, дигитално учење, аутоматизацију и анализу података, потребни су компјутерски системи које извршава ВИ технологија. На пример, ВИ у образовању ради преко компјутерских система који користе алгоритме за анализу резултата, препознавање образаца и прилагођавање наставе ученику. Дигитална технологија шири функције компјутерских система. Преко ВИ и машинског учења, компјутерски системи добијају нову функционалност: не само обрадом, већ и самосталним учењем, препознавањем, аутоматским одлучивањем (на пример: прихватањем кандидата у образовној установи). Компјутерски системи су платформа за дигиталну трансформацију. Едукативни системи користе компјутерске системе као што је инфраструктура за дигиталну технологију – за администрирање, праћење напретка, персонализована настава, итд.

Компјутерски систем се састоји од хардвера и софтвера. Хардвер укључује физичке компоненте као што су кућиште, монитор, тастатура, маус, хард диск, слушалице, микрофон, штампач и пројектор. Софтвер је збир програма који контролишу хардвер и омогућавају извршавање задатака. Улазни уређаји (као што је тастатура и маус) омогућавају унос података, док излазни уређаји (као што је монитор и штампач) приказују резултате. Сви извршавају разне функције и функционишу на другачији начин.

У кућишту је смештена централна процесорска јединица CPU (Central Processing Unit) која извршава инструкције и управља свим процесима у систему преко својих компоненти. Ради на тај начин што чита, обрађује и извршава инструкције из програма, прави прорачуне и доноси одлуке. Састављена је од:

- **ALU (Аритметичко-логичка јединица):** која врши математичке и логичке операције.
- **CU (Контролна јединица):** која управља протоком података и инструкција.

Подаци које обрађује централна процесорска јединица, CPU, привремено се складирају у радну меморију, названу **RAM** (Random Access Memory). Она функционише тако што привремено чува податке и програме који су у текућој употреби. Ради на следећи начин: кад отвараш апликацију, она се учитава у RAM меморији за бржи приступ. Када се искључи компјутер све се брише из RAM меморије.

Подаци се трајно чувају у трајној меморији (HDD/SSD). У њој се дугорочно чувају разне врсте података (документи, програми, слике). Постоји разлика између HDD/SSD, HDD (Hard Disk Drive) је механички диск, спорији, али јевтинији, а SSD (Solid State Drive) је бржи, без покретних делова и скупљи.

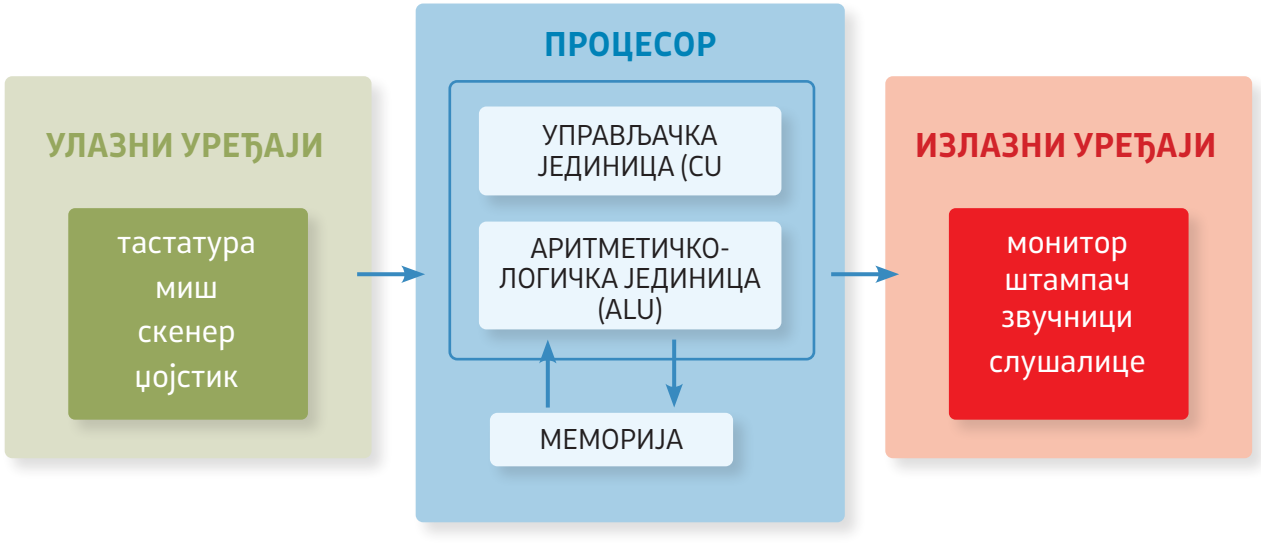
Ово представља основу компјутерске архитектуре која је предложена од Фона Најмана још 1945. године и описује како функционише компјутер

1.3 Фон-Нојманова архитектура и компоненте компјутера

Фон Нојманова архитектура представља основу на којој се заснивају модерни компјутерски системи. Овај концепт је предложен средином 20. века од стране мађарско-америчког математичара Џона фон Нојмана. Према овом моделу, компјутер се дефинише као уређај који извршава инструкције које су унапред записане у својој меморији. Фон Нојманова структура одређује начин на који функционишу основни делови компјутера – процесори, меморије, уређаји за улаз и излаз – и како они међусобно сарађују преко комуникацијских канала. Иако је технологија напредовала, основни принципи дефинисани овим моделом још увек се примењују у савременим компјутерима.

Фон Нојманова архитектура се базира на неколико кључних компонената: централна процесорска јединица (CPU), главна меморија, уређаји за улаз и излаз и комуникацијске стазе такозване магистрале (bus). Процесор контролише рад целог система и састоји се од две главне подјединице – аритметичко-логичка јединица (ALU) и контролна јединица (CU). Аритметичко-логичка јединица врши прорачуне и логичке операције, док контролна јединица интерпретира инструкције и координише све активности у систему.

Главна меморија служи за чување података и инструкција који су тренутно потребни за извршавање. Ова меморија је привремена (волатилна), што значи да се њен садржај губи након искључивања компјутера. У њој се чувају и програми који се се извршавају, као и резултати прорачуна. Меморија се организује у адресе и сваки податак се лоцира уз помоћ уникатне адресе.

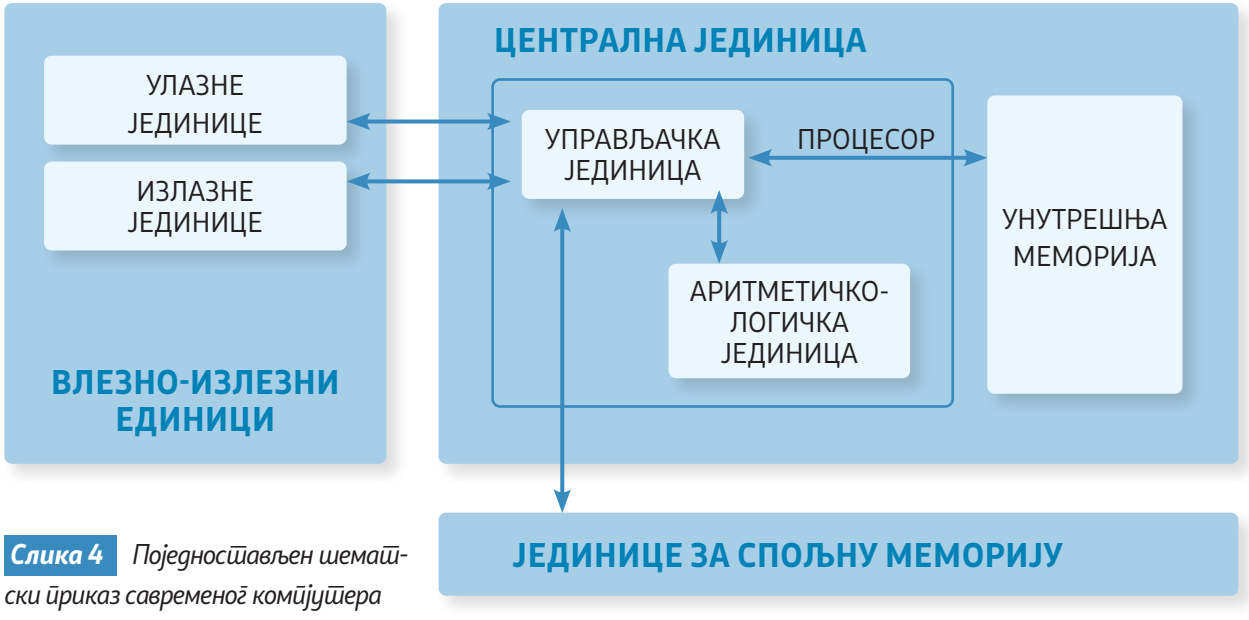


Слика 3 *Поједностављен шематски приказ Фон-Нојмановог модела компјутерског система*

Један од кључних принципа Фон Најманове архитектуре је то што се и подаци и инструкције чувају у истој меморији. Овај приступ омогућава већу флексибилност, јер инструкције могу да се модификују при раду, али истовремено представља и потенцијални безбедносни ризик уколико не постоји јасна разграниченост између кода и података. Управо овај модел је програмирање учинио могућим, јер програми могу бити записани, пренети, чувани и обрађени као било који други податак. Али, инструкције се преузимају и извршавају једна по једна, што ствара уско грло где процесор не може да преузима инструкције и податке истовремено. Ово је познато као уско грло Фон Најмана. За решавање овог недостатка, савремени процесори користе оперативну унутрашњу меморију односно технику за повећање перформанси процесора дељењем извршавања инструкција у секвенцијалним фазама (као преузимање, декодирање, извршавање, приступ меморији, записивање) и обрада разних инструкција истовремено. На тај начин се омогућава да се више инструкција извршава у разним фазама истовремено чиме се значајно повећава укупан проток у компјутеру.

Улазни уређаји омогућавају уношење информација у компјутеру – то су тастатура, маус, микрофон, скенер. Излазни уређаји служе за приказивање или пренос резултата обраде – то су монитор, штампач, звучник. Постоје и комбиновани уређаји који истовремено служе и као улаз и као излаз, на пример екрани на додир и мрежни адаптери.

Магистрале су електронске стазе које повезују све компоненте система. Постоје три основна типа: адресна магистрала (за адресирање меморијских локација), магистрала података (за пренос података) и контролна магистрала (за пренос сигнала контроле). Ове магистрале омогућавају брзу и синхронизовану комуникацију између процесора, меморије и периферних уређаја, чиме се одржава функционалност система.



Слика 4 *Поједностављен шематски приказ савременог компјутера*

Слика 5 *Улазни уређаји*

Централна процесорска јединица (CPU) функционише као код компјутера и њена брзина у великој мери одређује укупни перформанс система. Савремени процесори садрже милијарде транзистора и користе више језгара (cores), што омогућава паралелну обраду информација. Ипак, основни логички принцип редоследног извршавања инструкција као што је дефинисао Фон Нојман, остаје валидан и данас.

Главна меморија се мери у гигабајтима (GB), а за најбржу обраду користи се и кеш-меморија која се налази у самим процесорима. Подаци путују између процесора и меморије огромном брзином, али управо ова брзина представља једно од главних ограничења код модерних компјутера – познато као „Фон Нојманово уско грло“ (Von Neumann bottleneck). Овај термин се користи да се опише ограничена пропусна моћ комуникације између меморије и процесора, што успорава укупну брзину система.

Фон Нојманова архитектура поставља темељ свих модерних дигиталних компјутера. Иако технологија драматично напредује, основни принципи остају непромењени: инструкције и подаци чувају се у заједничкој меморији, процесор их извршава корак по корак, а све компоненте сарађују преко комуникацијских магистрала. Разумевање ове архитектуре је основа за свако даље учење у информатици и омогућава боље разумевање начина на који функционишу компјутери.

ЗАКЉУЧАК

Разумевање начина на који функционише компјутерски систем према архитектури Фона Најмана омогућава ти да схватиш како компјутер прима, обрађује и чува податке. Описивањем улоге процесора, меморије и уређаја за унос и износ, развијаш основну, суштинску дигиталну писменост.

Истовремено, поређење разних компјутерских система и правилно категорисирање улазних и излазних уређаја помаже ти да боље разумеш како сарађују разни делови компјутера. На тај начин се оспособљаваш да изабереш одговарајућу опрему, препознајеш разлике између уређаја и да ефикасније примењујеш технологију у свакодневним задацима. Ова знања не само што ће те припремити за даље учење у области информатике, већ ће те и охрабрити да постанеш свестан и компетентан корисник савремене технологије.

ПРИМЕР 1

Како процесор извршава инструкцију?

Кад се врши операција сабирања два броја, инструкција се чита из меморије, интерпретира се из контролног дела процесора, а затим се извршава у аритметичко-логичкој јединици (ALU), која израчунава резултат.

ПРИМЕР 2

Узајамна сарадња уређаја преко магистрала

Кад корисник притисне дугме на тастатури, сигнал иде преко улазног уређаја, преноси се преко магистрале до процесора, где се обрађује, након чега резултат може да се испрати до излазног уређаја, на пример, да се прикаже на екрану.

ПРИМЕР 3

Зашто постоји кеш-меморија?

Процесор ради брже од меморије и да не би чекао, користи се кеш меморија која најчешће чува коришћене податке. Кеш меморија је бржа, али много мања и значајно побољшава ефикасност извршавања.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Опиши улогу CPU у неком компјутерском систему.

2. Шта је ALU и која је њена функција?

3. Објасни разлику између RAM и кеш меморије.

4. Које су три врсте магистрала и чему служе?

5. Објасни поступно како се извршава нека инструкција према Фон Најмановој архитектури.

6. Истражи шта представља термин ‘Фон Најманово уско грло‘.

7. Зашто је важно да се подаци и инструкције чувају у меморији?

8. Направи поређење између улазног и излазног уређаја са примерима из свакодневице.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи да ли постоје алтернативе Фон Најманове архитектуре и зашто неки истраживачи предлажу нове моделе. Проучи шта представља ‘паралелно про- цесирање‘ и зашто се користе више језгара у модерним процесорима.

Која је улога графичке картице. **GPU – Graphics Processing Unit?** Шта је матична плоча (Motherboard)? Како ови уређаји добијају податке?

1.4 Улазни и излазни уређаји – класификација и поређење

У сваком компјутерском систему улазни и излазни уређаји играју кључну улогу у размени информација између корисника и машине. Преко улазних уређаја, подаци се уносе у систем, док излазни уређаји приказују резултате обраде. Ови уређаји омогућавају интеракцију са компјутером и неопходни су за извршавање свакодневних задатака – од рада са текстом и сликама до комуникације и игара. За правилно разумевање њихове улоге, потребно је да се класификују према функцији, типу података које обрађују и њиховој примени у разним областима.

Улазни уређаји служе за унос података у компјутер. Најчешће коришћени улазни уређаји су тастатура, миш, скенер, микрофон и камера. Тастатура омогућава унос алфанумеричких података, миш омогућава показивање и селектирање, док микрофон уноси аудиосигнал који се даље обрађује. Скенери омогућавају уношење слика или докумената, а дигиталне камере претварају визуелне информације у податке које компјутер може да обради.

Слика 6 *Улазни и излазни уређаји*

Излазни уређаји се користе за приказивање резултата обраде. Међу најпознатијим су монитори, штампачи, звучници и пројектори. Монитори приказују податке визуелно на екрану, штампачи претварају дигиталне информације у физичке копије, а звучници омогућавају аудио излаз. Сваки излазни уређај је дизајниран за специфичну врсту информација и њихов квалитет, брзина и прецизност се прилагођавају према потребама корисника и апликација.

Неки уређаји могу истовремено да функционишу као улазни и као излазни. Такви уређаји се називају улазно-излазни уређаји. На пример, екран на додир омогућава истовремено уношење података преко додира и приказивање информација. Модеми и мрежне картице користе се за размену података преко компјутерских мрежа и исто тако се сматрају I/O уређајима.

Слика 7 *Улазно-излазни уређаји*

Класификација улазних и излазних уређаја може да се изврши према разним критеријумима: према природи података (текст, слика, звук), према начину коришћења (ручни, аутоматски), према брзини уноса/излаза и према тачности. У индустријским и медицинским апликацијама од уређаја се тражи велика прецизност и робусност, док су у домаћим условима најважнији једноставност и удобност при коришћењу.

Савремени улазни уређаји користе напредне технологије за повећану интерактивност и прецизност. На пример, графички таблети користе дигитална пенкала осетљива на притисак за уметнички рад, док биометријски уређаји као скенери за отиске или системи за препознавање лица омогућавају безбедну идентификацију. У виртуелној реалности, улазни уређаји као рукавице и шлемови сакупљају податке о покретима и гестикулацијама, који се затим користе за навигацију и интеракцију у дигиталном свету.

Код излазних уређаја напредак се примећује у повећаној резолуцији, брзини и квалитету излаза. Савремени монитори подржавају високу дефиницију и ултра широки приказ, док 3Д штампачи омогућавају излаз сложених објеката у физичком облику. Ове технологије се користе не само у индустрији, већ и у медицини, архитектури и уметности, где визуелна и физичка репрезентација имају кључну улогу.

У неким системима, посебну категорију представљају сензорни уређаји који региструју физичке појаве (светлост, звук, кретање, температура) и претварају их у дигиталне податке. Ови уређаји се користе у паметним кућама, безбедносним системима и индустријској аутоматизацији. Излазни уређаји у овим системима могу да буду звучни аларми, осветљавање, или извештавање преко мобилне апликације, чиме се добија ефикасно управљање и удаљени мониторинг.

Слика 8 *Сензорни уређаји*

Поред хардверских карактеристика важан је и софтвер који управља овим уређајима – драјвер. Омогућава да оперативни систем препозна функцију уређаја и да комуницира са њим. Без одговарајућег софтвера, хардвер не може да функционише правилно, па зато постоји тзв. слој апстракције између уређаја и програма који их користе.

Улазни и излазни уређаји представљају кључне компоненте за интеракцију са компјутерским системом. Омогућавају подацима да се уносе, обрађују и приказују у разним облицима. Разумевање њихових типова, функционалности и критеријума класификације је основа сваког даљег учења у информатици. Развојем технологије ови уређаји постају све напреднији, омогућавајући нове облике комуникације, аутоматизације и визуелизације у свакодневном и професионалном животу.

ЗАКЉУЧАК

Категоризовање улазних и излазних уређаја помаже да се схвати како се размењују информације између корисника и компјутера.

Улазни уређаји служе за уношење података и команди, док излазни уређаји приказују резултате обраде.

Улазни уређаји (Input Devices)

Уређаји који уносе податке у компјутер:

- **Тастатура** – уношење текста и команди
- **Маус** – усмеравање и избор опција
- **Скенер** – претварање папирних докумената у дигиталне
- **Микрофон** – уношење звука
- **Веб-камера** – уношење видеосигнала
- **Графичка табла (graphics tablet)** – дигитално цртање
- **Тачскрин екран (као улаз)** – уношење додиром
- **Џојстик / gamepad** – управљање играма
- **Баркод читач** – читање кодова
- **Fingerprint скенер** – биометријска аутентикација

Излазни уређаји (Output Devices)

Уређаји који приказују или испоручују податке из компјутера:

- **Монитор** – визуелни приказ слике и текста
- **Штампач** – штампање докумената и фотографија
- **Звучници** – репродукција звука
- **Слушалице** – излаз звука
- **Пројектор** – пројекција слике на зиду/површини
- **Плотер** – штампање техничког цртежа
- **Тачскрин екран (као излаз)** – приказивање слике
- **LED индикатори** – светлосни сигнали

Оваквом представљеношћу се добија јаснија слика о функционисању компјутерских система и помаже свесно да се изабере права опрема за одређени задатак и да постане спретнији, сигурнији и самосталнији корисник дигиталне технологије.

ПРИМЕР 1

Скенирање документа

Када се документ поставља на скенеру, уређај снима текст или слику и прави дигитални облик који се затим приказује на екрану. Овај процес илуструје функционисање улазног уређаја.

ПРИМЕР 2

Штампање извештаја

Након извршене анализе података у табеларном прорачуну, корисник шаље извештај штампачу. Излазни уређај претвара дигитални документ у физичку папирну копију.

ПРИМЕР 3

Коришћење екрана на додир

Корисник на банкомату уноси податке преко екрана на додир, а истовремено добија визуелне и текстуалне повратне информације. Овај уређај је истовремено и улазни и излазни.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Наведи три улазна уређаја који се користе у свакодневном животу и објасни њихову функцију.

2. Који су најчешћи излазни уређаји и за шта се користе?

3. Објасни како функционише неки уређај који је истовремено и улазни и излазни.

4. Направи табелу у којој ћеш упоредити три улазна и три излазна уређаја према: типу података, тачности и брзини.

5. Истражи шта је графичка табла и за које циљеве се користи.

6. Објасни зашто су драјвери важни за правилно функционисање уређаја.

7. Дефиниши биометријски уређај и наведи примере.

8. Напиши како функционишу сензори уређаја у паметно

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се користе улазни и излазни уређаји у специјализованим професијама – на пример, у медицини, архитектури или гејмингу. Проучи који уређаји се користе за рад са виртуелном реалношћу и на који начин симулирају реалну интеракцију са дигиталним простором.

Слика 9 *Улазни и излазни уређаји*

1.5 Савремене дигиталне технологије: додир, холографија, 3Д пројекције

Савремена дигитална технологија се непрестано развија кроз увођење нових начина интеракције, визуелизације и манипулације са информацијама. Међу најупечатљивијим иновацијама су технологије базиране на додиру, холографији и тридимензионалној пројекцији. Ове технологије већ се користе у уређајима за свакодневну употребу, али њихов потенцијал се шири и на области као што је медицина, образовање, архитектура и забавна индустрија. Ови уређаји мењају начин на који људи комуницирају са уређајима и дигиталним садржајем, омогућавајући реалнија, динамичнија и природнија корисничка искуства.

Технологија осетљива на додир заснива се на могућности да корисник комуницира директно са дигиталном површином, најчешће преко екрана, односно физичког контакта. Најчешће се користе капацитивни и резистивни екрани, који на додир реагују прстом или оловком. Данас се додирни екрани интегришу у паметне телефоне, таблете, банкомате, киоске за информације и интерактивне табле у школама. Њихова примена се проширује и у аутомобилској индустрији, здравству и индустријским контролама, где директна и интуитивна интеракција значајно побољшава ефикасност.

Слика 10 *Технологија осетљива на додир*

Додирни екрани могу да препознају више тачака на додир истовремено – тзв. „multi-touch" технологија. Ова технологија омогућава операције као што је зумирање, ротирање и покретање објеката са два или више прстију. Ова функционалност је посебно значајна за корисничке интерфејсе где су визуелне манипулације кључне, као што су апликације за дизајн, игре или навигацију. У образовању ове интерактивне табле на додир повећавају ангажованост ученика и омогућавају визуелни приступ комплексним информацијама.

Холографија представља технологију помоћу које се креира тродимензионална слика која може да се види без потребе за специјалним наочарима. Холограм се формира интерференцијом на светлосне зраке, и садржи информације о дубини, облику и боји објекта из свих углова. Ове слике изгледају као да „левитирају" у ваздуху, и омо-

гућавају јединствено визуелно искуство. Холографске пројекције се користе у маркетингу, уметности, едукацији и презентацији за визуелно преношење информација на импресиван начин.

У медицини се холографија примењује за визуелизацију анатомских структура, омогућавајући боље разумевање и планирање хируршких интервенција. У музејима се холограми користе за реконструкцију артефаката или историјских догађаја, а у музичкој индустрији – за виртуелне концерте покојних уметника. Ова технологија наставља да се развија и обећава нове начине интеракције са дигиталним садржајима у простору.

Слика 11 *Немачки циркус користи холограм за животиње*

3Д пројекција омогућава приказ тродимензионалних објеката на равној површини, са употребом специјалних пројектора и понекад са употребом наочара. Користи се у киноиндустрији, виртуелним турама, архитектонским презентацијама и симулацијама. У образовању 3Д пројекције омогућавају да ученици посматрају структуру атома, анатомија тела или историја објаката са визуелном дубином што не може да се постигне класичним средствима.

Технологија „Heliodisplay“ представља један од најреволуционарних примера пројекције без физичке површине. Слике се пројектују у ваздуху, користећи маглу или пару као медијум. Корисници могу да ‘допру’ као и да манипулишу пројекцијама, што их ово решење ставља у ред сличних холографским екранима из научне фантастике. Ови уређаји се користе у изложбама, презентацијама и луксузним промоцијама, где се тражи висок степен визуелног утицаја.

Савремене дигиталне технологије као што је технологија на додир, холографија и 3Д пројекција нуде нове димензије интеракције и визуелизације. Ове технологије побољшавају корисничко искуство, олакшавају учење и омогућавају импресивну презентацију информација. Разумевање њиховог принципа рада и њихова примена представљају неопходност за младе генерације које ће живети и радити у свету у коме ће такве технологије бити све чешће и напредније.

ЗАКЉУЧАК

Савремене дигиталне технологије се брзо развијају и креирају нове начине комуникације, интеракције и визуелизације. Разликовање ових технологија помаже да се схвати њихова примена и да се прате новитети који се јављају на тржишту што омогућава разликовање разних типова интерфејса, начина уноса и приказивања информација, као и напредних визуелних технологија које све чешће постају део свакодневице.

Технологија на додир омогућава управљање уређајем преко једноставног додира, код старијих модела паметних телефона, банкомата, интерактивних киоска. Технологија вишеструког додира подржава више истовремених додира, што омогућава покрете као што су зумирање, окретање, клизање. Уграђена у савремене паметне телефоне (iPhone, Android), таблете, интерактивне табле. Технологија без додира (Touchless / Gesture-based technology) омогућава контролу уређаја без физичког контакта, користећи гестове, покрете или сензоре. Користи се код Xbox Kinect (контрола покретима), бесконтактних прекидача са сензором, аутоматских врата са сензорима и сличних уређаја.

3D технологија на слици која даје слику са дубином, која изгледа реалистично или „излази“ из екрана. Примењује се код 3Д телевизора и филмова, 3Д наочара (анаглифне, поларизоване, бленда, итд.), 3Д пројектора и 3Д моделирања у игри.

Холографија (Holography) је технологија која даје тродимензионалне слике у простору, видљиве из више углова, без потребе за наочарима.

Примењује се за холограмске концерте (Тупак, Витни Хјустон), код холограмских асистената на аеродромима/музејима, 3Д холограмским рекламама, итд. Хелиодисплеј (Heliodisplay) технологија је уређај који креира слику пројектовану у ваздуху, тако да изгледа као да „лебди“.

Корисник може да се креће кроз мени гестовима, без екрана и без додира.

Користи се за изложбе, интерактивне презентације, маркетинг инсталације са сликама „у ваздуху“ и сл.

Разликовање савремених дигиталних технологија омогућава да се схвати напредак технолошких интеракција – од обичног додира, до интеракције без додира и визуелних 3D и холографских приказа. Тиме се развијају способности да се прате трендови, да се процени применљивост сваке технологије и да се постане припремљен корисник нових дигиталних иновација.

ПРИМЕР 1

Интерактивна учионица

У учионици се користи интерактивна табла са екраном на додир, која омогућава ученицима да решавају задатке директно прстом, повлачењем објеката, као и да пишу дигиталном оловком.

ПРИМЕР 2

Холограм на изложби

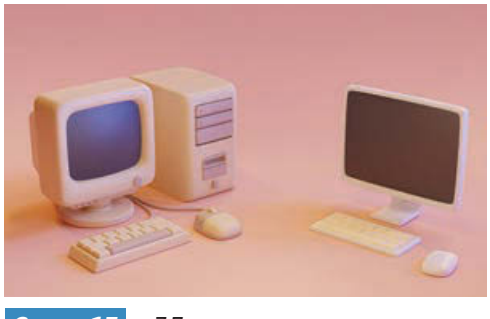
На изложби технологије посетиоци гледају холограм историјског објекта који ротира и приказује се из свих углова. Нема потребе за специјалним наочарима, а ефекат је као да објекат 'лебди' у ваздуху.

ПРИМЕР 3

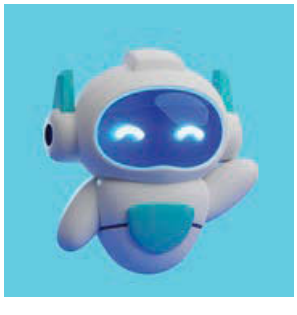
Виртуелна архитектура

Архитекта користи 3Д пројектор да би приказао тродимензионални модел зграде. Клијент може да види како изгледа зграда изнутра и споља, са реалном перспективом.

Слика 12 *Хелиодисплеј*



Слика 13 *3Д слике*



Слика 14 *3Д и холограм*

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни како функционише екран на додир и где се најчешће користи.

2. Шта представља „multi-touch“ и наведи пример где се користи.

3. Дефиниши појам холографија и објасни како се добија холограм.

4. Наведи три области у којима се примењује холографија и објасни зашто.

5. Шта је 3Д пројекција и како се разликује од класичног монитора?

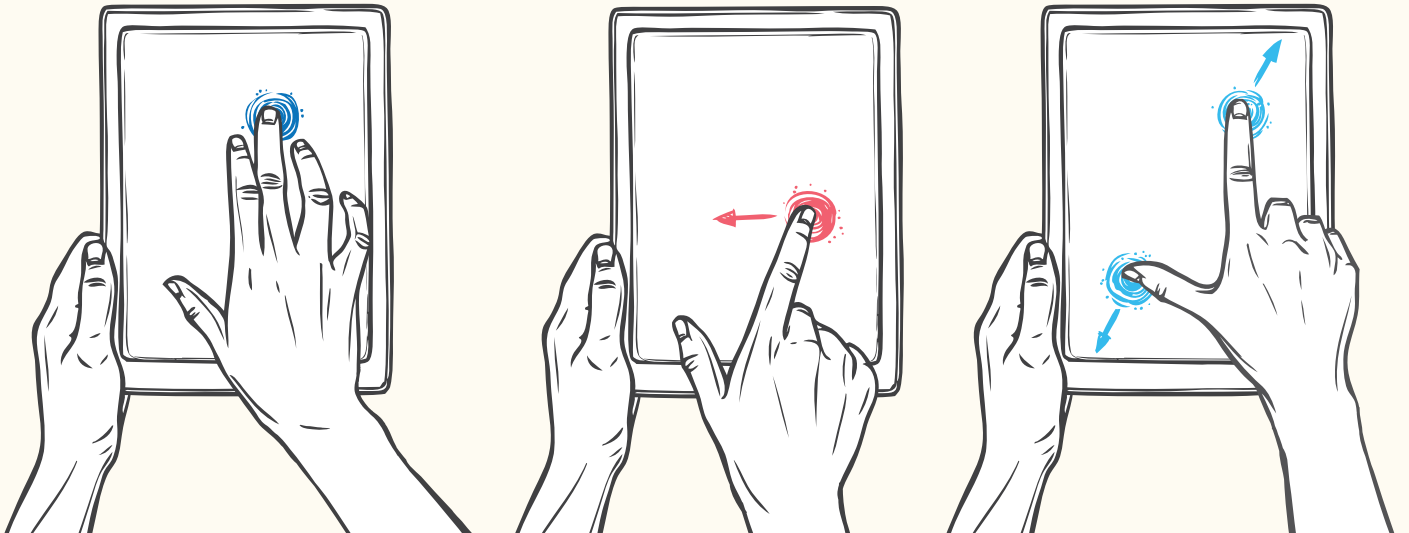
6. Истражи шта је „Heliodisplay“ и како функционише.

7. Објасни како савремене технологије побољшавају учење и презентације.

8. Напиши кратак текст о будућности додирних и холографских технологија.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се користе холограми и 3Д пројекције у медицини и научној визуелизацији. Проучи нове облике интерфејса који не захтевају физички контакт – на пример, управљање гестовима или гласовном контролом.



1.6 Радно окружење оперативног система и апликативног софтвера

Радно окружење неког компјутерског система представља основу за интеграцију и примену дигиталне технологије. Омогућава управљање, анализу, и аутоматизацију процеса преко интеракције са апликацијама базираним на вештачкој интелигенцији, алаткама за продуктивност, и мрежним услугама које повећавају ефикасност, безбедност и флексибилност радног процеса. Интеракције између човека, хардвера, софтвера и дигиталних алатки одвијају се преко интерфејса и његове инфраструктуре, односно радне околине компјутерског система. Радна околина се састоји од:

- оперативног система (Windows, Linux, macOS),
- графичког корисничког интерфејса (GUI),
- алатки и апликација (кориснички програми, базе података, уређивачи, ВИ-платформе),
- мрежне и безбедносне поставке,
- корисничких профила, дозвола и приступа.

Савремени компјутерски системи се базирају на постојању оперативног система који управља радом компјутера и апликативног софтвера, који омогућава извршавање корисничких задатака. Радна околина оперативног система представља интерфејс преко којег корисник комуницира са хардвером и програмима. Разумевање структуре и функције радног окружења, као и улога апликативног софтвера, је кључно за коришћење компјутера на ефективан и безбедан начин. Омогућава једноставну навигацију са оперативним системом, организовање података, стартовање програма и рад са различитим типовима софтвера.

1.6.1 Оперативни систем

Оперативни систем представља основни системски софтвер који координише све ресурсе компјутера – процесор, меморија, уређаји за улаз и излаз и апликације. Најпознатији оперативни системи су:

- Windows  ,
- Linux  ,
- macOS  и
- Android 

При укључивању компјутера оперативни систем се учитава у меморију и припрема радну околину за корисника. Та околина укључује радну површину (desktop), мените, иконе и прозоре за програме.

Најава (login) и одјава (logout) су основне активности при коришћењу компјутера са више корисника. При најављивању сваки корисник добија свој профил са личним поставкама, документима и апликацијама. Радна површина може да садржи иконе за брз приступ апликацијама, датотекама и системским функцијама. Преко траке задатака (taskbar) контролишу се активни прозори, покрећу се програми и прати се статус система. Одређене ставке можеш да прилагодиш сопственим потребама. На пример, језик интерфејса може да се измени на твом матерњем језику, да се дода језичка подршка на тастатури, да се промени скраћеница на тастатури за промену језика, аутоматско закључавање екрана након извесног времена или да компјутер тражи да се унесе лозинка при свакој најави.

Покушај да додаш више корисника на један компјутер са различитим лозинкама. Повежи свој уређај на бежичној мрежи. Које параметре треба да знаш?

Апликативни софтвер омогућава извршавање конкретних задатака као што су писање текста, табеларни прорачуни, графички дизајн и прегледање садржаја (сурфање) на интернету. За разлику од системског софтвера, апликативни софтвер не управља хардвером, већ користи услуге оперативног система за свој рад. Најчешће врсте апликативног софтвера су програми за обраду текста (на пример, Microsoft Word, Writer), програми за табеларне прорачуне (на пример, Excel, Calc), веб-претраживачи (на пример, Chrome, Edge), медијски плејери и образовне апликације.

Датотечни систем је један од најважнијих делова радног окружења. Омогућава чување, организовање и приступ подацима. Датотеке се чувају у директоријумима (фолдерима), који могу бити организовани хијерархијски. Свака датотека има име и екстензију (на пр. .docx, .jpg, .mp3), која означава њен тип. Преко графичког корисничког интерфејса оперативног система корисник може лако да креира, преименује, копира, премешта и брише датотеке и директоријуме.

Различити оперативни системи имају сличне функционалности, али се њихов изглед и начин коришћења могу разликовати. На пример, Windows користи мени Start, иконе и File Explorer, док Linux окружења као што су GNOME и KDE, имају другачије панеле и програме за управљање датотекама. На мобилним уређајима, Android и iOS нуде приступ датотекама преко посебних апликација и услуга у облаку, Захваљујући синхронизацији, датотеке се могу аутоматски усклађивати између више уређаја повезаних са истим корисничким налогом.

Апликације се могу инсталирати из различитих извора – онлајн продавница (као што су Microsoft Store, Google Play), помоћу инсталационих датотека (.exe, .msi) или преко локалне мреже. Важно је да корисник обрати пажњу на извор апликације и услов њеног лиценцирања. Лиценца одређује права коришћења софтвера, односно да ли је апликација бесплатна, софтвер отвореног кода, комерцијални софтвер или пробна верзија која се може користити ограничено време. Creative Commons лиценце се користе за дигиталне садржаје, као што су документи, слике, музика, видео-записи и образовни материјали. Оне омогућавају ауторима да одреде под којим условима други корисници могу да користе, деле или прилагођавају њихова дела.

Оперативни систем такође обухвата алатке за безбедност и одржавање – као што су **антивирусни програми** (антивируси), ажурирање система, контрола приступа корисника приступа и резервне копије. Ове функције су суштинске за заштиту од злонамерног софтвера и чување података у случају квара или губитка. Корисници треба редовно да ажурирају своје уређаје и да користе сложене лозинке, с циљем да се минимизира ризик од неовлашћеног приступа или губитка података. Поред уграђених механизама заштите које обезбеђују оперативни систем, на компјутеру могу да се инсталирају и додатни безбедносни програми.

За већу прегледност, повезаност дигиталних технологија са радним окружењем компјутерског система приказана је у наредној табели.

Дигитална технологија	Веза са радним окружењем
Вештачка интелигенција (AI)	Интеграција у оперативне системе за аутоматизацију (гласовни асистенти, интелигентне препоруке, безбедносно скенирање)
Облачне технологије (cloud)	Радно окружење се шири са локалног компјутера према онлајн радној средини (Google Workspace, Microsoft 365)
Алатке за даљински рад	Радно окружење постаје мобилно: приступ ресурсима са било ког места VPN, Remote Desktop, Teams, Zoom
Сајбер безбедност	Интеграција антивируса, „firewalls“, енкрипција – све то функционише у оквиру радног окружења
Алатке за продуктивност и аутоматизацију	Microsoft Copilot, Notion AI, Grammarly – део софтверског радног окружења које повећава квалитет рада
Интерактивни интерфејси	Дигиталне табле, тач-екрани, гестикулација и VR/AR улази у радно окружење

Табела 1 *Повезаност дигиталне технологије са компјутерским системом*

Дигитална технологија (ВИ, облак, аутоматизација) функционише у оквиру **радног окружења компјутерског система**, омогућавајући бржи, ефикаснији и безбеднији рад.

ЗАКЉУЧАК

Оперативни систем представља основни софтвер који контролише и управља целокупним функционисањем компјутера. Повезује хардверске компоненте са корисником, омогућава реализовање програма, управља датотекама, меморијама, уређајима и безбедносним процесима. Преко ове функционалности оперативни систем креира корисничку околину у којој корисници могу да раде користећи мени, радну површину, поставке, прозоре и друге графичке елементе који олакшавају коришћење компјутера.

Разликовање оперативног система и апликативног софтвера је један од основних дигиталних капацитета. Док оперативни систем представља основу без које компјутер не може да функционише, апликативни софтвер омогућава кориснику обављање конкретних задатака, као што су обрада текста, цртање, комуникација, играње игара, прегледање садржаја на интернету. Према намени, апликације се могу поделити на образовни софтвер, пословне апликације (офис пакете), мултимедијални софтвер, програме за комуникацију, игре, графичке алатке и безбедносни софтвер. Поред коришћења различитих програма, важно је и обезбедити заштиту компјутера и података. Зато корисници треба да познају основне критеријуме за процену квалитета антивирусних и других безбедносних програма, као што су: ниво заштите од вируса и злонамерног софтвера, брзина рада, редовност ажурирања, могућност заштитног зида (firewall), корисничка подршка, утицај на перформансе компјутера и додатне функције, као што су заштита од фишинга (phishing), заштита од рансомвера (ransomware) и слично. Применом ових критеријума повећава се безбедност рада на дигиталним уређајима и смањује ризик од губитка података и безбедносних претњи.

ПРИМЕР 1

Организовање докумената у директоријумима

Корисник креира директоријум на радној површини под именом „Домаћи задаци“. У њему чува све Word-документе које припрема за школу, подељене по предметима. Тиме се добија прегледност и лак приступ потребним датотекама.

ПРИМЕР 2

Стартовање апликација преко траке са задацима

На доњем делу екрана, ученик клика иконицу за веб-прелистач (Chrome) на taskbar. Тиме се апликација одмах отвара, без претраживања кроз мениje.

ПРИМЕР 3

Инсталирање софтвера са лиценцом

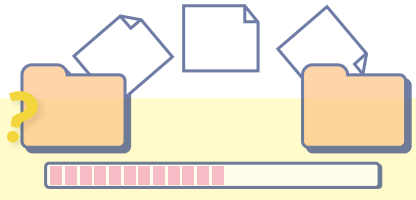
Наставник инсталира образовни програм за учење математике. При инсталацији бира бесплатну верзију са Creative Commons лиценцом, која му омогућава да је користи легално и без ограничења.

ПРИМЕР 4

Директна повезаност са дигиталном технологијом

Наставник ради у компјутерском радном окружењу које обухвата:

- LMS (Learning Management System),
- ВИ-алатку која анализира напредовање ученика,
- врши аутоматску синхронизацију помоћу облака,
- безбедносну заштиту преко енкрипције и двофакторске аутентикације.



ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни улогу оперативног система у компјутеру.

2. Шта се подразумева под појмом „радно окружење“?

3. Које су разлике између оперативног система и апликативног софтвера?

4. Наведи неколико активности које могу да се заврше преко радне површине.

5. Објасни шта је лиценца за софтвер и зашто је важна.

6. Напиши које апликације најчешће користиш и за шта их користиш.

7. Истражи разлику између отвореног и комерцијалног софтвера.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи који оперативни системи се користе у мобилним уређајима, како се разликује њихова радна средина од десктопа на компјутеру и које су њихове предности. Проучи шта су виртуелне машине и како се користе за тестирање и симулацију разних оперативних система!

1.7 Етичка правила за коришћење дигиталне технологије

Разумевање савремених дигиталних технологија — технологија на додир и вишеструког додира, технологије без додира, 3Д приказа, холографије и хелиодисплеја — омогућава корисницима да препознају њихове могућности и ограничења, али и да се припреме за њихово безбедно, креативно и одговорно коришћење. Важан део дигиталне писмености представља и познавање етичких правила коришћења дигиталних технологија. Поштовање приватности и заштита личних података, избегавање увредљиве комуникације и сајбер-насиља, поштовање ауторских права, као и провера тачности информација пре објављивања или дељења, доприносе одговорном коришћењу технологије и смањују могућност злоупотреба, превара и наношења штете. Етичко коришћење дигиталних технологија представља основу за безбедно, одговорно и свесно функционисање у савременом дигиталном друштву. Како се технологија непрестано развија, тако расте и потреба да корисници, посебно млади, науче како правилно да се понашају у дигиталном окружењу, како да заштите своје податке и како да покажу поштовање према другим корисницима.

Поред технолошке писмености, потребно је развијати и **дигиталну етику**, јер савремене технологије, поред бројних могућности, доносе и велику одговорност. Кроз конкретне примере и свакодневне ситуације ученици уче како да безбедно и правилно користе дигиталне технологије и како да постану одговорни дигитални грађани.

Основно правило етичког коришћења технологије јесте **поштовање приватности**. То значи да не треба објављивати туђе фотографије, видео-записе или личне податке без дозволе, чак и када то делује безопасно или је представљено као шала. Личне информације увек треба чувати. Лозинке треба да буду сложене, не смеју се делити са другим особама и не треба их уносити на сумњивим веб-страницама или у непоузданим апликацијама.

Пример: Учени не снимају мобилним (паметним) телефоном током наставе без дозволе наставника, нити снимају друге особе ван школе без њихове сагласности.

Следећи важан сегмент етичког коришћења технологије јесте **заштита личних података**. Посебну пажњу треба посветити биометријским подацима, као што су отисци прстију, препознавање лица или шаренице ока, јер они представљају осетљиве личне податке и морају бити адекватно заштићени.

Пример: Биометријски подаци не деле се са непоузданим апликацијама и сервисима, нити се користе на уређајима и терминалима којима се не верује.

Поштовање ауторских права такође је важан део дигиталне етике. У дигиталном свету слике, музика, видеа и текстови су лако доступни, али то не значи да се могу слободно користити. Увек треба навести извор, поштовати услове лиценце и користити материјале чија је употреба дозвољена, посебно у образовне сврхе. На тај начин развијају се култура поштовања туђег рада и академски интегритет.

Пример: При изради презентације која садржи 3Д моделе потребно је навести извор са којег је модел преузет и поштовати услове његове лиценце.

Дгитално окружење не сме да буде место омаловажавања, исмевања, вређања или ширења негативних садржаја. ости. Свака реч носи последице, чак и када је написана иза екрана. Треба да знамо да пишемо љубазно, да не користимо увредљиве коментаре и да се уздржавамо од учешћа у групним нападима или шалама које могу да повреде некога. Појединачни или групни напади подлежу законској регулативи и кажњиви су. У оваквим случајевима ћутање не помаже, пријављивање је храбар и правилан избор. Исто тако, и сајбер-насиље је једна од најозбиљнијих претњи у дигиталном свету. Етичко коришћење технологије значи активно избегавање оваквих облика понашања, али и препознавање и пријављивање кад се примете.

Данас када се информације шире огромном брзином, важан део дигиталне писмености је и **провера тачности података/информација.** То значи да пре него што објавимо неку вест, видео или фотографију, треба да проверимо да ли је извор веродостојан. На овај начин се спречава ширење лажних вести, манипулације и паника.

Етичко коришћење технологије значи и **одговорно коришћење уређаја.**

Технологија никада не треба да се користи за превару, преписивање, хакирање, лажно представљање или наношење штете. Овај принцип важи и у школској средини — уређаји не смеју да се злоупотребљавају за фотографисање тестова, преписивање или снимање ученика и наставника без дозволе.

Ништа мање важна је и **самоконтрола и балансирано коришћење технологије.** Прекомерно коришћење мобилних телефона, социјалних мрежа или игри може да утиче негативно на здравље, учење и међуљудске односе. Део етичких навика је и препознавање сопствених граница, постављање умерености и прављење свесних избора.

ПРИМЕР 1

Провера тачности пре прослеђивања садржаја

Пре него што проследи видео или вест на Instagram-у, TikTok-у или Facebook-у, ученик проверава да ли је извор веродостојан и наводи тачан извор. На тај начин спречава се ширење лажних вести и манипулација.

ПРИМЕР 2

Етичко коришћење школских уређаја и интернета

При раду у компјутерској лабораторији, ученик примећује да је претходни ученик остао најављен на Google Classroom-у. Он не прегледава профил, већ се одјављује и логира се свог налога. Ученик који користи школски компјутер или таблет не логира се на туђим профилима, не мења поставке без дозволе и не преузима нелегалне садржаје. Уместо тога, користи уређаје само за школске активности и поштује правила школе.

ПРИМЕР 3

Етичко снимање и коришћење дигиталних медијума

Ученик који снима пројекат за школу (на пример: видео-презентацију или интервју) претходно тражи дозволу од свих који се појављују на снимку и јасно објашњава за шта ће се видео користити. За пројекат из историје, група ученика интервјуише наставника. Пре снимања они траже дозволу, информишу га да ће се запис приказати само у разреду и шаљу му финалну верзију за одобравање пре објављивања.

Слика 15 *Паметно коришћење телефона*

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Шта представља етичко коришћење дигиталне технологије? Наведи два правила.

2. Објасни зашто је важно да се не објављује туђа фотографија без дозволе, чак и ако ти изгледа безопасно.

3. Замисли да добијеш поруку у којој неко вређа твог саученика. Шта би урадио поштујући правила етичког дигиталног понашања?

4. Прочитај следећи пример и кажи шта је етички правилно, а шта је неправилно:

5. „Ученик снима део часа својим мобилним телефоном и прослеђује видео у приватној групи без дозволе наставника.“ Образложи разлоге.

6. Процени следећу ситуацију: да ли је етички да проследи вест која звучи шокантно, а да се не провери извор? Објасни твоју одлуку.

7. Осмисли кратко правило које би додао у „Школском дигиталном кодексу“ да би побољшао дигиталну културу међу ученицима.

8. Опиши како би поступио уколико неко у групном чату проследи лажну информацију која може да изазове страх или панику. Које кораке би ти предузео како би етички решио ситуацију?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи колико је деце у свету било изложено претњама на интернету и његовом неетичком коришћењу. Прочитај најновију верзију Акта о дигиталним услугама Европске комисије на следећој адреси:

https://digital-strategy.ec.europa.eu/en/policies/digital-services-act-package.

1.8 Коришћење директоријума и датотека у хијерархијској структури

У сваком компјутерском систему информације се чувају у облику датотека,док се њихова организација врши помоћу директоријума (фолдера). Овај начин уређивања, познат као хијерархијска структура, омогућава логично груписање података, лакше проналажење датотека и ефикасно управљање великим количинама података. Без правилне организације појавиле би се потешкоће у проналажењу, отварању и чувању датотека, посебно када је количина информација велика. Разумевање хијерархије директоријума (фолдера) и именовање датотека је основа за дигиталну писменост и безбедно коришћење компјутера.

Директоријуми представљају просторе у којима се чувају датотеке или други директоријуми. Хијерархијски систем значи да директоријуми могу да садрже поддиректоријуме, чиме се креира структура слична стаблу дрвета. На врху ове структуре налази се коренски директоријум (root), из кога се гранају сви други поддиректоријуми и датотеке.

Свака датотека има своје име и екстензију која указује на врсту података. На пример, .txt означава текстуалну датотеку, .jpg слика, .mp3 аудио запис. Препоручено именовање обухвата описна имена (на пр. „Домаћа математика_мај.docx“) уз избегавање описних знакова. Таква имена олакшавају идентификацију и препознавање датотека.

Датотекама и директоријумима управља се помоћу програма за преглед датотека, као што су File Explorer у оперативном систему Windows или Finder у macOS-у. Корисник може да креира нове директоријуме, копира или премешта датотеку, као и да их сортира према називу, типу, величини или датуму последње измене. У појединим системима могуће је користити и ознаке (тагове) или категорије, ради лакшег организовања и претраживања.

Хијерархијска структура омогућава организовање датотека према различитим критеријумима, као што су предмет, пројекат, намена или временски период. На пример, у главном директоријуму „Школа“, могу се креирати поддиректоријуми „Математика“, „Хемија“, а у сваком од њих чувати домаћи задаци, презентације, фотографије и остали материјали. Оваква организација омогућава брзи присутп подацима и лакше сналажење.

Датотеке се могу креирати и према времену настанка. На пример: „2025 → Март * Екскурзија“. Овај приступ је веома погодан за архивирање фотографија, докумената и других материјала, као и за праћење активности током времена.

Исти принцип организације користе и услуге за складиштење података у облаку, као што су Google Drive, Dropbox. У њима корисници могу да креирају директоријуме, премештају датотеке методом „превуци и пусти“, деле документе са другим корисницима и постављају дозволе за приступ (само за читање или за уређивање). Поред тога, могу да прате измене које су направљене на документима, што омогућава ефикасну сарадњу у реалном времену. Овакве могућности су посебно корисне за тимски рад, израду школских пројеката и вођење дигиталног портфолија.

Добра пракса у организовању датотека подразумева редовно уклањање непотребних датотека, прављење резервних копија и планирање структуре директоријума већ на почетку рада на компјутеру. Неуређена организација може довести до губитка времена, дуплирања података и губитка важних информација.

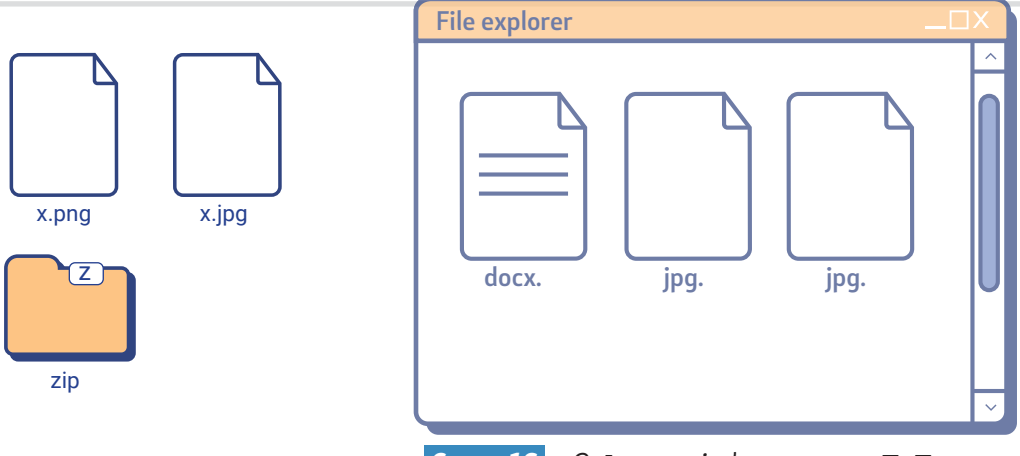
Организовање датотека и директоријума у хијерархијској структури представља основну вештину за сваког корисника компјутера. Добро осмишљена организација омогућава брже проналажење података, повећава ефикасност рада, смањује ризик од губитка информација и олакшава њихово дељење и коришћење. Дигитална писменост започиње управо усвајањем ових принципа, који остају подједнако важни без обзира на уређај, оперативни систем или програме који се користе.

ЗАКЉУЧАК

Постоје различити формати датотека, који одређују врсту садржаја и начин на који се датотека отвара. На пример, .docx и .pdf су формати за текстуалне документе, .jpg и .png за слике, .mp4 за видео-записе, .mp3 за аудио-записе, док су .zip и .rar формати за компресоване архиве. Препознавањем ових формата корисници лакше организују своје податке и повезују датотеке са одговарајућим програмима који их отварају, као што су Word за .docx, PowerPoint за .pptx, Adobe Reader за .pdf, прегледачи слика за .jpg/.png као и медија-плеери за аудио и видео формате.

Поред препознавања формата и правилне организације датотека, важно је разумети разлику између типа датотеке (формата) и програма којим се она отвара. Тип датотеке одређује начин на који су подаци записани, док програм омогућава њихово отварање, прегледање и уређивање. Организација датотека је важна за сваког корисника. Подразумева креирање директоријума и поддиректоријума по логичном редоследу. Оваква структура олакшава проналажење, уређивање и чување података и спречава стварање нереда и смањује могућност губљења важних датотека. Поред препознавања формата датотека и њихове правилне организације, важно је разумети разлику између типа датотеке (format/file type) и програма који је отвара. Тип датотеке одређује начин на који су подаци записани, док програм омогућава њихово отварање, прегледање и уређивање.

Један од најважнијих делова је анализа компресије и њен утицај на величину и квалитет датотека. Код аудио или видео датотека, компресија може да утиче на квалитет звука и слике, у зависности од начина компресовања. Код архивских формата као што су .zip, .rar датотеке се компресују и пакују у једну архиву, чиме се најчешће смањује њихова укупна величина без губитка оригиналног квалитета. Да би се датотеке из архиве користиле, потребно је да се најпре распакују.



Слика 16 *Организација фасцикли и датоте-*

ПРИМЕР 1

Организовање датотека за предмете

Ученици креирају главни директоријум под именом „Школа“, у којој организују три поддиректоријума: „Математика“, „Информатика“ и „Историја“. У сваком од ова три директоријума сачувани су документи у Word-формату са домаћим задацима, PDF-презентације и слике са табле.

ПРИМЕР 2

Структура према датумима

У директоријуму „Фотографије“, креирају се поддиректоријуми по месецима: „2025_Јануар“, „2025_Фебруар“ и „2025_Март“, где се чувају слике са сваког догађаја поређане по по хронологији.

ПРИМЕР 3

Облак-организација

Један ученик користи Google Drive да би чувао све своје пројекте. Он има директоријум „Пројекти“, са поддиректоријумима по предмету. У сваком директоријуму, документи су именовани према садржају и датуму (на пр. „Истраживање_биологија_2025_03.docx“).

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни шта представља хијерархијску структуру датотека.

2. Које су повољности од правилног именовања датотекама?

3. Направи шему како би организовао/ла документе за све предмете у једном полугођу.

4. Шта је екстензија датотеке? Наведи 5 различитих примера и објасни их.

5. Објасни како се креира нов директоријум и како се премешта датотека у њој.

6. Које функције су доступне у File Explorer-у или Finder-у?

7. У чему је разлика између локалног и облачног чувања датотека?

8. Истражи шта представљају резервне копије и зашто су важне.

ПРАКТИЧАН ЗАДАТАК

Оцени програме за заштиту компјутера сагласно критеријумима! Напиши и име програма!

Критеријум	Опис	Име програма	Оцена (1-5)
Ефикасност у откривању претњи	Способност програма да открије вирусе, малвер, шпијунски софтвер и друге претње.		
Реално-временска заштита	Да ли обезбеђује заштиту у реалном времену при сурфању, преузимању и отварању датотека.		
Брзина и перформанси	Колико програм утиче на брзину система. Да ли ради тихо у позадини.		
Употребљивост и интерфејс	Да ли је лака за коришћење, са интуитивним интерфејсом и јасним опцијама.		
Фреквенција ажурирања	Колико често се ажурира база података за нове претње.		
Додатне функције	Firewall, родитељска контрола, VPN, заштита веб-камера, итд.		
Подршка и документација	Доступност техничке подршке, упутства, FAQ.		
Цена и лиценцирање	Да ли је бесплатна, пробна верзија, или плаћена. Да ли је цена оправдана.		
Компатибилност	Подршка за различите оперативне системе (Windows, macOS, Linux).		
Рецензије и репутација	Мишљења корисника и експерата, награде и признања.		

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи шта значи појам „структура фајл-система“ и какви типови фајл-система постоје (на пример: FAT32, NTFS, ext4). Проучи добре праксе за организацију великог броја датотека. Сазнај које алатке омогућавају аутоматско именовање, преименовање и сортирање датотека и опшиши њихове основне могућности.

1.9 Лиценцирање софтвера и Creative Commons

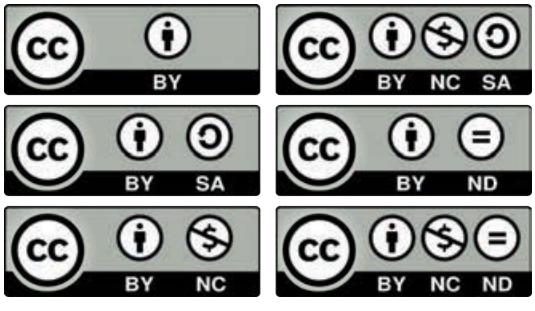
Када наставници и ученици користе различите дигиталне алатке у настави, важно је да провере под којим лиценцамасу те алатке доступне, да ли су отвореног или затвореног кода, да ли је дозвољена њихова измена и прилагођавање, као и под којим условима се могу користити без нарушавања ауторских права. Посебну пажњу треба обратити и на образовне садржаје настале помоћу дигиталних алатки, нарочито алатки које су засноване на ВИ (на пр. аутоматско генерисање текста). Потребно је проверити услове коришћења таквих алатки и утврдити да ли се добијени сасдржај може слободно делити, мењати и објављивати или подлеже ограничењима прописаним лиценцом или условима коришћења.

У свету информатике, софтвер и дигитални садржаји се креирају од аутора који имају одређена права на њихову употребу. Да би се регулисао начин на који други могу да користе те производе, користе се лиценце. Лиценце одређују право на инсталирање, копирање, модификовање и дистрибуирање софтвера или садржаја. Creative Commons представља савремени приступ ка лиценцирању који омогућава ауторима да прослеђују своја дела под јасно дефинисаним условима. Разумевање ових концепата је суштинско за етичко коришћење софтвера, мултимедијалних садржаја и образовних материјала.

Софтверске лиценце су правни уговори између аутора или произвођача софтвера и крајњег корисника. Постоје разне врсте лиценци, од којих свака дозвољава различите нивое коришћења. Комерцијални софтвер обично долази са лиценцом што ограничава коришћење само једне копије, без права на измену или прослеђивање. Такви примери су Microsoft Office, Adobe Photoshop и други професионални пакети. Корисник мора да прихвати лиценцне услове пре него што користи софтвер, најчешће у облику EULA (End User License Agreement).

Са друге стране, постоје и бесплатне лиценце као што је Freeware, које дозвољавају коришћење без новчане надокнаде, али најчешће без могућности за модификацију. Shareware дозвољава пробни период, након којег је потребна регистрација. Open Source (отворени код) код лиценци, дозвољава кориснику да користи, изучава, мења и пласира софтвер. Најпознатији примери су GNU General Public License (GPL), MIT License и Apache License.

Creative Commons (CC) је тип лиценцирања који се најчешће користи за садржаје као што су слике, текстови, презентације, видеа и музика. Уместо потпуног забрањивања, CC дозвољава разне нивое употребе у зависности од избора аутора. Постоји шест основних типова CC лиценци које комбинују услове као што су: обавезно навођење ауторства (BY), некомерцијално коришћење (NC), немењање (ND), и прослеђивање под истим условима (SA).



Слика 12 *Тиūови CC лиценци*

На пример, лиценца CC BY дозвољава коришћење, прослеђивање и модификацију, свѐ док се наводи ауторот. CC BY-NC дозвољава исто, али не у комерцијалне циљеве. CC BY-ND дозвољава прослеђивање, али не и модификацију. Ове комбинације омогућавају флексибилно коришћење ресурса, с поштовањем права аутора.

У образовању CC лиценце се све чешће користе за прослеђивање наставних материја ла. Многи универзитети, библиотеке и истраживачке организације објављују своје садржаје са овим лиценцама, да би их учинили доступним на глобалном нивоу. Ово доприноси отвореном знању, бољој сарадњи и дигиталној солидарности.

Важно је да сваки корисник води рачуна о лиценци софтвера или садржаја који користи. Нелегално копирање, дистрибуција или измена представља кршење ауторских права и кажњиво је законом. Поред правних последица овакво понашање нарушава етику дигиталног грађанства и поткопава рад креатора садржаја.

Постоје и јавни домени (Public Domain), где садржај нема ограничења за коришћење. Ови ресурси су слободни за употребу, без лиценци, јер су ауторска права истечена или их се аутор изричито одрекао. Ипак, и при коришћењу таквих извора, препоручује се да се наведе извор из поштовања према оригиналном креатору.

Лиценцирање представља основни механизам којим се регулише коришћење софтвера и дигиталног садржаја. Познавањем разних врста лиценци, корисници могу одговорно да употребљавају алатке и информације које су им доступне. Лиценце Creative Commons посебно доприносе отвореном прослеђивању, али и постављању јасних услова за то. Поштовањем ових правила гради се дигитална култура базирана на сарадњи, поштовању и правди.

ЗАКЉУЧАК

Лиценцирање софтвера и дигиталних садржаја има велику улогу у томе како корисници смеју да користе, прослеђују и мењају програме, документе, слике или видеа. Софтверске лиценце, као комерцијални, бесплатан, слободан или отворени код, одређују да ли корисник може да мења софтвер, да га дистрибуира или да га само користи под одређеним условима. У оквиру дигиталних садржаја велику важност имају Creative Commons (CC) лиценце. Оне су направљене да помогну аутору да одреди како други смеју да користе његов садржај. Све CC лиценце се базирају на комбинацији дозвола и ограничења, што омогућава аутору да задржи одређена права, а да ипак дозволи слободнокоришћење под одређеним условима. Најчешће Creative Commons лиценце су:

- CC BY (Наведи аутора) – дозвољава коришћење, прослеђивање и измену садржаја, али мора да се наведе оригинални аутор.

- CC BY-SA (Наведи аутора – Проследи под истим словима) – дозвољава измену, али ново дело мора да се проследи под истом лиценцом.
- CC BY-ND (Наведи аутора – Не мењај) – дозвољава коришћење, али не дозвољава измену.
- CC BY NC (Наведи аутора – Некомерцијално) – дозвољава коришћење и измену, али не за комерцијалне циљеве.
- CC BY-NC-SA / CC BY-NC-ND – комбинације некомерцијалног, без мењања и прослеђивања под истим условима.
- CC0 – аутор се одриче свих права, садржај је у јавном домену.

Разликовање CC лиценци омогућава да се правилно користе дигитални ресурси, да се цитирају извори и да се понашају етички и законски када се креира или преузима садржај. Познавање лиценци омогућава да се одлучи како да се заштити сопствени рад и како да се поштује туђи рад.

ПРИМЕР 1

Бесплатни, али ограничени софтвер

Један ученик користи уређивач текста који је бесплатан (Freeware). Иако га не плаћа, не сме да мења његов код нити да га дистрибуира другима, јер услови коришћења то забрањују.

ПРИМЕР 2

Отворен код со лиценца GPL

Наставник информатике користи Linux дистрибуцију са GPL лиценцом. Може да је мења и да је прослеђује другима, све док се нова верзија дистрибуира под истим условима.

ПРИМЕР 3

Презентација са Creative Commons

Наставник креира PowerPoint презентацију са сликама и музиком које је преузео са интернета са CC BY-NC лиценцом. Он је навео ауторе и користи садржај само за едукативне, некомерцијалне циљеве.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни зашто постоје лиценце за софтвер и дигиталне садржаје.

2. Које су разлике између комерцијалног софтвера и отвореног кода?

3. Наведи 3 типа Creative Commons лиценци и објасни их.

4. Наведи разлике између врста лиценци (CC BY, CC BY-SA, CC BY- ND, CC BY-NC, итд.)

5. Шта значи ако је неки софтвер у јавном домену?

6. Какве су последице коришћења нелиценцираног софтвера?

7. Истражи шта је MIT лиценца и за шта се користи.

8. Да ли су сви бесплатни програми и легални за коришћење? Објасни.

9. Пронађи неки софтвер са CC лиценцом и објасни како се користи.

ПРАКТИЧАН ЗАДАТАК

Креирај документ са оригиналним садржајем који има текст (есеј, блог пост, поезија) и слику / графику. Садржај треба да буде едукативан или креативан. Изабери одговарајућу CC лиценцу за њихов садржај. Објасни зашто си изабрао управо ту лиценцу. Шта дозвољава, а шта ограничава?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се лиценцирају мобилне апликације и који су услови коришћења према продавницама као што је Google Play или App Store. Проучи шта је софтверска пиратерија и које су њене правне и етичке импликације.

1.10 Вештачка интелигенција (општа и генеративна)

Вештачка интелигенција (ВИ) представља једну од најзначајнијих и најбрзорастећих технологија у савременом дигиталном свету. Помоћу ње, машине су способне да уче, закључују, комуницирају и решавају проблеме који традиционално траже човекову интелигенцију. Све више система, апликација и уређаја се ослања на некакав облик ВИ – од паметних телефона и навигатора до медицинских дијагностичких алатки и аутономних возила. Ова лекција има за циљ да представи појам, поделу и примену вештачке интелигенције, са акцентом на њена два савремена правца: општа и генеративна ВИ.

Вештачка интелигенција се дефинише као способност компјутерског система да извршава задатке који траже интелигентно понашање – као препознавање говора, решавање проблема, учење од искуства и доношење одлука. У зависности од нивоа интелигенције и способности, постоје две главне категорије: тзв. општа ВИ (AGI – Artificial General Intelligence) и „специјализована“, за одређену намену или генера- тивна ВИ.

Општа вештачка интелигенција има за циљ да створи систем који може да размишља, учи и да се сналази у разним ситуацијама као човек. Овај облик ВИ је још увек у теоретској фази и не постоји у практичној примени. Она би требало да комбинује логику, емоције, контекст и морал у свом функционисању. Истраживања у том правцу су фокусирана на стварање универзалних алгоритама и модела учења који би функционисали независно од задатка или околине.

Иако општа ВИ није развијена на потребном нивоу, наставља да буде активна тема академских и технолошких истраживања. Модели се развијају да симулирају човеково размишљање, са фокусом на машинско расуђивање, разумевање контекста и адаптација на нове ситуације. Циљ је да се створи систем који неће бити ограничен на један задатак, већ ће моћи да ‹размишља› кроз више домена – слично као човек

Име ВИ / Компјутер	Креиран	Шта ради?
за пројекат	Шта ради?	
ХАЛ 9000	2001: Свемирска одисеја	Контролише свемирски брод
Дата (андроид)	Стари трек	Машина са људским размишљањем и емоцијама
Џарвис/ Фрајдеј	Гвоздени човек/Осветници	Лични помоћник који има и емоционалну интелигенцију
Скајнет	Терминатор	Ратни ВИ систем који постаје самосвестан и опасан

Табела 2 *Примери опште ВИ*

Насупрот томе, генеративна вештачка интелигенција представља практични тип ВИ која се данас већ користи на разним пољима. Ова ВИ је специјализована за креирање нових садржаја – текста, слике, музике, видео или чак програмског кода. Генеративни модели као што су они базирани на дубоком учењу (deep learning), обучавају се огромним количинама података и затим се користе за стварање нових, оригиналних израза који наликују на човеково стваралаштво.

Генеративна ВИ је већ присутна у свакодневном животу. На пример, језички модели попут оних који омогућавају аутоматско дописивање реченица у мобилним телефонима, превођење текста, или чак писање целих статија. Визуелни генеративни системи, као што је DALL•E или Midjourney, користе се за стварање уникатних слика на основу текстуалних описа. У музичкој индустрији алгоритми стварају композиције, а у филмској – виртуелне гласове или ликове. Све ове апликације комбинују податке и статистику помоћу креативности која подсећа људски начин размишљања.

Карактеристика	Општа ВИ (AGI)	Генеративна ВИ (GenAI)
Област примене	Универзална (као човек)	Ограничена, специфична
Способност учења	Самостално учење, универзално	Учи од големи податоци
Свесност / разумевање	Потенцијално (теоретски)	Не, само је статистички модел
Пример	Фиктивни компјутер из научне фантастике	ChatGPT, DALL•E, Copilot
Статус	Теорија, још увек не постоји	Реална и коришћена данас

Табела 3 *Поређење опште и генеративне ВИ*

Алгоритми генеративне ВИ најчешће се користе у архитектури то су такозване неуронске мреже – посебно тзв. генеративне супротстављене мреже (GANs) и трансформери. Ови системи уче обрадом милиона примера, при чему извлаче основни образац, структуру и стил садржаја. Затим, користе те „учене" обрасце за креирање нечег новог, нечег што личи на оригинал, али није оригинално. На пример, може да се генерише слика „замка на Месецу" који никада није постојао, али изгледа реално.

Ипак, и поред напретка, постоје бројни изазови повезани са генеративном ВИ. Проблеми као што су нетачне информације наводе на погрешну представу о појави или процесу, лажне слике (deepfakes), непожељну пристрасност и ауторска права за која је потребна посебна пажња. Потребни су етички оквири, закони и алатке за откривање и контролу садржаја који генеришу ове системе, посебно када се користе у образовању, новинарству, или социјалним мрежама.

Вештачка интелигенција у својим општим и генеративним облицима појављује се као алатка са огромним потенцијалом за побољшање разних аспеката живота. Док се општа ВИ још увек развија као визија за будућност, генеративна ВИ већ се користи на бројним пољима. Познавање ових технологија је неопходно за разумевање модерног дигиталног света, али и за његово одговорно коришћење. Одговарајућом применом и регулисањем, ВИ може да се претвори у моћног савезника у образовању, науци, уметности и свакодневници.

Осим тога, вештачка интелигенција се примењује и у персонализованим препорукама садржаја, као што су филмови, музика или производи на платформама као што је YouTube, Netflix и Amazon. Алгоритми прате навике корисника, сакупљају податке и креирају предлоге који олакшавају избор и повећавају задовољство од коришћења. Ови системи постају рафинираније и не само што уче од индивидуалног корисника, већ упоређују са хиљадама других корисника како би понудили најрелевантнији садржај.

У здравству, генеративна ВИ се користи за анализу медицинских података и генерирање извештаја, моделирање молекула за нове лекове, па чак и симулацију ефеката од одређених терапија. На овај начин се убрзава истраживање и побољшава се дијагностика. Неки ВИ системи већ показују способност да препознају симптоме болести са тачношћу сличном или вишом од људских експерата.

При коришћењу ВИ треба да се поштује велики број препорука на глобалном нивоу, законске регулативе ЕУ, али и националне. Моћ коју има у себи не треба да се злоупотребљава. При коришћењу ВИ треба да се поштују следеће препоруке:

1. Приватност и заштита личних података

- ВИ не сме да сакупља или обрађује личне информације без сагласности.
- Подаци морају бити заштићени од злоупотребе, крађе или неовлашћеног приступа.
- Сваки корисник треба да зна који подаци се сакупљају и зашто.

2. Транспарентност (јасност и објашњивост)

- ВИ-системи треба да буду јасни у свом функционисању.
- Корисник треба да зна да комуницира са ВИ, а не са човеком.
- Важно је да ВИ може да објасни како је донела одређену одлуку (на пример: зашто је доделила одређену оцену неком ученику).

3. Фер понашање и одсуство дискриминације

- Подаци са којима се тренира ВИ треба да буду разноврсни и избалансирани.
- ВИ не сме да буде пристрасна према раси, полу, узрасту, религији или другим основама.

Човекова контрола и одговорност

- Корисници треба увек да имају крајњу одговорност за одлуке ВИ.
- ВИ не треба да доноси одлуке без могућности за човекову интервенцију (посебно у осетљивим областима, као што је здравство или образовање).

Безбедност и сигурност

- ВИ-системи треба да буду заштићени од сајбер напада, злоупотреба или грешака.
- Треба редовно да се тестирају и ажурирају да би функционисали безбедно.

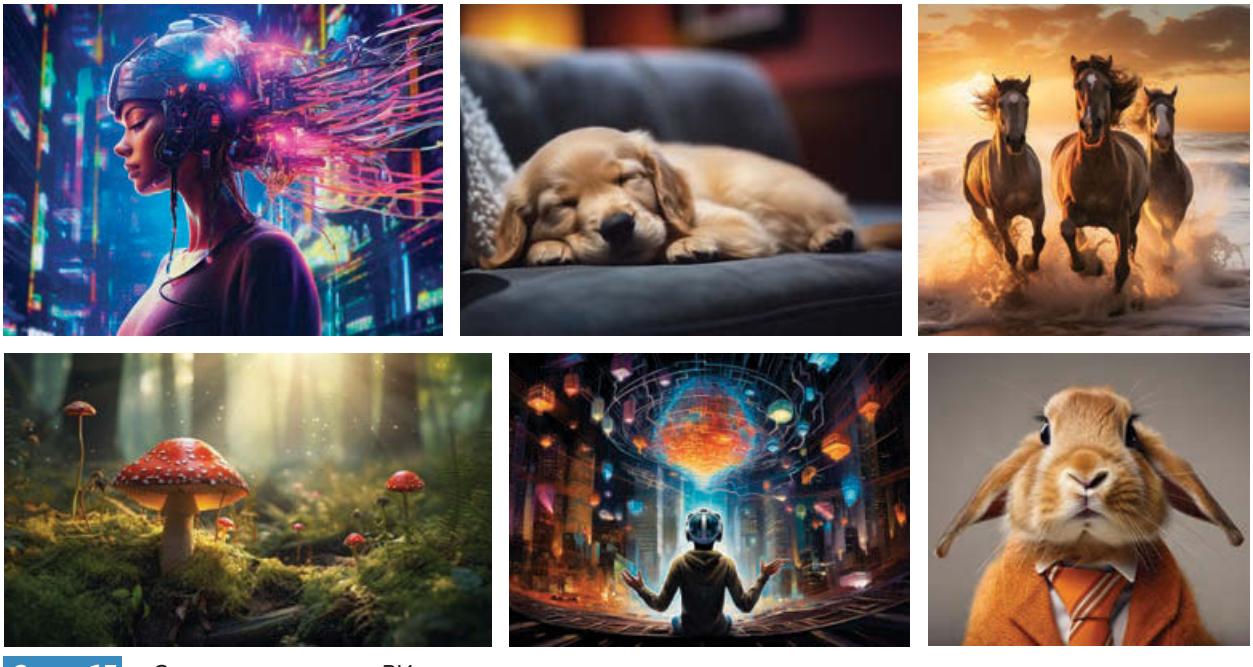
Етичка употреба сходно циљу

- ВИ треба да се користи за добро људи – у образовању, здравству, заштити животне средине итд.
- Не сме се користити за лажне вести, манипулације, надзор без дозволе или војне циљеве против цивила.

Поштовање људског достојанства и права

- ВИ мора да поштује основна људска права, као што је слобода изражавања, приватност, слобода мишљења и сл.
- ВИ не сме да замењује људе на начин који деградира или одузима вредност.

Етичка употреба ВИ значи одговорност, транспарентност, људскост, фер понашање и безбедност. Технологија треба да служи људима, а не обрнуто.



Слика 13 *Слики ґенерирани од ВИ*

ЗАКЉУЧАК

Вештачка интелигенција (ВИ) представља способност компјутерских система да извршавају задатке који обично траже човекову интелигенцију — учење, препознавање слика и говора, доношење одлука и решавање проблема. У свакодневици, ВИ се користи у великом броју ситуација: препоруке на видеима и музика, навигација, паметни асистенти као што је Сири и асистент Гугла, филтери против спама, системи за препознавање лица, аутоматски преводиоци и чат-ботови. Општа ВИ је замишљена као врста ВИ која би могла да мисли и да учи слично као човек, и да се сналази у многим различитим ситуацијама. Она још увек не постоји у реалности, већ је само теоретска идеја.

Истраживачи покушавају да створе системе који ће разумети контекст, који ће се прилагођавати новим условима и решаваће разне задатке, а не само један конкретан посао. Циљ је да се развије интелигенција која би могла да функционише у више области – као човеков ум.

Генеративна ВИ способна је да креира нови текст, слике, музику, видео, програмски код и друге врсте садржаја на основу података којима је обучена. Може да пише есеје, помаже у решавању задатака, генерише слике, креира програмски код и предлаже нове идеје и садржаје.

- **Примери опште ВИ**: аутопилот у аутомобилу, системи за предвиђање временске прогнозе, медицинска анализа, спам филтри, препорука филмова.
- **Примери генеративне ВИ:** ChatGPT, DALL•E, Midjourney, Gemini и други системи за креирање музике или 3Д модела.

Како се повећава примена ВИ, тако се повећава и значај етичких правила за њено коришћење. Потребно је контролисано прослеђивање личних података са системима као и провера информација произведених од ВИ, јер могу бити погрешне. Ауторство је све битније, копирање туђих идеја без дозволе је строго регулисано. ВИ треба да се користи за учење, а не за преписивање.

Етичке дилеме повезане са општом и генеративном ВИ све више добијају на важности. Овде спадају питања приватности, јер неки системи сакупљају много података. Постоји делема у вези тачности, зато што ВИ може да прави грешке или да даје лажне информације. Ризик од зависности од ВИ уместо самосталног размишљања, као и питање да ли је фер да се користе ВИ алатке у школским задацима уколико није дозвољено. Друге дилеме су: како да се спречи злоупотреба генеративне ВИ за креирање дезинформација, лажних фотографија или аудио материјала и да ли је етички да машине замене човека на радном месту.

ПРИМЕР 1

Генерисање текста

Онлајн платформа користи генеративну ВИ да аутоматски направи резиме дугих текстова. Корисник уноси текст, а ВИ га обрађује и приказује крађу, сажету и прегледну верзију.

ПРИМЕР 2

ВИ у медицини

Систем за анализу рендген-снимака користи неуронску мрежу да открије потенцијална обољења плућа. Дијагнозу затим потврђује лекар, који користи ВИ као подршку.

ПРИМЕР 3

Виртуелни уметник

Ученик користи генеративни модел да креира дигитални постер базиран на опису: „фантастичан пејзаж са љубичастим планинама и небом са два сунца“. ВИ креира слику за неколико секунди.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни својим речима шта је вештачка интелигенција.

2. Које су разлике између опште и генеративне ВИ?

3. Наведи три реална примера где се користи генеративна ВИ.

4. Шта представља појам „неуронска мрежа“ и која је њена функција?

5. Истражи које генеративне апликације постоје данас и шта оне омогућавају.

6. Који су етички проблеми повезани са генеративном ВИ?

7. Шта су „deepfake“ видеа и зашто могу бити опасни?

8. Замисли како ће се ВИ користити у учионици у будућности. Опиши неки пример.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се обучавају генеративни језички модели као што је GPT и какви подаци се користе за њихово тренирање. Проучи значење термина етичка ВИ и ВИ транспарентност и зашто су важни за будући развој технологије.

ПРАКТИЧАН ЗАДАТАК

Замисли да твоја школа жели да уведе ВИ систем за:

- аутоматско препознавање лица за улазак у школу,
- праћење ученика на одмору (камерама) због безбедности,
- аутоматско оцењивање писмених задатака са анализом текста.
- Уради етичку процену.

Одговори на следећа питања:

1. Који етички проблеми могу да се појаве код овог ВИ система? (напр. приватност, пристрасност, лажни резултати...)

2. Ко би могао да буде оштећен или дискриминисан и зашто?

3. Ког етичког правила се тиче?

4. Са којим дилемама ћеш се суочити?

5. Како може школа да омогући да се ВИ користи одговорно?

6. Шта мислиш – да ли је добро или није да се овај систем уведе у вашу школу? Зашто?

7. Шта би урадио да се проблем реши?

Користи табелу на слици наниже да би дошао до правилног решења

Бр.	Опис ситуације / дилеме	Погођена етичка		
правила	У чему је проблем? (објашњавање сукоба)	Могућа решења (према етичким правилима)		

Табела 4 *Решавање дилеме*

ПОНАВЉАЊЕ ТЕМЕ 1

1. Наведи три примера дигиталне технологије коју користиш у свакодневном животу!

2. Објасни шта представљају директоријуми и датотеке и зашто су важни за организовање података на компјутеру!

3. Креирај нови директоријум на радној површини под именом Моје фотографије и додај једну фотографију. Затим премести директоријум у твоје документе и преименуј по свом избору!

4. Упореди две савремене дигиталне технологије (на пример: паметан телефон и паметан часовник) и објасни разлике у њиховој намени!

5. Процени које апликације су најодоговарајуће за: писање текста, прављење табела и креирање презентација. Објасни зашто!

6. Осмисли кратка корисничка упутства за прилагођавање основних поставки оперативног система (ажурирања — провера и инсталација најновијих системских исправки, корисничких рачуна и лозинки, постави јаку лозинку локалног корисника са 2FA, повезивање на Wi-Fi мрежи, укључивање фајрвола, креирање дозвола да само неопходне апликације имају приступ до камере/микрофона/локације, иницијализација антивируса и провера система, оптимизација екрана за учење, јасност, контраст, подешавање режима мировања/искључивање екрана и уклањање непотребних програма и апликација)!

7. Наведи две разлике између опште (General AI) и генеративне вештачке интелигенције!

8. Објасни како дигитална технологија утиче на образовање, комуникацију или рад у савременом друштву!

9. Анализирај како се различити оперативни системи (Windows, Android, Linux) разликују у изгледу, радном окружењу и могућностима!

10. Замисли да треба да имплементираш генеративну АИ у школи. Предложи три начина како би могла да се користи, објасни како функционише и који су могуће користи и ризици!

ТЕМА 2: Онлајн живот

Данас је веома лако дописивати се са пријатељем који живи у другој земљи, да гледаш филм онлајн, заказујеш лекарски преглед путем апликације, или да играш виртуелну игру са људима из целог света — све то се дешава захваљујући компјутерским мрежама и интернету. У данашње време, свет је повезан као никада до сада. Без разлике да ли је реч о мобилном телефону, лаптопу, паметном часовнику или индустријском роботу — сви ови уређаји „разговарају“ међусобно путем мрежа. Управо зато, познавање компјутерских мрежа и интернета није само корисна вештина, већ је и један од најважнијих корака ка разумевању света око нас.

Нови појмови

- компјутерска мрежа, сервер, клијент, LAN, WAN, интернет, веб, „сурфање“ на интернету, интернет-сервиси, www, ftp, машине за претраживање (search engines), електронска пошта, видеопозиви, е-трговина, електронско банкарство, интерактивно комуницирање, дигитални отисак, насиље, мржња, промотивни материјал за коришћење наркотика или самоповређивање, дезинформације, лажне вести, подешавање безбедности и приватности, правни аспекат, механизми за пријављивање, благостање.

Већ знаш да...

- разликујеш веб-претраживач и веб-прелистач;
- приступаш до интернет-ресурса и да претражујеш потребне информације;
- препознајеш валидне изворе информација на вебу;
- препознајеш позитивне и негативне стране „дигиталног отиска“;
- комуницираш преко интернета.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

Патујућа авантура

Стојиш на аеродрому, пртљаг ти је спреман, а у руци имаш магичну карту која те води до било ког места на свету. Где ћеш отићи? Изабери неку земљу или град коју си одувек волео/ла да посетиш. Каква је њена култура – језик, традиција, храна? Како можеш да стигнеш тамо (летови, цене, пут)? Колико би те коштао боравак 3 дана (нађи реалне примере)? Које три ствари треба да видиш или урадиш тамо?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објашњаваш појмове компјутерска мрежа и интернет;
- описујеш интернет као средство за добијање и прослеђивање информација;
- разликујеш разне облике интернет сервиса (услуга);
- користиш валидне изворе информација на вебу;
- анализираш позитивне и негативне стране „дигиталног отиска“ који остављаш;
- креираш дигитални садржај од пронађених података и информација.

2.1 Појам компјутерске мреже

Компјутерска мрежа подразумева постојање најмање два компјутера који су међусобно повезани како би могли да комуницирају и да размењују ресурсе. Под ресурсима се подразумевају хардверске компоненте (компјутер, сервер, скенер, принтер и друго) и софтверске компоненте (игре, базе података, апликације). Компјутери се повезују у мрежу како би се повећала ефикасност и смањили трошкови (штеде време и новац). Ови уређаји могу да буду повезани преко физичких каблова или преко бежичне комуникације, зависно од инфраструктуре и потребе корисника.

2.1.1 Намена компјутерских мрежа

Преко компјутерских мрежа корисници прослеђују податке, софтвер, хардвер и комуницирају једни са другима.

Прослеђивање података и информација. У мрежном окружењу можеш да приступиш подацима из других компјутера у мрежи. Тако на пример, преко компјутерске мреже у некој компанији, сви запослени могу да приступе заједничком календару састанака или бази података о клијентима. У компјутерској мрежи која је постављена у кућним условима, један члан породице може да проследи филм или документ на свој компјутер, а други чланови могу неометано да приступе тим садржајима, користећи таблет или други компјутер, односно не треба да их копирају на свом уређају.

Прослеђивање софтвера. У компјутерској мрежи је могуће користити софтвер који се извршава на другим компјутерима. На пример, на неком компјутеру (сервер) у школи може да се инсталира програм за графички дизајн. Уместо да сваки ученик има инсталирани програм на свом компјутеру, он може да приступи том програму преко школске мреже и да ради на својим пројектима.

Прослеђивање хардвера. У мрежном окружењу неколико корисника може да прослеђује хардверске ресурсе (на пример, сервер, штампач, скенер). Често пута у компанијама или домаћинствима постоји један штампач бољег квалитета који је повезан са неколико компјутера преко мреже и са сваког од ових компјутера може да се штампа на том штампачу.

Комуникација. Компјутерске мреже су промениле начин комуникације код људи – данас се најчешће користе е-пошта, социјалне (друштвене) мреже, интернет, телефон итд. Ове мреже омогућавају да комуникација буде бржа, економски исплатљива и доступна у реалном времену, без обзира на физичку удаљеност. Захваљујући мрежама можеш да разговараш са пријатељима у иностранству, да пратиш онлајн наставу, да радиш на заједничким пројектима са људима из целог света и да прослеђујеш фотографије и видео садржаје за само неколико секунди.

2.1.2 Интернет

Интернет је глобална компјутерска мрежа која обезбеђује различита информацијска и комуникацијска средства, састављена је од међусобно повезаних мрежа које користе стандардизоване комуникацијске протоколе. Интернет користи клијент-сервер архитектуру.

Компјутер-сервер је специјално конфигурисан компјутер који прима захтеве клијената и обезбеђује услуге, ресурсе или информације другим компјутерима преко мреже. Стално је активан. Сервери су јаке машине са великом процесорском снагом, великим простором за чување података, дизајнирани за сигурност и непрестани рад, често у центрима за податке. Могу бити виртуелне машине, облачне услуге или софтверске компоненте.

Примери услуга које може да нуди сервер су:

- чување и прослеђивање датотека (фајл сервер),
- приказивање веб-страница (веб-сервер),
- слање и примање е-поште (имејл-сервер),
- управљање подацима (сервер за базе података).

Компјутер-клијент је уређај (лаптоп, телефон, таблет, претраживач и др.) који иницира комуникацију са сервером с циљем да се добије нека информација или услуга. Пример: Када неки корисник користи интернет-прелистач да посети веб-страницу, његов компјутер је клијент, а компјутер који хостира веб-страницу је сервер. У компјутерским мрежама увек постоји двосмерна комуникација између клијента и сервера.

Клијент (на пример, веб прегледач) шаље захтев серверу који проналази и враћа тражени садржај. Ова комуникација се одвија преко протокола као што је HTTP, HTTPS, TCP/IP. Сваки уређај на интернету има IP адресу – јединствени број који служи за идентификацију и усмеравање података.

Осим тога, ове мреже се конфигуришу разним поставкама у зависности од потреба – на пример, бројање активних уређаја, контрола приступа, приоритизација саобраћаја и мониторинг активности. У образовним институцијама, компјутерске мреже омогућавају централизовану администрацију, контролу приступа интернету, прослеђивање ресурса и сарадња између ученика и наставника. На глобалном нивоу, интернет је састављен од стотине хиљада међусобно повезаних мрежа које користе заједнички језик – TCP/IP протокол – да би се међусобно разумели.

DNS (Domain Name System) претвара људска имена веб-страница (на пр. www.wikipedia.org) у машински читљиве IP адресе. Ово чини интернет приступачнијим и лакшим за коришћење. Пакети података шаљу се кроз мрежу преко разних рута, а сваки корак се контролише од рутера и прекидача који омогућавају ефикасан пренос.

Рутери имају централну улогу у усмеравању података преко интернета. Они су одговорни за избор најбоље руте за сваки пакет који се шаље, како би брзо и сигурно стигао до одредишта. Прекомерно оптерећење или прекид на појединим рутама може изазвати кашњење или губитак података. Зато протоколи укључују механизме за проверу, поновно слање и исправљање грешака.

2.1.3 Врсте компјутерских мрежа

Да би се разумела структура и примена компјутерских мрежа, мреже се класификују према величини и обухватању – најчешће у две главне категорије:

- **LAN (Local Area Network)** – локална мрежа,
- **WAN (Wide Area Network)** – мрежа распрострањена на географској области (разна насеља-исти град, разни градови, и сл.)

Локална мрежа – LAN

LAN представља локалну компјутерску мрежу која омогућава међусобно повезивање разних уређаја – компјутера и периферних уређаја у оквиру ограничене географске области као што су домаћинства, учионице, школе, канцеларије, или цела зграда. Може бити једноставна (састављена од два компјутера) или сложена (састављена од стотину компјутера и периферних уређаја у некој великој компанији).

WAN компјутерска мрежа

Компјутерска мрежа је распрострањена на географској области. Примењује се за међусобно повезивање удаљених компјутера. Компјутерска мрежа уједињује и повезује мање мреже да би корисници могли на једној мрежи да комуницирају са корисницима друге мреже или да користе ресурсе на удаљеној локацији.

Карактеристика	LAN	WAN
Покријеност	Мала област	Велика растојања
Брзина	Брза (Gbps)	Спорија од LAN (довољно брза)
Цена	Мала	Велика (изнајмљивање линија, посебна инфраструктура)
Сопственост	Приватна (организација, дома, и сл.)	Јавна или приватна (инфраструктура интернет-провајдера)
Опрема	Свичеви, рутери	Оптичке линије, сателити, телеком мреже
Управљање	Једноставно	Сложено, са подршком од провајдера
Пример	Домаћа мрежа	Компанија са више удаљених сектора, интернет-мрежа

Табела 1 *Разлике између локалне и глобалне LAN и WAN мреже*

НАПОМЕНА:

Компјутерске мреже не треба да се поистовећују са интернетом. На пример, догађа се да су у школском кабинету компјутери умрежени и да размењују податке међу собом, али да ова мрежа није повезана са интернетом.

ЗАКЉУЧАК

Компјутерске мреже омогућавају повезивање више компјутера и уређаја с циљем прослеђивања информација, ресурса и услуга. Сервер обезбеђује услуге, податке или ресурсе, док клијент користи те услуге преко слања налога. Разлика између локалних и глобалних мрежа показује колико широко уређаји могу да буду повезани — од мале учионице до целог света.

ПРИМЕР 1

LAN мрежа у школи

У школи су све учионице повезане преко локалне мреже (LAN) која омогућава приступ интернету, централном принтеру и заједничкој бази података са наставним материјалима.

ПРИМЕР 2

DNS и претраживање веб-страна

Када ученик унесе „www.wikipedia.org“ у веб-прелистач, DNS сервер претвара ову адресу у IP број и усмерава је ка одговарајућем веб-серверу где се налази садржај.

ПРИМЕР 3

Коришћење Wi-Fi мреже код куће

У кућним условима, уређаји се повезују бежжично преко Wi-Fi рутера. Он усмерава интернет-саобраћај између мобилног телефона, лаптопа спољне интернет конекције.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Објасни појмове компјутерска мрежа, компјутер сервер и клијент?

2. У чему је разлика између LAN и WAN мреже?

3. Наведи два примера за сервере који се користе у свакодневном животу.

4. Користиш интернет-прелистач да би отворио веб-страну. Који уређај је клијент, а који сервер у тој ситуацији?

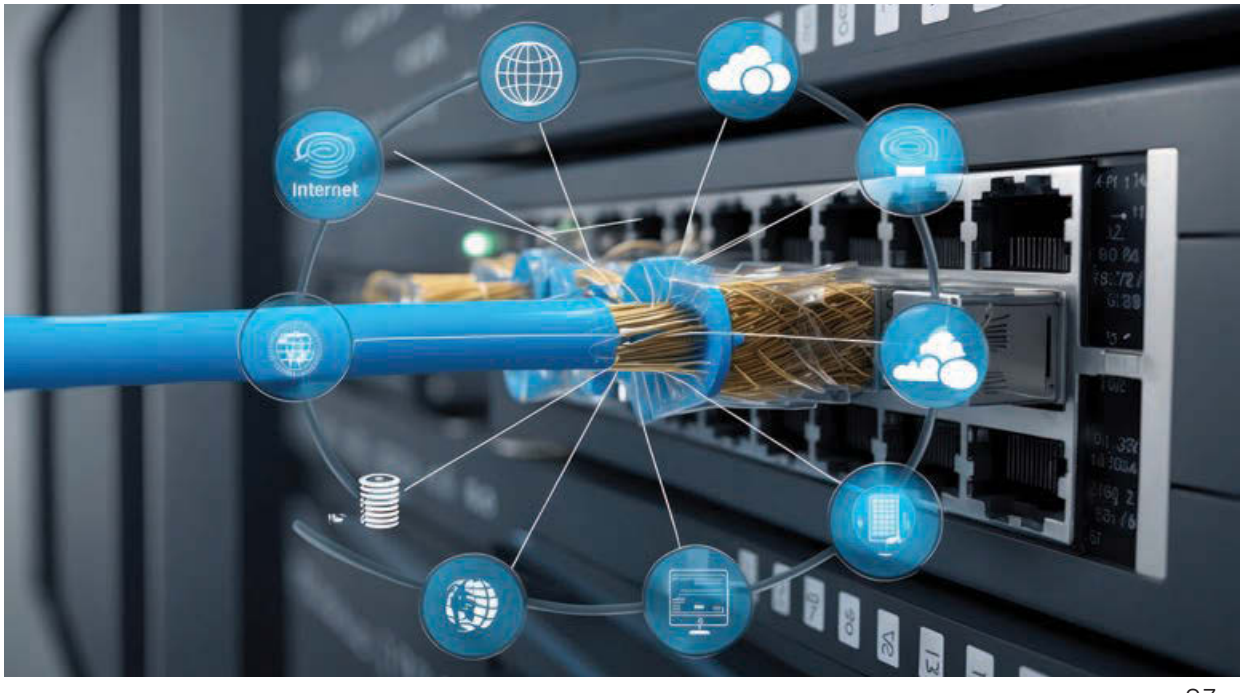
5. Зашто је важно да сервери буду стално активни, а клијенти не?

6. Да ли сваки компјутер може да постане сервер? Образложи твоје мишљење.

7. Осмисли једноставну мрежу за неку школу. Опиши како би организовао сервере и клијенте.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи шта се дешава у позадини кад унесеш веб-адресу у прелистач и притиснеш Enter. Покушај да направиш мапу процеса преноса података из твог уређаја до сервера и назад. Осим тога, разгледај какве каријере могу да се граде у областима повезаним са мрежама – као мрежни администратор, инжењер за сајбер-безбедност или ИТ подршку.



2.2 Интернет-сервиси

Интернет омогућава приступ огромном спектру услуга које олакшавају свакодневни живот. Ови интернет сервиси се користе за комуникацију, информисање, образовање, куповање, банкарство и много више. Разумевање њихове структуре, намене и функционисања је кључно за правилно, безбедно и ефикасно коришћење. У овој лекцији ће се објаснити најважнији интернет сервиси, како функционишу и које су њихове предности и ризици.

WWW

Најчешће коришћен интернет сервис је World Wide Web (WWW). Он омогућава прегледање веб-страна са текстом, сликама, видео и интерактивним садржајима. Веб-странама се приступа преко веб-прелистача (на пример, Chrome, Firefox, Edge), а садржаји се преносе коришћењем HTTP и HTTPS протокола.

Веб-странице се налазе на серверима (веб сервери) и представљају групу повезаних веб страна. Свака локација на вебу има јединствену адресу такозвану URL (Uniform Resource Locator) адресу. WWW је коришћен за учење, истраживање, забаву и комуникације преко форума, портала и социјалних мрежа. Ови интернет сервиси се развијају великом брзином, при чему се непрестано додају нове функционалности да би се побољшало корисничко искуство. Корисници треба непрестано да се информишу о новим могућностима и начинима безбедне употребе, посебно у случајевима када се тражи унос личних или финансијских података.

Важно: WWW није исто што и интернет – то је услуга која ради преко интернета. Док интернет представља мрежу повезаних компјутера, WWW је збир апликација или дигиталних садржаја којима приступамо путем те мреже.

Машине за претраживање (search engines)

Данас на интернету постоје милиони веб-страна, што њихово проналажење понекад чини сложеним и дуготрајним. Интернет је постао огроман, непрегледен и практично неисцрпан извор информација. За лакше претраживање садржаја, можеш да користиш веб-директоријуме (који групишу стране према тематским областима) или, што је чешћа пракса, да претражујеш уз помоћ машина за претраживање (search engines).

Уношењем налога у веб-претраживач (Google, Edge, Yahoo, Bing) састављен од кључних речи, активира се алгоритам за претраживање на интернету. Као одговор на тај налог, претраживач приказује списак веб-страна које садрже кључне речи из налога. Сваку од наведених веб-страницу у резултатима претраге можеш да отвориш кликом на њен назив. Уз назив су обично приказани и веб-адреса и кратак опис садржаја странице.

Електронска пошта

Електронска пошта је један од најстаријих и најкоришћенијих сервиса на интернету, који омогућава размену текстуалних порука и разних врста датотека између корисника путем компјутерских мрежа. Представља дигитални облик традиционалне поште, али са много већом брзином, удобношћу и доступности.

Да би електронска пошта могла да се користи, сваки корисник треба да има своју електронску адресу. Корисник бира своју имејл адресу, имејл сервер му је одобрава.

За сваку електронску адресу додељује се електронско сандуче у коме се чувају електронске поруке корисника. Имејл сервер је компјутер који је повезан на интернету и који прима и шаље електронске поруке корисницима електронске поште.

Имејл адреса се састоји од два дела: корисничко име и име имејл сервера. Ова два дела су одвојена знаком @ ("at" који се чита ет, називају га и мајмунче). Корисничко име је део имејл адресе који корисник сам бира приликом њеног креирања. Оно може да се користи само ако већ није заузето од стране другог корисника на истом имејл серверу. Део адресе који се налази псоле знака @ представља назив имејл сервера преко кога се обавља размена електронске поште. На пример, у имејл адреси maja_ matevska@t-home.mk корисничко име је maja_ matevska, а назив сервера је t-home.mk.

При раду са електронском поштом имејл адресе са малим и великим словима су исте.

пример: biljana@gmail.com=BILJANA@GMAIL.COM.

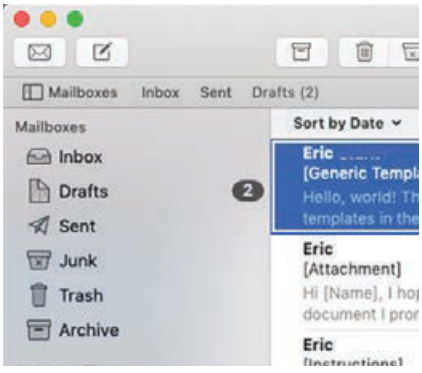
Структура електронске поруке:

- адреса пошиљаоца **(From):** аутоматски се попуњава
- адреса примаоца **(To):** пише се у адресном делу
- **Carbon Copy (cc):** уноси се имејл-адреса другог лица које ће примити копију поруке, а примаоц да је порука послата и неком другом
- **Blind carbon copy (bcc):** примаоц поруке не зна да се копија поруке шаље и лицима чија адреса је наведена у bcc пољу
- наслов поруке – тема **(Subject):** кратак опис садржаја. Примаоц поруке осим имена (адресе) пошиљаоца и датума види и предмет поруке. На основу тих података примаоц поруке одлучује о томе да ли ће и када ће прочитати поруку
- текст поруке: пише се у делу текста поруке. Текст поруке зависи од тога коме је намењен прилог **(Attachment):** фајл који се додаје у поруци. Може бити документ, слика, презентација, програм...
- потпис **(Signature).**

Слика 1 *Електронска поруке*

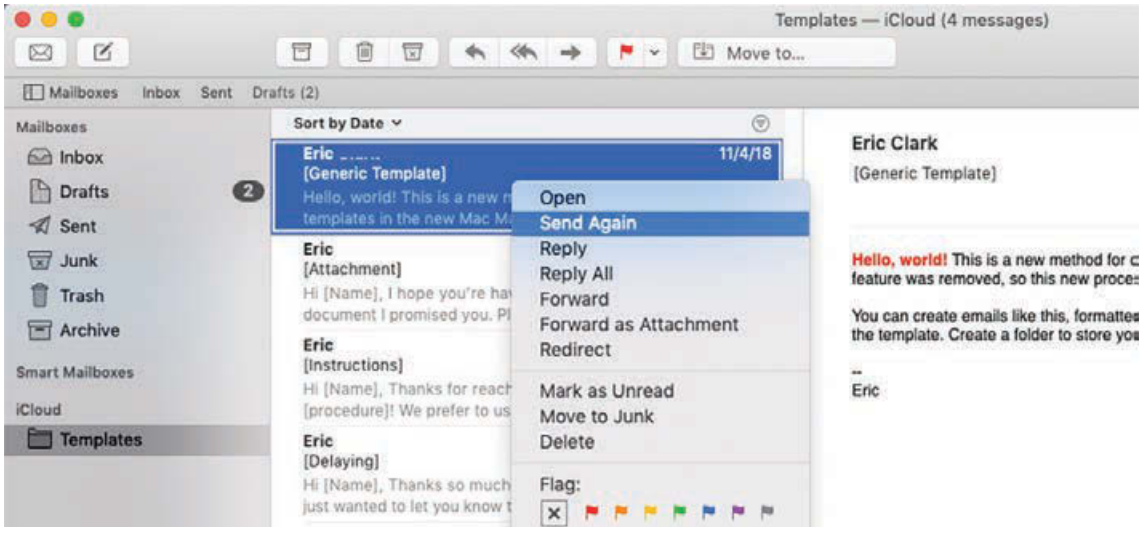
Сандучићи за електронску пошту најчешће садрже стандардни број директоријума (фасцикла) као што су:

- **Inbox** (фасцикла са примљеним порукама, која се назива поштанско сандуче),
- **Sent** (фолдер са послатим порукама),
- **Trash** (или Deleted, фолдер са обрисаним порукама),
- **Drafts** (фолдер за непослате поруке у смислу уређивања).



Слика 2 *Структура на електронско сандаче*

Сви програми за рад са е-поштом омогућавају организовање е-поште. Корисници могу да креирају нове фолдере, да премештају и копирају поруке из једног фолдера у други и да дефинишу правила на основу којих ће поруке бити сортиране у одређеним фолдерима одмах по пријему.



Слика 3 *Целосна структура на електронско сандаче*

Програми електронске поште имају следеће функције:

- Креирање електронске поруке – Нова порука (New);
- Слање поруке – Пошаљи (Send);
- Одговор на примљену поруку – Одговор (Reply);
- Прослеђивање примљене поруке другој особи – Проследи (Forward);
- Брисање поруке – Обриши (Delete);
- Штампање поруке – Штампај (Print);
- Убацивање других докумената у поруку – Окачи (Attach);
- Коришћење адресара.

ftp

FTP је скраћеница од File Transfer Protocol, што значи протокол за пренос датотека. То је један од најстаријих интернет-сервиса који омогућава слање и преузимање датотека између два компјутера преко интернета – најчешће од клијента (твог компјутера) до сервера и обрнуто.

Видеопозиви

Видеопозиви су начин комуникације преко интернета где можемо да се видимо и чујемо уживо са неким. За то ти је потребно:

- уређај – компјутер, лаптоп, таблет или мобилни телефон
- камера (уграђена или спољна) – за слику
- микрофон и звучници (или слушалице) – за звук
- интернет-конекција – боља конекција значи квалитетнији позив
- апликација или платформа за видеопозиве.

Видеопозиви су део наше свакодневице. Можеш да их користиш за онлајн наставу, пословне састанке, породичне разговоре, онлајн интервјуа и обуке и слично. Популарни сервиси за видеопозиве су: Zoom, Microsoft Teams, Google Meet, Viber, WhatsApp.

Други сервиси који се све више користе су облачне услуге (Cloud Services) које омогућавају снимање података на интернет серверима и приступање са било које локације, као што су: Google Drive, OneDrive, Dropbox, затим стриминг услуге (Streaming Services) за гледање видеа, слушање музике или пренос уживо преко интернета. Најпознатије су YouTube, Netflix, Spotify и Twitch.

Електронска трговија

У данашњем дигиталном свету захваљујући развоју интернета и модерних технологија, све више људи бира да купи производе и услуге онлајн, са само неколико клика на својим уређајима. Овај облик трговине назива се електронска трговина (или е-трговина).

Електронска трговина омогућава комуникацију и размену производа између купца и продавца преко интернета, без разлике на физичку удаљеност. Безбедне веб-странице, онлајн-продавнице, мобилне апликације и електронска плаћања су само део алатки које чине ову трговину брзом, практичном и доступном. Овај облик трговине обухвата:

- продају робе и услуга индивидуалним клијентима од стране компанија - B2C (business to customer),
- продају робе и услуга од стране компанија другим компанијама - B2B (business to business),
- продају од стране продавца који појединачно продају своју робу и услуге индивидуалним клијентима - C2C (customer to customer).

Многе стране данас омогућавају куповање и продају разних производа. Куповање се реализује тако што купац плаћа робу и услуге кредитном картицом, готовином при испоруци или стандардном уплатницом. Продавац доставља робу поштом или курирском службом. Већина светских давалаца услуга омогућава електронску трговину. На пример, данашње путовање може у потпуности да се организује преко интернета. Авионске, аутобуске или карте за воз најлакше се купују преко веб-страница саобраћајних компанија. Корисник добија електронску карту, коју може да користи и у електронском и у штампаном облику (не препоручује се) и да путује авионом, аутобусом или возом својим пасошем. Хотели и хостели могу да се резервишу и преко њихових веб-страна или још безбедније, преко добро познате веб-странице специјализоване за овакве услуге.

При коришћењу услуга е-трговине, посебна пажња треба да се посвети безбедности. С обзиром на то да је реч о трошењу правог новца, непажња може да има веома непријатне последице.

Електронско банкарство

Данас банке нуде услуги за електронско банкарство. Електронско банкарство (или е-банкарство) представља коришћење интернета и дигиталних уређаја како би се приступило банкарским услугама. То омогућава корисницима да управљају својим финансијама преко компјутера, мобилног телефона или банкомата, без потребе да физички одлазе у експозитуру.

Након што се пријавиш у систем, можеш да провериш стање на свом рачуну, вршиш уплате, исплате, као и да преносиш средства са једног рачуна на други рачун. Услуга електронског плаћања рачуна је веома практична, јер омогућава плаћање од куће, без одласка у банку или пошту и чекања у реду. Безбедност је при томе од великог значаја, па банке користе различите механизме заштите, као што су шифровање података, двофакторска аутентификација и мобилни токени.

Интерактивно комуницирање

Интернет сервиси омогућавају интерактивно комуницирање. Осим разговоре уживо (Chat), видеопозива и видеоконференција, интернет-комуникација укључује и форуме и онлајн дискусије као и социјалне (друштвене) мреже.

Форуми су веб-странице које омогућавају корисницима да дискутују (обично само на теме утврђене правилима форума). Дискусија је организована на тему – неко започиње низ на одређену тему објављивањем прве поруке, а даља дискусија на ту тему наставља се одговарањем на поруке из тог низа.

Блогови (weB LOG – блог) су један облик личних дневника у којима корисници објављују своја размишљања о одређеној теми и прослеђују јавности. Корисници читају статије објављене од аутора блогова и могу јавно да коментаришу и да дискутују о њима.

Социјалне (друштвене) мреже су креиране средином прве деценије 21. века и веома брзо су постале неки облик социолошког феномена. За мање од једне деценије, број активних корисника порастао је на неколико милијарди. Социјалне мреже омогућавају корисницима да се повежу и да успоставе контакт са профилима њихових личних пријатеља и познаника или са профилима личности из јавног живота. Социјалне мреже имају своје појединачне карактеристике:

Facebook – највећа социјална мрежа која омогућава прослеђивање статуса, фотографија, видеа и повезивање са пријатељима, групама и страницама.	**Instagram** – платформа за прослеђивање фотографија и кратких видео садржаја, популарна међу младима. Користи се и од инфлуенсера и бизниса.	**TikTok** – мрежа за кратке, креативне видео садржаје са музиком, ефектима и изазовима.
Twitter (сада X) – платформа за кратке текстуалне објаве („твитови“), идеална за вести, мишљења и брзу комуникацију.	**Snapchat** – апликација за слање фотографија и видеа који исчезавају након прегледа. Популарна код тинејџера.	**LinkedIn** – професионална мрежа за умрежавање, тражење посла и пословну комуникацију.

YouTube – иако је видеоплатформа, има и функције социјалне мреже преко коментара, објава и заједница.	**Pinterest** – мрежа за проналажење инспиративних слика и идеја које су најчешће повезане са модом, уметношћу, храном, дизајном итд.	**Reddit** – форумска платформа на којој људи дискутују о разним темама, постављају питања и деле мишљења.

Важан аспекат социјалних мрежа је приватност корисника, а при постављању статуса (на пример, фотографија од ваших најближих), треба да се има у виду да они постају видљиви за шири круг људи. Уколико не желиш да твоје информације буду јавно видљиве, боље је да их поставиш тако што не може да их види нико осим твојих пријатеља. Број корисника на социјалним мрежама доказује да су оне изузетно лаке за коришћење.

Интернет је одлично место за учење, истраживање, дружење и забаву. Али, као и у реалном свету и тамо постоје ризици. Зато је важно да поштујеш правила за безбедну комуникацију, да би заштитио себе и друге. То предвиђа да никада не размењујеш личне податке са непознатим лицима, да избегаваш кликање сумњивих линкова и да се понашаш учтиво и одговорно.

При креирању онлајн налога (на пр. за е-пошту, школски портал или апликацију) мораш да будеш пажљив и да водиш рачуна о својој безбедности. При регистрацији многе веб-странице траже корисничко име и лозинку. При овом кораку, најважније је да изабереш безбедну лозинку – која треба да буде довољно дугачка (најмање 12 знакова) и да садржи комбинацију малих и великих слова, бројева и специјалних знакова (на пр. @, #, $, %). Није препоручљиво да користиш информације које лако могу бити откривене, као што је датум рођења или име кућног љубимца. Исто тако, веома је важно да не делиш лозинку са другим лицем. Препоручљиво је да лозинку редовно мењаш и да користиш разне лозинке за различите профиле. На овај начин се смањује ризик од злоупотребе података.

Безбедна комуникација је темељ одговорног коришћења интернета и учења у безбедној онлајн средини.

ЗАКЉУЧАК

Интернет сервиси представљају основу савременог дигиталног друштва, омогућавајући брз и једноставан приступ информацијама, комуникацијама и разним услугама. Они трансформишу нашу свакодневицу, побољшавајући начине рада, учења, забаве и социјалне интеракције. Сталним развојем технологија, интернет-сервиси постају све интегрисанији и незаменљиви у савременом животу, омогућавајући глобалну повезаност и иновације које обликују будућност.

ПРИМЕР 1

Коришћење е-поште за школске циљеве

Ученик од свог наставника добија домаћи задатак преко е-поште. Он отвара поруку, преузима окачени документ, решава задатак и шаље га назад са новим прилогом. Овај процес се одвија брзо и ефикасно, без физичког сусрета

ПРИМЕР 2

Онлајн куповање преко е-трговине

Корисник жели да купи слушалице преко интернета. Посећује веб-сајт продавнице, бира производ, уноси податке за испоруку и плаћа преко интернет банкарства. Ускоро добија потврду и број за слање пошиљке.

ПРИМЕР 3

Коришћење услуга у облаку

Наставница чува све своје наставне материјале на Google Drive. Креира заједнички фолдер са ученицима, где они могу да прегледају документе, коментаришу и уређују пројекте у реалном времену са било ког уређаја.

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Шта су интернет-сервиси и која је њихова главна улога?

2. Који су најчешћи облици интернет-сервиса које користиш у свакодневном животу?

3. Како интернет-сервиси утичу на комуникацију међу људима?

4. На које начине интернет-сервиси олакшавају учење и образовање?

5. Које су највеће предности коришћења интернет-сервиса?

6. Какве безбедносне ризике могу да носе интернет-сервиси?

7. Како технологија утиче на развој интернет-сервиса?

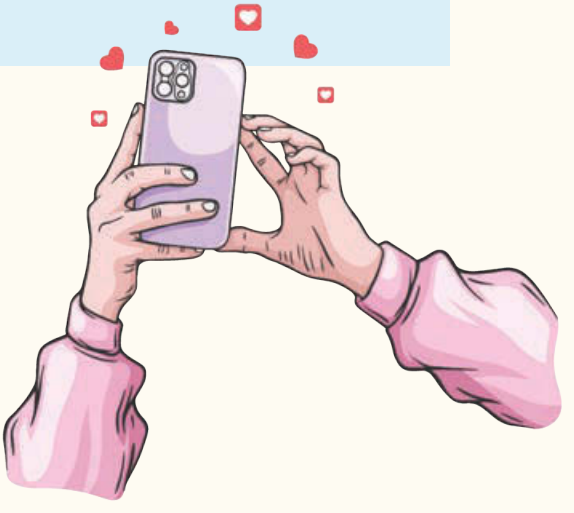
8. Зашто је важно да се има приступ интернет-сервисима у савременом друштву?

9. Које примере интернет-сервиса можеш да набројиш?

10. Како интернет-сервиси доприносе глобалној повезаности?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како функционишу системи стриминга (како YouTube и Netflix) – од чувања података до преноса видео садржаја до твог уређаја. Осим тога, разгледај како се интернет сервиси користе у едукацији, на пример преко Moodle, Google Classroom или Microsoft Teams, и које су њихове предности у учењу на даљину.



2.3 Развој веб-технологија

Интернет се стално развија, а свака његова фаза представља нову генерацију технологија и начине коришћења веб-технологија. Када говоримо о еволуцији и иновацијама на интернету, најчешће се користе појмови Web 1.0, Web 2.0, Web 3.0, Web 4.0 и Web 5.0 – термини који описују пут кроз који се интернет мењао и наставља да се мења. Ове фазе нису одвојене јасним границама, већ се надовезују једна на другу.

Web 1.0 је прва генерација веба, која доминира деведесетих година прошлог века и почетком двехиљадитих. У овој фази, веб-стране су статичке, што значи да је садржај фиксни и не може да се мења од стране корисника. У овој фази постоји мала или такорећи никаква интеракција са корисницима који су пасивни посматрачи. У овој фази веб-страница је једносмерна – од креатора према кориснику. Примери ове фазе су енциклопедије као што је Britannica.com које садрже текст без могућности за коментаре или уређивање као и прве верзије Yahoo! – само листа линкова и информације, без интеракције.

Доласком веба 2.0, интернет почиње да се креће према двосмерној комуникацији – од корисника према вебу, али и обрнуто. Сада је интернет динамичан, интерактиван и са могућношћу за сарадњу. Корисници више нису само читаоци, већ и креатори садржаја. Примери из ове фазе развоја веба су: социјалне мреже (платформе на којима свако може да објављује, коментарише, прослеђује и комуницира), Wikipedia (енциклопедија коју свако може да уређује), Blogspot и WordPress (платформе за блогирање где људи прослеђују мишљења, приче и савете). Ова фаза је донела велики напредак, али и нове изазове – као што је злоупотреба података и дезинформације.

Web 3.0 представља следећу велику промену у начину на који се комуницира, прослеђују информације и користе дигиталне услуге. Ова нова фаза се разликује од претходних генерација по томе што омогућава децентрализацију, већу приватност и аутономију корисника. Уместо да подаци буду складирани на централизованим серверима, они су распоређени преко блокчејн технологија (једноставно речено – систему у коме сви имају копије података, и нико не може да их промени без да сви примете). То значи да уместо да се подаци чувају на једном централном серверу (као што је банка или компанија), они су распоређени међу многим компјутерима у свету. Ова фаза обухвата и вештачку интелигенцију и представља покушај да се веб направи „паметан“. У овој фази укључени су гласовни асистенти као што је Siri, Alexa и Google Assistant који иако су централизовани, јер су њихови подаци и функционалност контролисани од великих компанија (Apple, Amazon, Google), ипак, користе вештачку интелигенцију (AI) и машинско учење за боље разумевање корисничких налога. Будућност Web 3.0 може да доведе до демократскијих AI асистената који неће зависити од великих компанија.

Пример за ову фазу је – Brave Browser, веб-прелистач који не прати кориснике и награђује их. Web 3.0 је „семантички веб“ – не само подаци, већ подаци са значењем.

Web 4.0 представља следећу еволуцију где интернет није само паметан, већ и предвидљив и аутономан. Ова генерација се базира на интернету ствари (IoT), где уређаји међусобно комуницирају и самостално доносе одлуке, често без људске интервенције. Web 4.0 укључује интернет који „размишља“ (анализира податке и предлаже најрелевантније информације још пре него буде питан), природна комуникација (комуникација са гласовним асистентима и виртуелним помоћницима као са живим човеком) и уређаје (паметне фрижидере, аутомобиле, часовнике) који ће комуницирати међусобно и помагаће у свакодневном животу. На пример, твој часовник може да каже твом аутомобилу да се загреје пре него што изађеш. Као примери из ове фазе могу се навести паметне куће – системи који аутоматски регулишу осветљавање, температуру или безбедност, као и аутомобиле тесла који уче и возе самостално уз помоћ вештачке интелигенције.

У претходним фазама интернета, корисници су углавном претраживали информације (Web 1.0), комуницирали и креирали садржаје (Web 2.0), добили су паметније системе и децентрализовали платформе (Web 3.0 и Web 4.0). Али, Web 5.0 иде корак даље — он разуме људске емоције и омогућава потпуну контролу над твојим дигиталним идентитетом. Иако је још увек у раној фази развоја, Web 5.0 представља визију интернета који не само што „мисли“, већ и „осећа“. Реч је о фази у којој веб није само користан већ и емпатичан. Пример из ове фазе је виртуелни асистент који препознаје када си тужан и нуди савете и помоћ, чак и одговарајућу музику за боље расположење.

ЗАКЉУЧАК

Интернет-сервиси представљају основу савременог дигиталног друштва, омогућавајући брз и једноставан приступ информацијама, комуникације и разне услуге. Они трансформишу нашу свакодневицу, побољшавајући начине рада, учење, забаву и социјалну интеракцију. Сталним развојем технологије, интернет-сервиси постају интегрисанији и незаменљивији у савременом животу, омогућавајући глобалну повезаност и иновације које обликују будућност.

ПРИМЕР 1

Статична веб-страна од Web 1.0

Једна школска веб-страна из 1999. године приказује само информације о распореду часова без могућности интеракције или коментара. Садржај је статичан и ретко се мења.

ПРИМЕР 2

Социјална мрежа у Web 2.0

Ученици користе Instagram за прослеђивање фотографија и комуникацију. Они креирају садржаје, коментирају, прате друге и добијају повратне информације из заједнице.

ПРИМЕР 3

Гласовни асистент у Web 4.0

Ученик користи Google Assistant да би поставио аларм, претражује дефиницију или жели да закаже састанак. Асистент разуме глас и даје персонализовани одговор на основу навика корисника.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Које су главне карактеристике сваке од фаза развоја веб-технологија?

2. Наведи примере за сваку од фаза развоја веб-технологија!

3. Шта је „корисничко генерисани садржај“ и у којој веб-генерацији се појављује?

4. Шта значи „семантички веб“ и са којом генерацијом се повезује?

5. Да ли можеш да наведеш уређај из свакодневног живота који је део Web 4.0?

6. Да ли постоји јасна граница између Web 2.0, Web 3.0 и следећих генерација? Зашто?

7. Како се мења улога корисника кроз Web 1.0 до Web 5.0?

8. Шта мислиш, која веб-генерација је највише променила начин живота? Објасни зашто.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се користи блокчеин (blockchain) технологија у Web 3.0 за обезбеђивање транспарентности и безбедности. Осим тога, разгледај етичке аспекте употребе вештачке интелигенције у Web 4.0 и Web 5.0 – као што су дигитално благостање, приступ технологијама и заштита менталног здравља.

2.4 Веб како извор информација

Интернет је огроман извор информација које долазе из разних извора као што су владине институције, универзитети, медијумске куће, блогови, форумске дискусије, социјалне мреже. Део ових извора су научни и проверени, а други су неофицијални и можда нетачни. Како ћеш знати који извори су кредибилни и веродостојни, односно како ћеш знати да ли су информације које читаш тачне и сигурне?

Да би оценио да ли је нека веб-страница кредибилна, постави себи следећа питања:

- Ко је објавио информацију? Да ли информација долази из познатог и сигурног извора, као што је школа, универзитет, државна институција или познати медијум?
- Да ли има аутора? Да ли је наведено име аутора и шта ради (да ли је новинар, научник или експерт)?
- Када је текст објављен? Да ли је написан недавно или је веома стар? Да ли су информације још увек важеће и тачне?
- Одакле су преузете чињенице у тексту? Да ли су споменути извори или линкови до других страница које потврђују то што је написано?
- Како је текст написан? Да ли текст има пристрасан тон – покушава да те убеди у нешто снажним емоцијама или пристрасношћу, или је објективан и фактографски?

Кад будеш утврдио да нека веб-страница изгледа безбедно и сигурно, следећи корак је да провериш да ли је тачна и сама информација коју читаш. Да би то урадио треба да:

- је упоредиш са другим изворима – провераваш да ли и друге странице потврђују ову информацију.
- тражиш чињенице и доказе – да ли има објашњења, бројке или научне податке које поткрепљују ту информацију.
- будеш изузетно обазрив на преваре – пазиш на лажне вести, дезинформације и наслове који желе само да те натерају да кликнеш (т.н. „clickbait“).

За успешно претраживање на интернету, поред техничких аспеката претраживања, веома је важно да се подстакне медијумска писменост код корисника – односно способност да се разликују чињенице од мишљења, истините од лажних информација, и објективни извештаји од пристрасних интерпретација. Критичко вредновање извора подразумева не само процену техничких параметара, већ и контекстуалну анализу: ко објављује текст, са каквом намером, и да ли постоје алтернативне верзије исте информације. Информацијске манипулације, дезинформације и 'лажне вести' често су ширене преко социјалних мрежа и лажних веб-страница. Зато се препорачује употреба проверених медија, провера чињеница путем платформи као што је Snopes, FactCheck.org или локални извори за верификацију, као и консултација више извора при анализирању одређеног догађаја. Ученици треба да развију навике да не прихватају прву информацију коју ће пронаћи, већ да је упореде са другим изворима и да процене да ли постоји логички континуитет, подршка реномираних аутора или институција, и да ли су изнесени подаци проверљиви.

Веб претраживање представља основну вештину у дигиталној ери, али само претраживање није довољно без одговарајуће процене извора. Потребно је да се примењује критичко размишљање при анализирању садржаја, њен аутор, структура, и контекст. Применом напредних техника за претраживање и процену кредибилитета, добијају се сигурније информације које могу да се користе за учење, истраживање и информисање. Само оваквим приступом могу да се избегну манипулације, дезинформације и погрешни закључци који могу да утичу на доношење одлука.

ЗАКЉУЧАК

Интернет је моћна алатка у учењу, али само ако знаш како да препознаш праве и сигурне информације. Пажљивом проценом извора и поређењем информација, можеш да се заштитиш од лажи и превара и да доносиш правилне закључке. Веб се користи не само као претраживач, већ као критичан мислилац.

ПРИМЕР 1

Академско претраживање

Ученик/ученица припрема презентацију за тему „климатске промене". Уместо да користи прве резултате, он претражује уз помоћ наводника („климатске промене у Македонији") и додаје site:.gov.mk да би добио официјалне изворе.

ПРИМЕР 2

Провера веродостојности

Корисник наилази на вест на социјалној мрежи која изгледа сензационалистички. Проверава садржај на независним медијима и користи платформу за проверу чињеница да би открио да ли је вест тачна.

ПРИМЕР 3

Филтрирање резултата

Наставник претражује PDF материјале о дигиталној писмености. Користи Google са операторима: filetype:pdf intitle:„дигитална писменост“ site:.edu.mk, да би добио тачне и релевантне документе.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Шта значи да се интернет користи као извор информација?

2. Шта је кредибилан (сигуран) извор? Дај пример.

3. Како можеш да препознаш да ли је неки веб-сајт сигуран?

4. Зашто је важно да знаш ко је написао информацију?

5. Шта значи да информација буде неутрална? Како ћеш препознати пристрасан текст?

6. Шта је „clickbait“ и зашто треба да пазиш?

7. Која су три најважнија питања која треба да одговориш када провераваш веб-информације?

8. Шта треба да урадиш ако нађеш различите информације на две веб-странице?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како функционишу претраживачи са техничког аспекта – шта је web crawler, како се индексирају веб-странице и који алгоритми се користе за рангирање. Осим тога, разгледај концепт ‘информацијског мехура’ (filter bubble) и како утиче на резултате које добијамо кад претражујемо на интернету.

2.5 Дигитални отисак

Увек када користиш интернет – када претражујеш нешто на Google, објављујеш фотографију на Instagram, гледаш видео на YouTube или чак само кликаш на неком линку – остављаш траг иза себе. Тај траг се назива дигитални отисак. Твој дигитални отисак је твоја дигитална историја – и може да каже много о теби.

Дигитални отисак може детаљније да се објасни као збир свих информација које нека особа оставља на интернету – било свесно (као што су профили на социјалним мрежама, објаве, фотографије), било несвесно када систем аутоматски сакупља информације без да унесемо нешто конкретно (као што су подаци о локацији, претраживања, колачићи, или активност на апликацијама). То значи да не само што активно можеш да прикажеш себе у дигиталном простору, већ и пасивно остављаш податке иза себе – као што су тзв. колачићи (cookies) који следе твоје навике на интернету.

Дигитални отисак има своје позитивне и негативне стране. Позитивне стране дигиталног отиска:

- Помаже при приступу услугама као што су онлајн куповина и банкарство.
- Може да помогне у грађењу позитивне слике о теби – позитивни отисак може да ти помогне да добијеш стипендију, праксу или посао зато што многи универзитети, организације и компаније проверавају шта има о некоме на интернету.
- Интернет памти твоје интересе и нуди ти релевантне садржаје.
- Омогућава комуникацију и грађење мрежа са људима који имају сличне интересе са којима можеш да сарађујеш и од којих можеш да учиш.

Негативне стране дигиталног отиска:

- Твоји подаци могу бити злоупотребљени или праћени без твоје сагласности.
- Све што објављујеш на интернету може да остане доступно заувек. Фотографије, коментари, видеа или објаве које су ти некада изгледале забавно, касније могу да те прикажу у лошем светлу.
- Непромишљене објаве могу да утичу на слику о теби у будућности. Непримерни садржај, увредљиве изјаве или непоштовање других може да утиче негативно на твој углед.

Одговорно коришћење интернета и пажљиво бирање онога што ћеш проследити може да ти помогне да оставиш позитиван и снажан дигитални отисак који ће бити корисан и за твоје образовање и твоју будућу каријеру. Зато:

- Пре него што проследиш нешто, запитај се: „Да ли би моји родитељи, наставници или будући послодавац хтели да виде ово?“ Да ли ово може да ми наштети у будућности?“
- Провери поставке за приватност на твојим апликацијама и профилима. Не треба цео свет да зна о теби.
- Не прослеђуј личне податке (као што је адреса, телефон, лозинке, подаци о картицама) – нити своје, нити туђе.
- Пази кога прихваташ за пријатеља. Неко може да изгледа пријатељски онлајн, али да нема добре намере.
- Изгради добар дигитални идентитет. Објављуј садржаје који показују твоје интересе, пројекте, идеје.

Поред свесног и несвесног начина остављања дигиталног отиска, важно је да се разуме како се ове информације користе. Компаније користе податке за таргетирани маркетинг – то значи да се корисницима приказују рекламе према њиховим интересима, претходним претраживањима или локацијама. На пример, уколико корисник претражује обућу, већ следећег дана може да добија рекламе за ципеле на више веб-страна. Ово је резултат алгоритама који анализирају активност и достављају релевантне садржаје. Ипак, иако персонализација може да буде корисна, она отвара питања надзора и дигиталне слободе. У многим случајевима, корисници нису информисани како, од кога и до ког степена су праћени. Најчешће су ови процеси наведени у дугим политикама приватности које се ретко читају. Током времена креира се комплексан дигитални профил који може да се прода трећим странама или да се користи за политичко таргетирање, манипулацију мишљењем или комерцијалне активности ван контроле самог корисника. Ови ризици наглашавају потребу за едукацијом, свесности и коришћењем алатки за заштиту приватности, као што су блокатори за праћење, VPN, енкриптирана комуникација и честе провере дозвола на апликацијама.

Још једна значајна компонента личне приватности је начин на који се подаци чувају и обрађују. Платформе сакупљају огромне количине информација – укључујући лична документа, фотографије, дописе, локације, уређаје који се користе и обрасце понашања. Ови подаци се складирају на серверима који могу да се налазе у разним земљама, при чему правна регулатива може бити различита и недовољна. Корисници често не знају које компаније имају приступ њиховим подацима, под којим условима и за који период. Чак и када су профили избрисани подаци могу да остану у резервним копијама или да буду продати другим организацијама. Ова питања се регулишу законима као што су Општа регулатива за заштиту података (GDPR) у Европској унији, али је потребно глобално усклађивање да би се обезбедила ефикасна заштита. На индивидуалном нивоу корисници треба да буду пажљиви при прослеђивању информација, да користе јаке лозинке, да не прихватају све дозволе без размишљања и да редовно прегледају своје дигиталне навике и поставке.

Дигитални отисак представља моћни показатељ навика, интересе и идентитет сваког корисника интернета. Иако је понекад користан за персонализацију и побољшање услуга, носи озбиљне ризике ако се злоупотребљава. Одговорност за заштиту личне приватности није само на институцијама, већ и на самим корисницима који треба свесно и пажљиво да се понашају онлајн. Коришћењем безбедносних алатки, критичким размишљањем и дигиталном писменошћу, сваки појединац може да контролише свој дигитални отисак и да заштити своју приватност у дигиталном свету.

ЗАКЉУЧАК

Дигитални отисак је моћна алатка, али треба да знаш како да управљаш њоме да би избегао ризике. Правилним коришћењем интернета, можеш да заштитиш твоје податке и да изградиш позитивну онлајн-репутацију.

ПРИМЕР 1

Несвесно остављање отиска

Ученик/ученица се најављује на неколико апликација са истом лозинком и дозвољава приступ локацији и контактима. Подаци се складирају и прослеђују према маркетинг мрежи без његове експлицитне дозволе.

ПРИМЕР 2

Селфи и показивање личних података

Корисник објављује селфи слику на социјалној мрежи у реалном времену са локацијама и тагираним пријатељима. Овакво објављивање оставља дигитални траг који може да се анализира и злоупотреби.

ПРИМЕР 3

Употреба VPN

Наставник користи VPN када путује да би заштитио личне податке од јавних Wi-Fi мрежа. Тиме се енкриптира комуникација и ограничава се остављање пасивног дигиталног отиска.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Шта значи појам „дигитални отисак“?

2. Који облици информација су део твог дигиталног отиска?

3. Зашто је важно да будеш пажљив када нешто прослеђујеш на интернету?

4. Како можеш да оставиш позитиван дигитални отисак?

5. Састави своју листу правила за одговорно коришћење интернета и остављање позитивног дигиталног отиска!

6. Да ли је избрисана фотографија заувек избрисана? Објасни где се чувају подаци на интернету.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како функционишу алгоритми који прате понашање на интернету – на пример cookies, tracking pixels, fingerprinting и data mining. Осим тога, сазнај како компаније сакупљају податке преко мобилних апликација и шта представља појам ‘подаци као валута’. Разгледај алатке за заштиту приватности као што су DuckDuckGo, Tor, Signal и друге платформе за безбедну комуникацију.

2.6 Опасности на интернету

У данашњем свету интернет је незаобилазни део наше свакодневице. Омогућава брзи приступ информацијама, комуникацију без граница и бескрајне могућности за учење и забаву. Захваљујући интернету можеш да се повезеш са људима из целог света, да истражујеш нове теме и да побољшаваш своје знање.

Поред многих повољности интернет крије и бројне опасности које могу да имају озбиљне последице на твој лични живот и друштво.

Једно од најчешћих опасности је дигитално насиље – намера да се понизи нека особа, да се повреди, да се нанесе штета путем употребе дигиталне технологије, тј. мобилних телефона и интернета. Дигитално насиље може да се манифестује на више начина. Неки од најраспрострањенијих облика дигиталног насиља су уцене, претње, узнемиравање, сексуална злоупотреба преко интернета, сајбер малтретирање, креирање и коришћење лажних профила, као и злоупотреба фотографија других људи.

Други ризик су садржаји који садрже мржњу, подстичу дискриминацију и нетрпељивост према одређеним групама људи због њихове расе, религије, пола или других карактеристика. Ови садржаји негативно утичу на суживот и разумевање између људи и могу да доведу до озбиљних конфликата.

На интернету могу да се сретну садржаји који отворено или прикривено промовишу штетна и опасна понашања, као што је употреба наркотика, злоупотреба алкохола, нарушивања у исхрани (анорексија, булимија) или чак и самоповређивање и самоубиство. Ови садржаји могу бити у облику видеа, блогова, постова на социјалним мрежама или музички текстови, и често се представљају као „начин за справљање са болом" или као нешто „нормално". Поруке које преносе могу да изгледају привлачно, бунтовнички или чак као израз личног идентитета, али заправо носе озбиљне ризике за ментално и физичко здравље. Ови садржаји могу да створе искривену слику о реалности, да подстакну самоуништавајуће понашање или да смање осећај вредности и самопоштовање код младих.

Веома су опасне и дезинформације и лажне вести, које се намерно шире да би збуниле људе или да им наштете. За дезинформацију можемо да кажемо да је то

Слика 4 *Крађа података*

намерно лажна или погрешна информација која се шири с циљем да се људи преваре, да се манипулише јавно мишљење или да се некоме нанесе штета. За разлику од ње, лажна вест је измишљена или погрешно представљена вест, која изгледа као да је права. Може да буде направљена са или без намере да се наштети. Ови садржаји често изгледају као праве вести – имају наслов који привлачи пажњу, фотографије и форматиране текстове који делују убедљиво. Али, уколико нису проверене, оне могу да изазову озбиљне последице. На пример, лажне информације о болестима или лековима могу да угрозе здравствену безбедност, посебно ако људи престану да прате препоруке медицинских лица. У време криза као што су природне катастрофе или пандемије, дезинформације могу да створе панику, неповерење и хаос.

На крају да не заборавимо и штетне компјутерске програме, вирусе, који могу да оштете твој компјутер или да украду твоје личне податке. Они се често крију на сумњивим веб-странама, линковима у е-пошти или непровереним апликацијама. Неки од њих могу да закључе твоје податке, чак могу тихо да прате твоје кретање на Интернету и да сакупљају осетљиве информације, као што су лозинке или банкарски подаци.

Зависност од интернета и дигиталних уређаја је још једна опасност која се све чешће среће, посебно код младих корисника. Дуготрајна изложеност екранима, играма, социјалним мрежама или платформама за видео-садржаје може да доведе до нарушавања сна, недостатка концентрације, социјалне изолације и смањене физичке активности. Игровна зависност је посебна категорија која је већ призната од Светске здравствене организације као ментално нарушавање. Карактеристично је да лице осећа снажан импулс да игра, чак и онда када то нарушава његово здравље, школске обавезе или социјалне односе. Сличне ефекте има и претерано коришћење социјалних мрежа, које може да доведе до поређења са другима, анксиозности или незадовољства собом. Промовисање дигиталне равнотеже, увођење времена без уређаја и активности ван екрана су неке од најважнијих стратегија за спречавање зависности.

„Ризични програми“ и „ризични садржаји“ који могу негативно да утичу на твоју безбедност, ментално здравље, приватност или понашање, обухватају:

непроверене апликације за преузимање које могу да садрже вирусе или шпиунски софтвер,

- игре са неприхватљивим сценама насиља или говора мржње,
- видеа, текстове или коментаре који вређају људе према вери, раси, полу, сексуалној оријентацији и сл.
- Платформе које представљају коришћење наркотика као моде или самоповређивање као начин за справљање са емоцијама.
- бесплатне „хакерске“ алатке којима се „откључавају“ одређени програми који нису бесплатни,
- онлајн-изазови који можда изгледају забавно, али су опасни и деструктивни и др.

Зато размисли кад користиш интернет:

- Да ли се тражи од тебе прослеђивање личних података?
- Да ли те садржај наводи да направиш нешто опасно или незаконито?
- Да ли негативно утиче на твоје расположење или начин на који видиш себе и друге?
- Да ли је извор сумњив или анониман?
- Како би поступио кад доживиш негативно искуство на интернету?

Увек користи интернет са фокусом на садржаје, критичко размишљање и осећај одговорности – према себи и другима.

ЗАКЉУЧАК

Интернет је одлично место за учење, дружење и забаву – али истовремено може да буде и простор у коме се крију озбиљни ризици. Зато је важно да будеш пажљив и одговоран корисник.

ПРИМЕР 1

Фишинг порука е-поштом

Ученик добија е-пошту у којој пише да је добио награду и да треба да кликне на линк за потврду. Након кликања уноси личне податке који се затим злоупотребљавају.

ПРИМЕР 2

Зависност од игрица

Средњошколац проводи по 6 часова дневно играјући онлајн игре, због тога запоставља школске обавезе, успављује се ујутру и избегава дружење са пријатељима.

ПРИМЕР 3

Ширење лажне вести

Корисник прослеђује сензационалистички напис о „открићу“ што се касније доказује као потпуно лажна информација. То изазива панику код неких његових пријатеља.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Наведи три примера злонамерних програма и интернетских садржаја.

2. Како могу дезинформације да утичу на људе и друштво?

3. Објасни разлику између дезинформације и лажне вести.

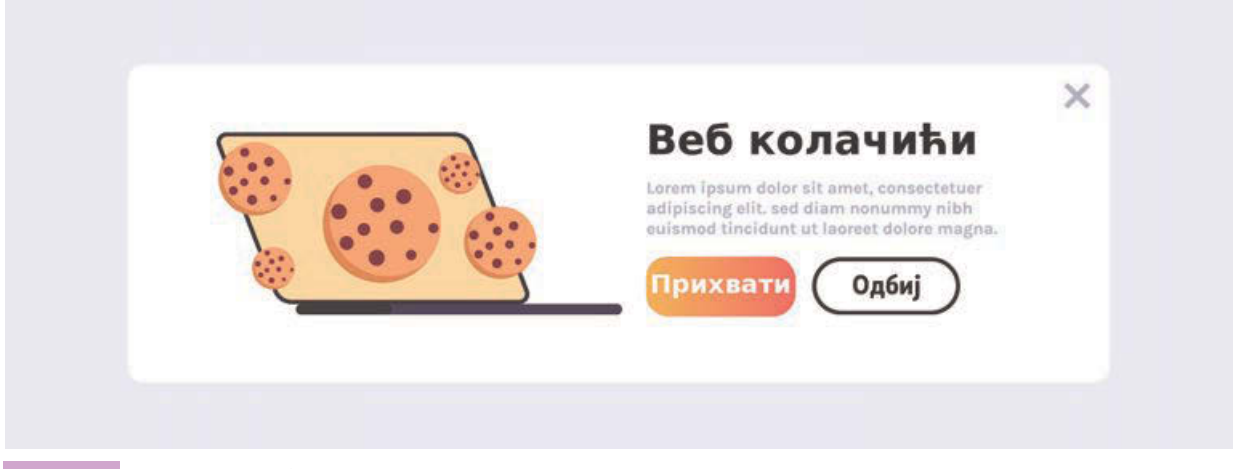
4. На који начин насилни садржај испуњен говором мржње на интернету може да утиче на тинејџере?

5. Зашто је важно да препознамо промотивни материјал повезан са наркотицима или самоповређивањем?

6. Шта можеш да урадиш ако откријеш садржај који те узнемирава или те забрињава?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи шта су „dark patterns“ – технике које веб-странице користе да натерају кориснике даураде нешто што можда не би урадили добровољно (на пример: прихватање свих колачића, куповање производа, пријављивање билтена). Успут, анализирај случајеве када су дезинформације довеле до реалних последица у друштву, као што су протести, паника или финансијске преваре.



Слика 5 *Услови коришћења веб-странице*

2.7 Безбедно коришћење интернета

Безбедност на интернету обухвата многе аспекте, укључујући разумевање савремених технологија, заштиту личних података и паметно и правилно понашање на интернету. Бити безбедан онлајн значи да користиш интернет на начин који је користан за тебе, а истовремено да будеш заштићен од разних ризика и претњи.

Најважнија је заштита личних података. Прослеђивање информација на интернету треба да се врши свесно, са разумевањем да подаци који се објављују – као што су адресе, слике, навике и мишљења – могу бити искоришћени ван контекста или злоупотребљени. Сваки корисник има право на приватност, али и одговорност да поштује то право код других. Није етички да се снима, прослеђује или користи лични садржај других лица без њихове сагласности. Истовремено, важно је да се препознају лажни и манипулативни обрасци на интернету – од лажних огласа до сумњивих порука – и да се не прослеђују нити подржавају такви садржаји. На тај начин доприносиш стварању сигурније и веродостојније дигиталне средине.

Следећи савети ће ти помоћи да заштитиш свој онлајн-идентитет и избегнеш ризике:

- Користи јаке и уникатне лозинке – важно је да имаш разне и комплексне лозинке за сваки твој налог. Користи најмање 12 знакова, комбинуј велика и мала слова, бројеве и специјалне знаке, избегавај очигледне речи, или личне податке као што је име или датум рођења.
- Пази на фишинг преваре и злонамерне линкове – никад не кликај на сумњиве линкове у е-пошти или поруке од непознатих извора. Фишинг је покушај крађе твојих личних података преко лажних веб-страница или порука, зато увек проверавај поверљивост извора.
- Активирај двофакторску аутентикацију (2FA) – ово је допунски безбедносни слој који тражи, поред лозинке, и додатни код (на пример, од мобилног телефона). Са 2FA, чак и ако неко сазна твоју лозинку, неће моћи лако да приступи твом профилу.
- Заштити твоје уређаје – редовно ажурирај софтвер и оперативни систем, јер та ажурирања често садрже безбедносне поправке. Користи антивирусне програме који ће открити и блокирати потенцијалне претње.
- Проверавај да ли веб-страница започиње са „https://", што указује на безбедну конекцију. Избегавај коришћење јавних Wi-Fi мрежа за сензитивне активности, као што су банкарске операције или куповање преко интернета.
- Преузимај апликације, филмове и музику од проверених и безбедних веб-страница и одговарајућих твом узрасту.
- Постави одговарајуће поставке за приватност уређаја и платформе које користиш.
- Више платформи и веб-страна има могућност пријављивања неприкладних садржаја и њихово обележавање или злоупотреба од злонамерних корисника.

- Прилагоди опције прослеђивања информација да би ограничио приступ непознатим лицима. Буди обазрив какве податке прослеђујеш и с ким.
- Уколико се суочиш са непријатном ситуацијом као што је сајбер насиље или онлајн предатор разговарај са поверљивом личношћу или са твојим родитељима који ће ти помоћи у њеном разрешавању.

Постоје различити програми који ће ти помоћи да заштитиш компјутер од опасности вируса, малициозних софтвера и хакерских напада. Следећи примери се користе за разне функционалности:

- **Антивирусни програми**: Касперски, Аваст, Битдифендер, (Kaspersky, Avast, Bitdefender) – откривају и уклањају вирусе.
- **Фајрвол (заштитни зид)** који контролише интернет саобраћај и спречава неовлашћени приступ.
- **Анти-малвер програми, Малвербајтс (Malwarebytes)** – заштита од шпијунских софтвера и опасних апликација.
- **Филтри за садржаје** – родитељске контроле и филтри који блокирају непожељне веб-странице.
- **Програми за ажурирања оперативног система**, са редовним надградњама повећавају безбедност система.

Ови програми помажу да компјутер ради безбедно и стабилно.

Безбедан рад компјутера подразумева да га користиш правилно да ти не би утицао на физичко и ментално здравље. Наниже је наведено неколико активности које одржавају благостање.

1. **Одмори очи од рада пред екраном**, прави паузу на сваких 30–40 минута. Прошетај, протегни се или одмори очи.
2. **Постави здраве навике односно** ограничи време проведено на социјалним мрежама. Исто тако, не користи уређаје пре спавања (минимум један сат).
3. **Разговарај са родитељима/наставницима.** Ако се суочиш са непријатношћу или онлајн узнемиравањем, разговарај са одраслим.
4. **Дигитално благостање** буди љубазан у онлајн комуникацијама, пријави непримерни или опасан садржај.
5. **Води рачуна** о менталном здрављу. Не упоређуј се са људима на интернету – многи садржаји су нереални. Користи интернет за учење, хоби и развој, не само за забаве.

НАПОМЕНА:

У РС Македонији:

- Сагласно Закону о заштити личних података, сваки корисник има право на заштиту својих личних података. При прослеђивању или обради података, треба да се поштује приватност, а лични подаци да се користе само у сагласности или према законским овлашћењима.
- Кривични законик предвиђа санкције за сајбер-кривична дела, као што су сајбер-насиље, ширење вируса и злонамерни софтвер.

- Закон о електронским комуникацијама у Републици Северној Македонији игра важну улогу у омогућавању безбедног коришћења интернета преко стварања оквира који помаже да се обезбеди заштићена, сигурна и одговорна употреба интернета у земљи.

НАПОМЕНА:

Дан безбедног интернета слави се глобално у фебруару сваке године како би се промовисало безбедно и позитивно коришћење дигиталне технологије за децу и младе.

ЗАКЉУЧАК

Паметним и правилним коришћењем интернета, можеш да искористиш све његове предности, а истовремено да се заштитиш од потенцијалних ризика. Корисник треба да буде добро информисан, пажљив и одговоран, с циљем да интернет остане безбедан простор за свакога.

ПРИМЕР 1

Поштовање личне приватности

Ученик/ученица прави групну фотографију школског догађаја, али пре него што је објави на социјалним мрежама, тражи дозволу од свих присутних и поставља ограничења приступу.

ПРИМЕР 2

Пријављивање увредљивих садржаја

Корисница примећује дискриминаторски коментар у јавној групи и одлучује да то пријави на платформи, уместо да одговори са истом агресивношћу.

ПРИМЕР 3

Укључивање у едукативну дискусију

Ученик се укључује у онлајн школску дебату и изражава супротно мишљење, али то ради на учтив и аргументован начин, подстичући позитиван дијалог.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Која три програма или алатке се користе за одржавање безбедности при коришћењу компјутера?

2. Објасни зашто је важно да се поштују правила за безбедно коришћење интернета према законској регулативи!

3. Примени једно правило за безбедност и објасни како би поступио/ла ако добијеш сумњиву поруку или линк!

4. Анализирај две ситуације и утврди у којој постоји већи ризик по безбедност корисника:

(а) прослеђивање лозинке са пријатељем или

(б) преузимање програма од непознате веб-стране. Објасни зашто!

5. Процени да ли коришћење социјалних мрежа више од пет сати дневно може да утиче на опште благостање! Аргументуј свој став!

6. Осмисли пет правила за безбедно коришћење интернета која би препоручио/ла ученицима у твом одељењу!

ПОНАВЉАЊЕ ТЕМЕ 2

1. По чему се разликују компјутерска мрежа и интернет?

2. Наведи три интернет сервиса (услуге)!

3. Шта значи појам „дигитални отисак“?

4. Објасни својим речима како интернет омогућава добијање и прослеђивање информација!

5. Зашто је важно да се користе валидни и проверени извори веба?

6. Објасни једну позитивну и једну негативну последицу дигиталног отиска!

7. Пронађи неку веб-страну за коју сматраш да је валидни извор информација и наведи два разлога њене поверљивости!

8. Дај пример како би користио електронску пошту или облак услугу за прослеђивање школског пројекта!

9. Опиши како би се заштитио од неке конкретне опасности на интернету (на пример: вирус, лажни профил, phishing порука).

10. Упореди два вида интернет услуга: на пример, електронска пошта и социјалне мреже. У чему су сличне, а у чему се разликују?

11. Анализирај зашто неке веб-стране нису безбедне (препознај три знака).

12. Објасни како дигитални отисак може да утиче на будуће могућности неке особе!

13. Процени да ли је следећи извор валидан ако има следеће елементе: веб-страна са много реклама, без аутора и без датума! Објасни твоју процену!

14. Да ли је етички прихватљиво да се прати слика неког друга на социјалним мрежама, без његовог знања? Објасни зашто!

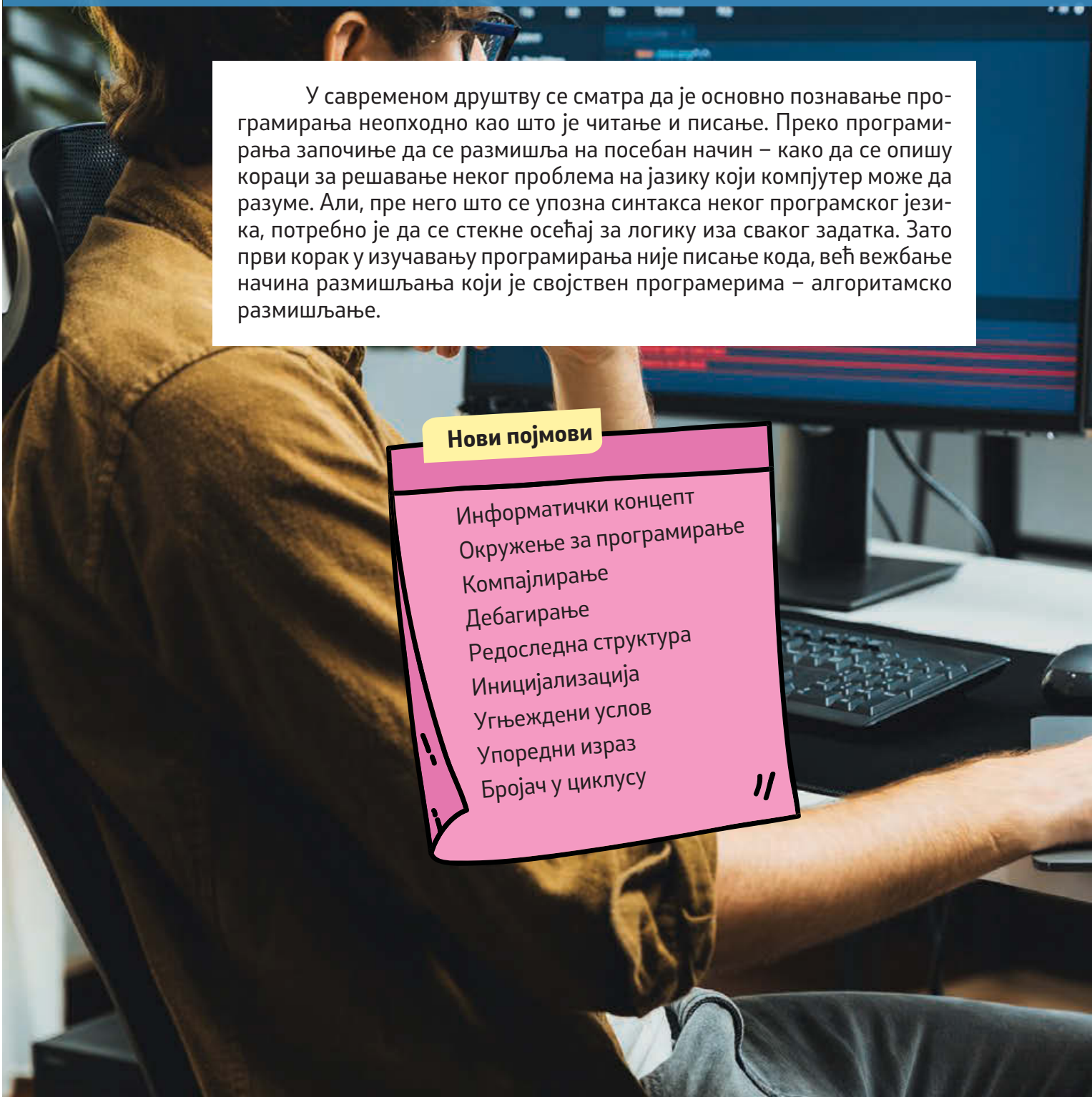
15. Процени да ли је безбедно да се прихвати пријатељски позив од непознате особе на некој социјалној мрежи!

16. Креирај кратку интерактивну презентацију или инфографик о томе како безбедно да се користи интернет са минимум три правила!

17. Направи мини-водич за ученике твог узраста. Нека садржи дефиницију интернета, два безбедносна савета и један пример доброг и један пример лошег дигиталног отиска!

18. Креирај дигитални садржај (постер, кратак текст, слика или мини-видео) ко- ристећи информације које си пронашао на валидним веб-изворима!

ТЕМА 3: Програмирање у C++



У савременом друштву се сматра да је основно познавање програмирања неопходно као што је читање и писање. Преко програмирања започиње да се размишља на посебан начин – како да се опишу кораци за решавање неког проблема на језику који компјутер може да разуме. Али, пре него што се упозна синтакса неког програмског језика, потребно је да се стекне осећај за логику иза сваког задатка. Зато први корак у изучавању програмирања није писање кода, већ вежбање начина размишљања који је својствен програмерима – алгоритамско размишљање.

Нови појмови

- Информатички концепт
- Окружење за програмирање
- Компајлирање
- Дебагирање
- Редоследна структура
- Иницијализација
- Угњеждени услов
- Упоредни израз
- Бројач у циклусу

3.1 Логички и алгоритамски задаци

Већ знаш да...

- препознаш једноставне логичке ситуације из свакодневног живота,
- решаваш проблеме користећи здрав разум и логичко размишљање,
- пратиш упутства корак по корак ,
- приметиш обрасце и правила у игри и задатке, правиш разлику из
- међу тачне и нетачне тврдње.

ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Шта је то Scratch?

2. Шта је такмичарски тип задатака?

3. Шта су променљиве?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- разликујеш логичке задатке из математичког прорачуна,
- анализираш начин решавања алгоритамског проблема,
- пратиш ток логичког размишљања корак по корак,
- користиш основне појмове: услов, наредба, резултат,
- идентификујеш кораке једноставног алгоритма,
- разумеш зашто је важно алгоритамско размишљање.

Програмирање као нови језик размишљања

Програмирање представља дисциплину која повезује математику, логику и решавање проблема. Ради се о процесу који почиње анализом проблема, наставља се планирањем решења преко корака и завршава се реализацијом. У овој првој лекцији неће се користити ни један ред кода. Улазиће се у суштину логичког и алгоритамског размишљања преко занимљивих и једноставних примера. Они ће помоћи да се разуме како се поставља неки проблем, како се анализира и како се приступа његовом решавању.

Логичким задатком означава се свака ситуација где неки проблем може да се реши само уколико се размишља логички, корак по корак, без потребе за сложеним прорачунима. Најпре се поставља циљ који треба да се постигне? Затим се сагледавају све информације које су дате. Анализира се њихова повезаност и идентификује се који од њих су кључни. Коначно, планирају се могућа решења.

ПРИМЕР 1

Место предмета

На столу се налазе три кутије: црвена, плава и жута. У једној од њих налази се предмет. На свакој кутији написана је по једна тврдња:

- **Црвена:** „Предмет је у овој кутији.“
- **Плава:** „Предмет није у црвеној кутији.“
- **Жута:** „Предмет није у овој кутији“

Само једна тврдња је тачна. У којој кутији је предмет?

Анализа почиње претпоставком. Ако је тачно тврдња на црвеној кутији, онда је предмет тамо, али и тврдља на жутој (да није у њој) би била тачна, што значи две тврдње тачне – што није дозвољено. Ако је тачна тврдња на плавој – да није у црвеној, онда је предмет у жутој, али и жута би имала тачну тврдњу. Једина опција где је само једна тврдња тачна, је ако је предмет у **плавој** кутији. Ово је пример где се користи логика без прорачуна.

Друга важна способност која се развија при решавању логичких задатака је препознавање образаца (шаблона) и повезивање услова. То значи да се при датом броју података, тражи њихов међусобни однос и шта они садрже. На овај начин се развија способност да се планирају кораци који воде до циља.

ПРИМЕР 2

Кретање стазом

Робот се креће табелом од 5 редова и 5 колона. Почиње из горњег левог угла. Са сваким потезом може да иде само навише или десно. Колико различитих путева постоји до доњег десног угла?

Ово је класичан задатак типа „бројање путева“, где је сваки потез ограничен одређеним правилима. Иако се још увек не израчунава ништа, могу да се замисле све могуће комбинације кретања: колико пута се иде десно и колико пута наниже. Ово развија осећај за структурисање решења и за бројност могућих опција – основа за алгоритамско размишљање.

Следећи важан аспекат логичких задатака је идентификација услова и последица. У многим задацима постоји збир правила или ограничења која морају да се поштују, и на основу тих ограничења закључује се шта може или шта не може да се догоди. То се назива анализа услова.

ПРИМЕР 3

Загонетка са искључивањем

Пет ученика седи у реду. Зна се да:

- Петар не седи поред Марине.
- Сашко седи десно од Ане.
- Ана није поред Петра. Ко где седи?

Преко логичке елиминације и извођења може да се нађе тачан распоред. Овакви задаци траже пажљиво читање и постепено искључивање немогућих опција. Не користе се бројеви, већ везе између учесника и ограничења. Ово је одличан тренинг за развој систематичности и логичке строгости.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Алгоритми се често користе у свакодневном животу: праћење упутстава, навигација, чак и решавање судоку. Такмичење „Дабар“ је међународни догађај где ученици решавају логичке задатке сличне овима. Ако су ти се допали ови примери, можеш да пробаш да решиш још задатака са официјалне странице такмичења или да креираш своје логичке задатке и да их решаваш са пријатељима.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта значи логичко размишљање у контексту алгоритамских задатака?

2. Зашто су важни кораци при решавању задатака?

3. Кад се каже да је нека тврдња логички немогућа?

4. Зашто задаци без бројева могу да буду корисни при учењу програмирања?

5. Пронађи логичну лажну реченицу ако су две од следећих тачне:

- Плави живи лево од Жутог.
- Зелени живи десно од Плавог.
- Жути није поред Плавог.
- 6. Три пријатеља имају по једну књигу: роман, енциклопедију и бајку. Зна се да Јана не чита енциклопедију, а Бобан не чита бајку. Ко шта чита?
- 7. Осмисли свој логички задатак са највише три услова и најмање два могућа решења.

Запамти шта си научио!

- Логички задаци траже пажљиву анализу података и тврдњи.
- Алгоритамско размишљање започиње јасним дефинисаним циљевима и корацима.
- Нису потребни бројеви или код да би се вежбало решавање проблема.
- Сваки проблем може да се реши систематски, идентификацијом могућности и искључивањем немогућих опција.

3.2 Информатички концепти

Већ знаш да...

- размислиш логички о редоследу догађаја,
- препознаш значење редоследа и правила у свакодневним задацима,
- употребљаваш стратегије за доношење одлука,
- се организујеш при решавању проблема,
- препознајеш када нешто може да се уради ефикасније.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Шта представља анализу услова?

2. Замисли стазу у лавиринту са ограниченим смером (само десно или доле). Колико корака је потребно до циља?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта представља структуре података,
- препознајеш стратегије за оптимизацију задатака,
- разликујеш приступе као што је „завади па владај“ и „похлепно оптимизирање“,
- разумеш зашто је важно распоређивање ресурса,
- примењујеш концепте у свакодневним проблемима.

У свету информатике концепти су темељи на којима се гради способност за решавање сложених задатака. Без основног разумевања начина на које се подаци организују, управљају и оптимизирају, не може да се развије солидна основа за програмирање. Сваки проблем са којим се суочава компјутер, а и човек, има структуру, ограничења и циљ који треба да се постигне. Зато је потребно да се изуче технике које помажу у рационалном доношењу одлука и ефикасном искоришћавању ресурса.

Поред тога, информатички концепти омогућавају да решења буду не само тачна, већ и оптимална. Начини организовања података, разумевање модела решавања, као и избор праве стратегије за конкретни проблем, су део вештина које ће се учити

преко ове лекције. То су вештине које се користе и у реалном свету, када треба да се изабере најбољи пут, да се распореде задаци или да се организује време.

Структуре података

Структуре података представљају начин на који се информације организују и чувају, да би лакше могле да се користе. На пример, списак ученика у дневнику или табела са распоредом часова су реални примери структура које служе за чување података. У информатици, структуре података могу бити редови, скупови, листе, стабла или графови, у зависности од начина на који се подаци организују и користе. Свака од ових структура има своје предности и примењује се када је потребан одређени начин организовања података, приступа информацијама или ефикасности обраде.

ПРИМЕР 1

Замисли библиотеку у којој су све књиге распоређене насумично. Проналажење потребне књиге било би споро и тешко. Међутим, ако су књиге организоване по жанру, аутору или наслову, претрага постаје много бржа и једноставнија. Исто важи и за рачунаре – када су подаци организовани у структуре података, њихово претраживање, чување и обрада су бржи и ефикаснији.

Распоређивање ресурса

Распоређивање ресурса је поступак додељивања ограничених ресурса, као што су време, меморија, простор за складиштење или процесорско време, различитим задацима или процесима. У информатици то може да значи како ће процесор поделити време извршавања између више програма или како ће се радна меморија распоредити за различите задатке.

ПРИМЕР 2

Замисли да треба да распоредиш време за учење пет школских предмета током једне недеље. Ако превише времена посветиш једном предмету, за остале ће остати мање времена. Добар план подразумева равномерно распоређивање времена у складу са значајем или тежином сваког предмета. Сличан принцип примењују и рачунари приликом распоређивања ресурса и управљања задацима.

Похлепни алгоритми

Похлепни приступ (енгл. greedy approach) је стратегија у којој се у сваком кораку бира решење које у том тренутку делује као најбоље, без разматрања свих могућих будућих последица. Иако овакав приступ не доводи увек до најбољег могућег решења, често омогућава да се до прихватљивог решења дође брже и уз мање утрошених ресурса. Због тога се користи у ситуацијама када је важније брзо пронаћи добро решење него трошити време на проналажење савршеног.

ПРИМЕР 3

Замисли да имаш 100 денара и желиш да купиш што више грицкалица. Ако сваки пут изабереш најјефтинији производ који видиш, можда ћеш купити већи број грицкалица, али не нужно и најквалитетније. То је пример похлепног приступа – у сваком кораку бира се оно што у том тренутку изгледа као најбоље решење, без разматрања свих могућих избора. На сличан начин раде и неки похлепни алгоритми, који у сваком кораку бирају локално најбоље решење.

Завади па владај

„Завади па владај" (divide and conquer) је стратегија решавања проблема при којој се велики проблем дели на мање и једноставније потпроблеме. Сваки потпроблем се решава посебно, а затим се добијена решења спајају у једно коначно решење. Овај приступ се често користи у информатици јер смањује сложеност проблема и омогућава ефикасније решавање. Уместо да се решава цео проблем одједном, он се разлаже на мање делове који се лакше обрађују.

ПРИМЕР 4

Замисли да припремаш презентацију. Не пишеш цео текст одједном, већ најпре направиш наслов, затим припремиш садржај по целинама, додаш слике, а на крају све спојиш у једну завршну презентацију. То је пример приступа „подели па владај", који се примењује и у решавању проблема помоћу компјутера.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља структуре података?

2. Зашто је важно распоређивање ресурса?

3. Да ли увек похлепни приступ даје најбоље решење?

4. Наведи пример из свакодневног живота који одговара структури података.

5. Осмисли план за учење где се користи распоређивање времена.

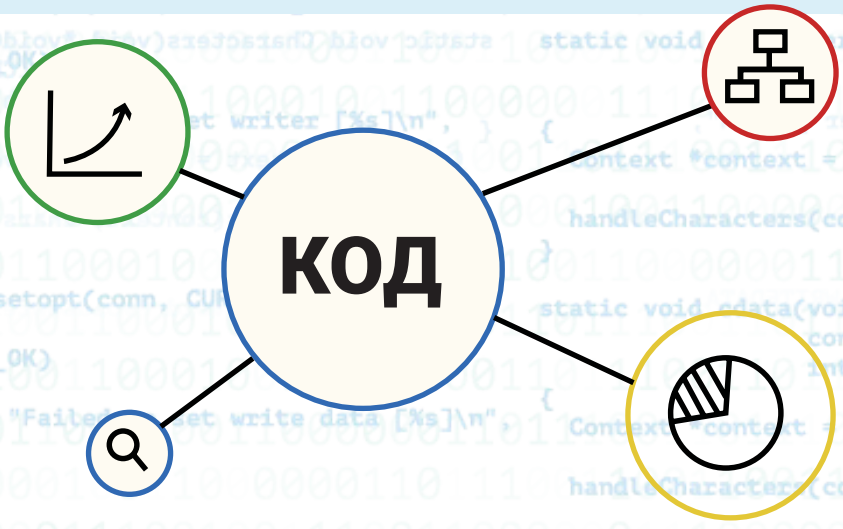
6. Опиши како би организовао/ла библиотеку користећи неку структуру.

Запамти шта си научио!

- Структуре података представљају основу за ефикасно организовање, чување и обраду информација.
- Распоређивање ресурса омогућава ефикасније коришћење времена, меморије и других рачунарских ресурса.
- Похлепни приступ бира локално најбоље решење у сваком кораку, али не доводи увек до глобално најбољег решења.
- Стратегија „Подели па владај“ омогућава ефикасније решавање сложених проблема тако што их дели на мање и једноставније потпроблеме.
- Информатички концепти примењују се не само у програмирању, већ и у многим ситуацијама из свакодневног живота.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Исти концепти који се користе у програмирању, као што су оптимизација, распоређивање ресурса и моделовање, примењују се и у економији, менаџменту, математици и другим областима. На пример, користе се за оптимизацију саобраћајних рута, планирање радних смена или распоређивање ресурса. Структуре података, као што су графови, користе се у друштвеним мрежама за анализу повезаности између људи. Похлепни алгоритми примењују се у многим мобилним апликацијама и навигационим системима за брзо доношење одлука, на пример при избору најкраће или најбрже путање. Истражи како се ови концепти примењују у стварним информационим системима и размисли у којим ситуацијама из свакодневног живота би и ти могао/могла да их примениш.



3.3 Бинарни бројеви: представљање и примена

Већ знаш да...

- разликујеш бројеве и начине њиховог представљања;
- препознајеш значај тачности при раду са бројевима;
- користиш различите мерне јединице и израчунаваш основне аритметичке операције;
- објасниш где и како се бројеви користе у технологијама које нас окружују.

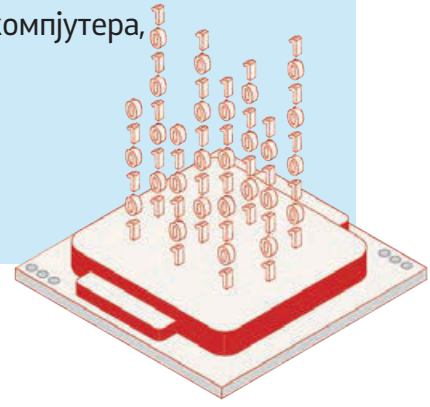
ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Што значи стратегија „завади па владај“?

2. Напиши сценарио у коме дељење задатка води ка лакшем решавању!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објашњаваш шта је бинарни систем и како ради,
- представљаш целе бројеве у бинарном запису,
- разликујеш бинарни и децимални систем,
- повежеш бинарне бројеве са основним функционисањем компјутера,
- представиш информације као битове и бајте,
- примењујеш бинарне бројеве у практичним ситуацијама.



У дигиталној ери, такорећи свака технологија је подржана процесима који користе бројеве. Бројеви нису само део математике, већ су срж комуникације између човека и машине. Сваки компјутер, телефон, дигитални часовник или микропроцесор комуницира и обрађује информације користећи бројеве. Али, ови бројеви нису исти као ини који се свакодневно користе. Компјутери користе посебан систем који омогућава прецизну, брзу и поверљиву обраду: **бинарни систем.**

Све што је на екрану – слике, звуци, текстови – у основи се трансформише у серију бинарних вредности. Ово значи да свака информација коју уноси или добија неки компјутер, на крају се своди само на вредности састављене од две могућности – 0 и 1. Другим речима, компјутери не „размишљају" децималним бројевима (0 – 9), већ бинарним, који су основа дигиталне логике.

Представљање бројева у компјутерској науци се базира на **бинарном систему,** који користи само две цифре: 0 и 1. Овај систем је изабран зато што је много једноставније и ефикасније да електронски уређаји препознају два стања – присуство или одсуство струје, односно високу или ниску вредност напона. Овом једноставношћу се омогућава грађење сложених уређаја који процесуирају огромне количине информација брзо и прецизно.

ПРИМЕР

Да се представи број 5 у бинарном систему.

Користи се метода дељења са 2:

5 / 2 = 2 остатак 1

2 / 2 = 1 остатак 0

1 / 2 = 0 остатак 1

Чита се одоздо навише: 101 Значи,

5 у бинарном систему је 101.

Представљање информација помоћу нула и јединица назива се битовима. Један бит је најмања количина податка и може да буде или 0 или 1. Осам битова формирају бајт, што је најчешће коришћена мерна количина у складирању и трансферу података.

У дигиталним уређајима као што су фотоапарати користи се бинарни систем како би се одређеном вредношћу кодификовао сваки пиксел. Колико више битова се користи за сваки пиксел, толико је већа прецизност боја и детаља. У сваком уређају, па и у најмањим компонентама, бинарни код има главну улогу у процесирању и меморисању података.

ПРИМЕР

Како се кодира слово **А** у бинарни код?

У **ASCII стандарду**, слово **А** има број 65. Представљено у бинарном систему, то је:

65 / 2 = 32 остатак 1
32 / 2 = 16 остатак 0
16 / 2 = 8 остатак 0
8 / 2 = 4 остатак 0
4 / 2 = 2 остатак 0
2 / 2 = 1 остатак 0
1 / 2 = 0 остатак 1

Чита се одоздо навише: **1000001**

Ово значи да се слово А чува као **1000001** у меморији компјутера.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта је бинарни број?

2. Зашто компјутери користе бинарни систем?

3. Претвори број 9 у бинарни систем.

4. Шта представља бајт и колико битова садржи?

5. Претвори бројеве 10, 12 и 15 у бинарни систем.

6. Са колико битова може да се представи тачно 256 различитих вредности?

Запамти шта си научио:

- Бинарни систем користи само две цифре: 0 и 1.
- Компјутери раде са бинарним бројевима због електронске природе њихових компонената.
- Сваки број, слово или податак у компјутеру представљен је у бинарном облику.
- Бит и бајт су основне јединице за мерење података.
- Претварање децималног у бинарни систем врши се преко дељења са 2.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Испитај како изгледа бинарна аритметика и како се врши сабирање бинарних бројева.
- Истражи како компјутер користи двоичан систем за представљање слика и звукова.
- Прочитај о IEEE форматима представљање децималних бројева (floating point).
- Истражи како се представљају негативни бројеви у компјутеру (на пример, са „допуњавањем до два" – two's complement).

3.4 Основи кодирања и енкрипција

Већ знаш да...

- препознајеш бинарне бројеве и повезујеш их са дигиталним представљањима,
- разликујеш логичке и алгоритамске кораке,
- схватио си шта је информација и како се представља симболима,
- пратиш основне кораке у анализи проблема.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Колико вредности може да се представи са 6 битова?

2. Напиши како би се представио број 100 у бинарном облику!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта представља кодирање и где се примењује,
- разликујеш кодирање од енкрипције,
- препознајеш примере кодирања у свакодневном животу,
- разумеш зашто се користе шифре и како помажу у заштити информације,
- разликујеш јавне и тајне поруке,
- размишљаш како би се на различите начине представили подаци.

У свету дигиталне комуникације постоји стална потреба да се информација представља, преноси и чува на начин који ће бити сигуран, тачан и разумљив. Због тога, од најранијих дана људске цивилизације развијане су разне методе трансформисања порука из једног облика у други – процес познат као кодирање. Путем кодирања могла су се преносити значења чак и без употребе писаног језика, а енкрипција, као посебна грана, је омогућила да те поруке остану тајне. Да би се програмирање разумело као и дигитална обрада података, потребно је најпре да се савладају ови основни концепти, јер су управо они основа на којој се гради безбедност и размена информација у савременом свету.

Разлика између кодирања и енкрипције често се губи када се користе у свакодневном говору, али у информатици имају веома прецизна значења. Кодирање подразумева претварање информације из једног облика у други због лакшег читања, обраде или чувања. На пример, коришћење бинарног кода за представљање слова и симбола у компјутеру је облик кодирања. Са друге стране, енкрипција представља методу којом се сакрива прави садржај поруке, тако да само овлашћена лица могу да је дешифрирају. Разумевање ових појмова представља неопходну основу за будуће учење о безбедности података, дигиталних сертификата и компјутерских мрежа.

Кодирање

У историји човечанства кодирање је било коришћено као начин преноса порука међу људима, посебно у ситуацијама када су информације требале да остану разумљиве само одређеним примачима. Један од најпознатијих историјских примера оваквог облика кодирања је Морзеова азбука, која се користила у телеграфској комуникацији. У овом систему, свако слово азбуке је представљано комбинацијом кратких и дугих сигнала (тачке и цртице). Иако данас изгледа примитивно, Морзеов код се сматрао револуционарним начином кодирања, јер је омогућавао комуникацију преко једноставних електричних импулса. У информатичком смислу, овај начин кодирања представља најаву дигиталне представе информације, где се комбинације симбола (или сигнали) користе за тачно представљање значења.

ПРИМЕР 1

Морзеов код

Слово „А“ у Морзеовом коду се представља као „.-“, што значи једна кратка тачка праћена дугом цртицом. На исти начин, „Б“ се представља као „-...“. Цела енглеска азбука, као и бројеви и неки специјални знаци, имају свој код у овој азбуци. Примена Морзеове азбуке се састојала од слања електричних сигнала или светлосних импулса који су били разумљиви примаоцима од примача који су познавали шифру. Тиме се омогућавао пренос информација чак и у отежаним условима када гласовна комуникација није била могућа.

У информатичким системима кодирање се не примењује само за прикривање података, већ и за њихово представљање на начин који машине могу да обраде. Најосновнија и најважнија врста кодирања у компјутерском свету је **бинарно кодирање**. Цео дигитални свет се базира на представљању информација са само два стања – 0 и 1. Ове бинарне вредности се користе за представљање сваког типа података: бројеви, текст, слике, звуци и видео. Сваки знак из текста, на пример, представља се преко стандардног кода као што је ASCII (American Standard Code for Information Interchange), где свако слово, број или специјални знак има своју уникатну бинарну вредност.

ПРИМЕР 2

ASCII кодирање

Слово „А“ према ASCII стандарду има вредност 65. У бинарном запису то је 01000001. Слично, слово „а“ има вредност 97, што у бинарном облику изгледа као 01100001. Уз помоћ оваквих кодова, текст може да се претвори у низове од бинарних вредности које компјутер лако чува и обрађује. Ово показује да кодирање не мора увек да има тајну функцију, већ је кључно за комуникацију између човека и машине.

Енкрипција

Енкрипција, као процес скривања значења поруке, одувек је била важан део безбедне комуникације. Један од првих и најпознатијих примера јесте Цезарова шифра, названа по Јулију Цезару, који ју је користио за шифровање војних порука. Принцип ове енкрипције заснива се на једноставном померању слова у поруци за одређени број места у азбуци. Тако се свако слово изворне поруке замењује другим словом које се налази неколико места даље у азбуци. Иако се данас ова техника сматра слабом и лаком за разбијање, она представља одличан увод у концепт симетричне енкрипције, у којој се исти кључ користи за шифровање и дешифровање.

ПРИМЕР 3

Цезарова шифра

Ако се у Цезаровој шифри користи померање за 3 позиције у српској ћириличној азбуци, слово „А“ постаје „Г“, а „Б“ постаје „Д“. Реч „ИНФО“ претвара се у „ЛПЧС“. Да би дешифровао поруку, прималац треба да зна да свако слово треба померити три позиције уназад. Овај једноставан метод показује како се информација може прикрити, али и колико је важно да пренос кључа буде безбедан – ако неко открије кључ, може лако да дешифрује поруку.

Савремено дигитално друштво се ослања на напредне методе енкрипције како би се гарантовала безбедност комуникација, трансакција и чувања података. За разлику од једноставних техника из прошлости, као што је Цезарова шифра, данашње методе користе сложене математичке алгоритме и криптографске протоколе. Два од најчешће коришћених модела су симетрична и асиметрична енкрипција. Код симетричне енкрипције исти кључ се користи за шифровање и дешифровање података, што значи да обе стране у комуникацији морају да поседују исти кључ и да га чувају у тајности. Са друге стране, код асиметричне енкрипције као што је RSA алгоритам користе се два кључа – јавни и тајни – што омогућава безбедну комуникацију и аутентикацију без претходне размене тајни.

ПРИМЕР 4

Асиметрична енкрипција помоћу RSA

Када једно лице жели да пошаље поруку другом лицу преко интернета, може да је шифрује јавним кључем примаоца. Након што порука стигне, прималац је дешифрује својим приватним (тајним) кључем. На овај начин, чак и ако неко пресретне поруку, неће моћи да је прочита без приватног кључа. RSA и слични алгоритми представљају основу данашње безбедне комуникације преко HTTPS-а, онлајн банкарства и дигиталних потписа. Криптографија представља науку о заштити информација њиховим претварањем у облик који се не може разумети без одговарајућег кључа. Користила се још од античких времена, али је у дигиталној ери постала незаменљив део свакодневног функционисања интернета, банка ских система, здравствених служби и државних институција. За разлику од обичног кодирања, циљ криптографије није само да претвори податке већ и да их заштити од неовлашћеног приступа. Криптографски системи заснивају се на математичким принципима и алгоритмима и омогућавају аутентификацију, поверљивост, интегритет и непроменљивост података. Сваки пут када се неко лице пријави на веб-страницу, изврши онлајн трансакцију или пошаље електронску пошту, криптографија обезбеђује да информације буду заштићене. Савремена криптографија заснива се на принципу претварања читљивих података (тзв. „отвореног текста“) у шифроване податке („шифрат“), коришћењем криптографских алгоритама и кључева. При томе податак постаје бесмислен свакоме ко га пресретне, осим ако не поседује одговарајући кључ за дешифровање. Овај процес може бити симетричан, при чему се исти кључ користи и за шифровање и за дешифровање, или асиметричан, при чему се користе два различита кључа – јавни и приватни. Симетрични системи су бржи, али захтевају безбедан начин размене кључева, док асиметрични системи пружају већу флексибилност и користе се у ситуацијама у којима безбедан пренос кључа није могућ. Криптографија се примењује код електронске поште, банкарских трансакција, дигиталних потписа, сертификата и шифровања података сачуваних на рачунару или мобилном уређају. Без ових технологија поверење у дигитални свет било би готово немогуће.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља појам „кодирање“ и по чему се разликује од енкрипције?
2. Објасни разлику између симетричне и асиметричне енкрипције!
3. Који су основни елементи Цезарове шифре и како она функционише?
4. Какву улогу имају јавни и приватни кључ у RSA алгоритму?
5. Претвори следећу поруку у Морзеову азбуку: „HELLO“!
6. Шифруј реч „DATA“ користећи Цезарову шифру са кључем 3!
7. Објасни како би се RSA алгоритам користио приликом слања електронске поште!
8. Замисли да имаш тајни кључ за симетричну енкрипцију! Опиши како би га поделио/поделила са сарадником, а да га нико други не пресретне!

Запамти шта си научио!

- Кодирање је процес претварања информације из једног облика у други ради лакше обраде, складиштења или преноса.
- Енкрипција се користи за заштиту података тако што се информација шифрује и без одговарајућег кључа не може да се прочита.
- Пример кодирања је Морзеова азбука, у којој су слова представљена тачкама и цртама.
- Цезарова шифра један је од најстаријих облика енкрипције и заснива се на померању слова у азбуци.
- RSA је асиметрични криптографски алгоритам који користи јавни и приватни кључ за безбедну комуникацију.
- У симетричној енкрипцији исти кључ користи се и за шифровање и за дешифровање.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- У савременој криптографији, поред RSA алгоритма, користе се и други алгоритми, као што су AES (Advanced Encryption Standard) и ECC (Elliptic Curve Cryptography).
- Дигитални сертификати и SSL протокол користе енкрипцију како би обезбедили безбедност интернет комуникације.
- Хеш-функције (на пример, SHA-256) користе се за проверу интегритета података.
- Квантна криптографија је напредна област која користи квантне појаве за стварање непробојних шифара.
- Принципи енкрипције и кодирања представљају основу сајбер-безбедности и без њих би савремена дигитална комуникација била лака мета злоупотребе.

3.5 Шта је програмирање?

Већ знаш да...

- објасниш шта је компјутер и које су његове главне компоненте,
- разликујеш хардвер и софтвер,
- користиш дигиталне уређаје за свакодневне задатке.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Зашто се Морзеова азбука сматра обликом кодирања, али не и енкрипције?

2. Напиши упутство за шифровање и дешифровање поруке коришћењем симетричног кључа!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта представља програмирање,
- опишеш и објасниш поступак за одређено решавање проблема.

У свакодневном животу рачунари се користе за велики број задатака: израчунавање, обраду текста, цртање, комуникацију и још много тога. Међутим, рачунар сам по себи не зна како да извршава ове задатке. Он мора да добије јасна и прецизна упутства за сваки корак који треба да обави. Управо то представља суштину програмирања. Програмирање је процес стварања низа инструкција написаних на посебном језику који рачунар разуме – програмском језику. Ове инструкције групишу се у такозвани програм, који упућује рачунар шта тачно треба да уради, којим редоследом и под којим условима. Циљ ове лекције је да се ученици упознају са основном суштином програмирања, његовом применом и значајем ове вештине у данашњем дигиталном свету.

Програмирање је активност у којој људи – програмери – стварају упутства за рачунар. Ова упутства, односно инструкције, називају се кодом и пишу се на програмском језику. Програмски језик је средство помоћу којег човек може да комуницира са машином. Сваки језик има своја правила, граматику и структуру, слично природним језицима. На пример, као што реченица „Укључи светло“ има јасно значење за човека, тако и инструкција turnOn(light) може имати значење за рачунар ако је напи-

сана програмском језику који он разуме. Циљ је да опишемо шта желимо да рачунар уради – корак по корак.

ПРИМЕР 1

Замисли да треба да даш упутства роботу да обави кућни посао. Ако желиш да опере судове, упутства би гласила: „иди до судопере“, „пусти воду“, „стави детерџент“, „опери суд“, „испери га“, „стави га да се суши“. Слично томе, програмер задаје рачунару јасне кораке – сваки корак мора бити прецизан и недвосмислен.

Када се програм једном напише, рачунар може увек на исти начин и без грешке да извршава исти задатак, све док је код исправан. То програм чини моћним средством за аутоматизацију – активности које би одузимале много времена и биле подложне људским грешкама претварају се у задатке које машина обавља брзо и тачно. Програми се не користе само за велике и сложене пројекте – они се налазе и у уређајима које свакодневно користимо: калкулатору, мобилном телефону, банкомату, фрижидеру или паметном сату. Сви они раде према неком програму који је написао човек.

ПРИМЕР 2

Када се на калкулатору унесе „5 + 3 =“, у позадини се извршава програм који чита бројеве, препознаје операцију „+“, израчунава збир и приказује резултат. Сви ови кораци унапред су описани у програмском коду.

Важно је разумети да програмер не пише само код – он логички размишља, дели задатке на мање делове, проверава да ли су све могуће ситуације узете у обзир и тестира код како би се уверио да исправно ради. Када се програм не понаша на очекивани начин, обавља се такозвано „дебаговање“ – проналажење и исправљање грешака. Програмирање је спој логичког размишљања, креативности и дисциплине. Није неопходно велико знање математике, већ способност решавања проблема.

ПРИМЕР 3

Да би се направила апликација која израчунава колико је времена потребно за путовање, потребно је унети растојање и брзину, а програм треба да израчуна време према формули: време = растојање / брзина. Међутим, програмер мора да предвиди шта ће се догодити ако неко унесе брзину = 0. То би довело до грешке, па ће добар програмер увести додатну проверу: „Ако је брзина = 0, прикажи поруку: Унеси исправну вредност.“

ЗАКЉУЧАК

Програмирање није само писање кода – оно је уметност прецизног и логичног изражавања корака које рачунар треба да изврши. Помоћу програмирања рачунари се користе за аутоматизацију, убрзавање и поједностављивање процеса у свакодневном животу. Учити програмирање значи стицати способност решавања проблема, стварања нечег новог и разумевања начина на који функционишу дигитални алати које свакодневно користимо.

ПИТАЊА ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља програмски језик?

2. Која је разлика између хардвера и програма?

3. Да ли програмирање увек захтева знање математике?

4. Које су основне активности програмера?

5. Шта је дебаговање и зашто је важно?

Запамти шта си научио!

- Шта представља програмски језик?
- Која је разлика између хардвера и програма?
- Да ли програмирање увек захтева знање математике?
- Које су основне активности програмера?
- Шта је дебаговање и зашто је важно?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи који су програмски језици данас најпопуларнији!
- Покушај да решиш логичку загонетку у којој сваки корак мора бити јасан и прецизан!
- Посети неку бесплатну онлајн платформу за учење програмирања, као што су Scratch или Code.org!

3.6 Програмски језик и преводилац

Већ знаш да...

- разликујеш хардвер и софтвер,
- објасниш шта представља програмирање,
- опишеш логички поступак преко секвенце из инструкција.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Објасни својим речима шта је програмирање!

2. Опиши један задатак из свакодневног живота који се може аутоматизовати помоћу програма!

3. Наведи неколико уређаја који користе програме!

4. Шта програмер ради када се појави грешка у програму?

5. Зашто је важно да свака инструкција буде јасна и прецизна?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта је преводилац,
- објасниш шта је компајлер,
- направиш разлику између разних начина рада преводиоца и компајлера.

Када људи желе да комуницирају са рачунарима, користе посебне језике које машина може да разуме. Ови језици, названи програмски језици, представљају формалан и логички организован начин изражавања који омогућава писање инструкција које рачунар може да изврши. За разлику од природних језика које људи користе у свакодневној комуникацији, програмски језици морају бити прецизни, структурирани и недвосмислени. Без таквог језика програмирање не би било могуће, јер рачунар не разуме људски говор, већ само машински код – низ бинарних наредби.

Програмски језик омогућава изражавање логике и корака које треба извршити ради решавања одређеног задатка. Сваки језик има сопствену синтаксу – правила према којима се пише код. Постоје многи програмски језици, као што су C++, Python, Java, JavaScript и други, а сваки од њих има свој стил писања и области примене. Избор језика зависи од задатка који се решава – неки језици су погоднији за мобилне апликације, неки за научна израчунавања, а други за веб-развој. Иако се разликују по изгледу, сви они имају заједнички циљ: да људске идеје претворе у облик који рачунар може да изврши.

ПРИМЕР 1

У језику C++ пише се инструкција cout << «Zdravo»; да би се приказао текст. У језику Python иста намера изражава се инструкцијом print(«Zdravo»). Оба примера постижу исти резултат – приказивање текста – али на различите начине, у складу са синтаксом сваког језика.

Рачунар не разуме програмски језик непосредно. Написани код најпре мора бити „преведен" на машински језик. Ову улогу имају преводиоци, који могу бити две врсте: компајлери и интерпретери. Компајлер преузима целокупан изворни код, анализира га и претвара у извршни програм који се може самостално покренути. Интерпретер, с друге стране, чита код ред по ред и непосредно га извршава, без стварања посебне датотеке (фајл).

ПРИМЕР 2

Када се пише програм у језику C++, компајлер најпре преводи целокупан код у извршну .exe датотеку. Затим се та датотека може покренути на рачунару. У језику Python интерпретер непосредно чита оно што је написано и одмах то извршава. Разлика је слична оној између представе која је снимљена као филм (компајлирана) и представе која се изводи уживо пред публиком (интерпретирана).

Преводилац не само што претвара код у машински језик већ и проверава исправност синтаксе. Ако у коду постоје грешке – недостаје зарез или заграда, или је употребљено неодговарајуће име променљиве – преводилац ће пријавити грешку и неће наставити процес. Ова провера је изузетно важна јер спречава извршавање погрешних или небезбедних инструкција. Тако програмски језик и преводилац заједно омогућавају безбедан и ефикасан процес развоја софтвера.

ПРИМЕР 3

Ако програмер у језику C++ напише cout << «Zdravo» без тачке са зарезом на крају, компајлер ће пријавити грешку: „expected ';' before end of statement". Тако програмер одмах зна где треба да интервенише и исправи грешку пре него што се програм изврши.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи како ради компајлер за C++ (g++) или интерпретер за Python!
- Покушај да напишеш кратак код у језику Python и погледај шта ће се догодити ако направиш синтаксичку грешку!
- Посети страницу https://www.learnpython.org/ и испробај једноставне примере!

ЗАКЉУЧАК

Програмски језици и преводиоци представљају кључне алате у процесу програмирања. Без њих не би било могуће написати, превести и извршити инструкције на рачунару. Они омогућавају прецизну, поуздану и ефикасну комуникацију између човека и машине. Разумевање начина на који функционишу поставља темеље за сва будућа знања повезана са развојем софтвера и дигиталних решења.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Упоредите начин рада компајлера и интерпретера!

2. Зашто је синтакса важна у програмском језику?

3. Да ли се један програмски језик може користити за све задатке?

4. Која је улога преводиоца у процесу извршавања програма?

5. Шта ће се догодити ако у коду постоји грешка?

Запамти шта си научио!

- Програмски језици користе се за писање инструкција.
- Компајлер претвара целокупан програм у машински код.
- Интерпретер извршава програм ред по ред.
- Синтакса је скуп правила која се морају поштовати.
- Преводилац упозорава ако у коду постоје грешке.

3.7 Разлика између програмских језика: Scratch, C++, Java, Python, PHP, Lisp

Већ знаш да...

- објасниш шта је програмски језик,
- правиш разлику измеуу компајлера и интерпретера,
- препознајеш грешке у синтакси.

ПИТАЊА ЗА ПОНАВЉАЊЕ:

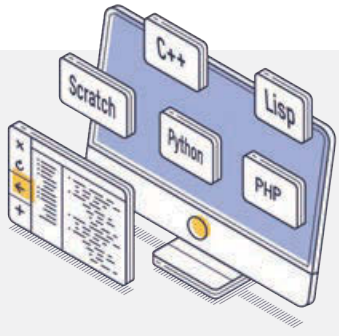
1. Шта представља програмски језик?

2. Које врсте преводиоца постоје?

3. Који је принцип рада компајлера?

4. Који је принцип рада интерпретера?

5. Шта представља синтаксичку грешку?



У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш у чему је разлика између различитих програмских језика,
- разликујеш начин рада различитих програмских језика.

У свету програмирања постоје десетине, па чак и стотине програмских језика, од којих сваки има своје предности, правила, област примене и циљну групу. Сваки језик створен је са одређеним циљем: неки су погодни за образовне сврхе, други за развој системског софтвера, трећи за веб-апликације, а четврти за вештачку интелигенцију. Иако сви служе за писање инструкција које ће рачунар извршити, начин на који се те инструкције пишу, читају и извршавају може знатно да се разликује. Зато је важно да се ученици упознају са неколико различитих језика како би стекли шире разумевање могућности програмирања.

Први језик са којим се почетници најчешће упознају јесте Scratch. Као што је већ познато, реч је о визуелном програмском језику који је развио MIT, у којем се уместо писања кода користе графички блокови који се повезују попут слагалице. Овај приступ елиминише синтаксичке грешке и омогућава ученицима да се усредсреде на логику, редослед операција и услове у програму. Scratch се користи за стварање игара, анимација и једноставних симулација, чиме се омогућава интерактиван начин учења без застрашујућег изгледа „правог“ кода.

ПРИМЕР 1

У програмском језику Scratch, ако желите да се лик помери напред 10 корака, једноставно изаберете блок „помери се 10 корака“ и повежете га са осталим блоковима. Нема потребе за писањем текстуалног кода нити за вођењем рачуна о тачкама са зарезом.

Следећи важан језик је C++, један од најстаријих и најмоћнијих програмских језика који се и данас користи. То је компајлирани језик, што значи да се целокупан програм најпре преводи у машински код, а тек затим извршава. C++ је познат по својој брзини, контроли над меморијом и могућности извођења операција ниског нивоа. Користи се за развој оперативних система, уграђених система, па чак и видео-игара. Ипак, због сложене синтаксе не препоручује се као први језик апсолутним почетницима, али је одличан за напредне ученике. C++ представља основу за развој многих друштвених мрежа, као и криптовалута. Његова брзина и стабилност чине га идеалним програмским језиком за многе врсте софтвера.

ПРИМЕР 2

Да би се приказао текст у језику C++, користи се код cout << «Zdravo»;, али је претходно потребно укључити датотеку заглавља помоћу наредбе #include <iostream> и написати функцију main(). То представља знатно виши почетни праг у односу на Scratch.

Java и Python су два савремена програмска језика са широком применом. Java је објектно оријентисан и веома структуриран језик, а користи се за развој мобилних апликација (посебно за Android), десктоп програма и веб-апликација. Python је, с друге стране, скриптни језик, веома популаран због своје једноставне синтаксе и читљивости. Користи се у машинском учењу, обради података, науци и аутоматизацији. Велики број почетника започиње учење програмирања управо језиком Python због његове једноставности.

ПРИМЕР 3

У језику Python за приказивање текста довољно је написати print(«Zdravo»), док је у језику Java за исти резултат потребно више редова кода и позивање објеката. То чини Python интуитивнијим за први сусрет са програмирањем.

PHP и Lisp представљају специјализоване програмске језике. PHP се најчешће користи за серверско веб-програмирање, посебно за израду динамичких веб-страница. Он се уграђује у HTML код и омогућава интеракцију са базама података и обраду образаца. Lisp је, с друге стране, један од најстаријих програмских језика, познат по примени у вештачкој интелигенцији и функционалном програмирању. Иако се ретко користи међу почетницима, Lisp пружа сасвим другачији начин размишљања јер користи рекурзију и симболичку обраду података.

ПРИМЕР 4

У PHP, ако желите да прикажете текст, пише се: <?php echo «Zdravo!»; ?>, што показује како се програмски језик уграђује у HTML код. У језику Lisp израз (+ 1 2) означава сабирање – оператор се налази на почетку, што је супротно начину писања у већини програмских језика.

ЗАКЉУЧАК

Разумевање разлика међу програмским језицима омогућава ученицима да донесу информисану одлуку о томе који ће језик наставити да уче. Сваки од ових језика има своје предности, недостатке и области примене. Од језика Scratch, који развија визуелну интуицију, преко језика C++, који омогућава контролу над системом, и језика Java и Python, намењених развоју апликација, до језика PHP и Lisp, који се примењују у веб-програмирању и вештачкој интелигенцији – спектар је широк и вредан истраживања.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Упоредите Scratch и Python према начину писања кода!

2. Зашто се PHP често користи у веб-развоју?

3. Које сличности постоје између језика Java и C++?

4. Који је језик погоднији за апликације засноване на вештачкој интелигенцији?

5. Опишите разлику између компајлираног и интерпретираног језика и наведите пример!

Запамти шта си научио!

- Scratch је визуелни програмски језик погодан за почетнике.
- C++ је компајлирани језик, погодан за напредне задатке.
- Python има једноставну синтаксу и широку примену.
- Java је објектно оријентисан и стабилан програмски језик.
- PHP се користи за веб-програмирање, а Lisp за вештачку интелигенцију.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Инсталирај Scratch и покушај да направиш анимацију!
- Напиши кратки програм у Python и упореди са истим задатком у Java!
- Истражи како се користи PHP на веб странама које посећујеш!
- Испробај Lisp у онлајн компајлеру и анализирај његову синтаксу!

3.8 Интегрирана околина за програмирање

Већ знаш да...

- објасниш шта је IDE,
- разликујеш компајлирање од интерпретирања,
- препознајеш грешке у синтакси.

ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Шта је визуелни програмски језик?

2. За шта се најчешће користи C++?

3. Који језик је најпогоднији за почетнике?

4. Које су предности Python-а?

5. Шта је карактеристично за Lisp?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта је IDE,
- препознаеш разноврсно развојно окружење.

У савременом програмирању процес писања, тестирања и извршавања кода не одвија се у празном прозору конзоле нити ручно, преко командне линије. Уместо тога, користе се специјализовани алати који олакшавају целокупан процес развоја. Ови алати познати су под називом интегрисано развојно окружење (IDE). IDE представља комплетан софтверски пакет који садржи све компоненте потребне за ефикасно програмирање: уређивач кода, компајлер, дебагер и конзолу за извршавање. Ово целовито окружење користе и почетници и професионални програмери јер омогућава организован начин рада и визуелну контролу над процесом развоја.

Интегрисано развојно окружење представља „дом“ програмера – место на којем се програми пишу, проверавају и извршавају.

За разлику од једноставног уређивача текста, IDE нуди функције као што су аутоматско довршавање кода, означавање синтаксичких грешака, провера типова и брз приступ алатима за компајлирање. Ове могућности не само да убрзавају развој већ га чине и продуктивнијим. Помоћу IDE-а ученик може одмах да види да ли је неки део кода правилно написан, као и да га тестира притиском на једно дугме.

ПРИМЕР 1

У програму Visual Studio Code, док се пише cout, IDE аутоматски нуди наставак израза, додајући << и предлажући endl. То смањује ризик од грешака и повећава брзину писања кода.

У оквиру IDE-а постоји и могућност непосредног компајлирања и извршавања кода. Нема потребе да ученик ручно отвара командну линију, тражи датотеку и покреће је помоћу посебне наредбе. Једним кликом целокупан програм се преводи и извршава. Поред тога, IDE омогућава истовремено уређивање више датотека, праћење структуре пројекта и управљање зависностима између делова кода.

ПРИМЕР 2

У програму Code::Blocks, након што се програм напише, притисне се дугме „Build and Run“, после чега се резултат одмах приказује у доњем прозору. Нема потребе за засебним тражењем грешака изван развојног окружења.

Још једна важна компонента интегрисаног развојног окружења јесте дебагер – алат који омогућава праћење извршавања кода корак по корак. Помоћу ове функције ученик може да постави такозвану „тачку прекида“ (енгл. breakpoint) – место на којем ће се програм зауставити – и да провери које вредности променљиве имају у том тренутку. Ово је од непроцењивог значаја за разумевање логике програма и откривање такозваних логичких грешака.

ПРИМЕР 3

Ако у неком програму променљива x треба да има вредност 5, али IDE показује да она добија вредност 3, помоћу дебагера може се пронаћи ред у којем настаје грешка и предложити одговарајућа исправка.

Савремена интегрисана развојна окружења нуде и додатне модуле, као што су симулатори хардверских уређаја, уграђени терминали и интеграција са системима за контролу верзија, попут Git-а. У образовном контексту нека интегрисана развојна окружења прилагођена су почетницима и имају поједностављен изглед и ограничен број функција – на пример, Thonny за Python или Arduino IDE за микроконтролере. Ове варијанте пружају почетну равнотежу између функционалности и једноставности, чиме се омогућава лакши прелазак са визуелних алата на писање стварног кода.

ЗАКЉУЧАК

Интегрисано развојно окружење представља централно место за писање и тестирање програма. Без обзира на то да ли је реч о почетничком пројекту или професионалној апликацији, коришћење IDE-а знатно олакшава рад, повећава ефикасност и смањује могућност настанка грешака. Познавање његових компоненти и функција представља први корак ка стварном програмирању.

ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Шта представља IDE?
2. Које су три главне компоненте једне IDE?
3. За шта се користи дебагер?
4. Да ли је могуће да се изврши програм без IDE?
5. Који IDE се користе за Python?

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Опиши разлику између IDE и обичног уређивача текста!
2. Која је улога аутоматског завршавања у IDE?
3. Како помаже дебагер при откривању грешака?
4. Шта су breakpoints и када се користе?
5. Која је најодговарајућа IDE за почетнике и зашто?

Запамти шта си научио!

- IDE представља интегрисану средину за писање, проверу и извршавање кода.
- Укључује уређивач, компајлер и дебагер.
- Помаже у откривању и одстрањивању грешака.
- Омогућава брз и визуелни преглед резултата.
- Идеално за почетнике и професионалце.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Инсталирај Visual Studio Code и напиши програм у језику Python!
- Испробај функцију дебаговања и користи тачке прекида (breakpoints)!
- Истражи како функционише интеграција Git-а у IDE-у!
- Упореди два интегрисана развојна окружења и запиши разлике!

3.9 Писање, извршавање и дебаговање програма

Већ знаш да...

- објасниш шта је интегрисано развојно окружење,
- разликујеш основне програмске језике,
- наводиш примере IDE.

ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Зашто служи IDE?

2. Које окружење за C++ знаш?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта је дебаговање,
- компајлираш.

Процес стварања програма не завршава се писањем кода. Напротив, писање је само први корак у ширем процесу развоја софтвера. Програм треба компајлирати, извршити и проверити да ли се понаша на очекивани начин. Прва верзија кода обично садржи грешке – синтаксичке, логичке или грешке током извршавања. Зато важну фазу у програмирању представља дебаговање – процес проналажења и отклањања грешака у коду. Разумевање целокупног овог процеса важно је за свакога ко учи да програмира јер га води од теорије ка практичној примени.

Писање програма започиње употребом **уређивача текста**, који је део IDE-а. У њега се уносе инструкције на изабраном програмском језику, при чему се води рачуна о синтакси, распореду елемената и исправности логике. Код треба да буде читљив, уредно распоређен и раздвојен празним редовима, са коментарима који објашњавају делове логике. Правилним писањем смањује се вероватноћа настанка грешака и олакшава каснија анализа.

ПРИМЕР 1

Приликом писања програма у језику C++, ако заборавимо да ставимо ; на крају реда, током компајлирања добићемо синтаксичку грешку. IDE, као што је Code::Blocks, аутоматски ће означити погрешан ред и приказати поруку: „expected ';' before 'return'".

Након писања, код мора да се **компајлира** – то значи да се преведе из изворног кода у код који машина може да разуме. Током овог процеса проверава се да ли постоје синтаксичке грешке, као што су погрешно написане кључне речи, изостављене заграде или неправилни изрази. Ако постоје грешке, компајлер приказује поруке које указују на ред и врсту грешке. Ученици уче да читају и тумаче ове поруке, што представља део свакодневног рада сваког програмера.

ПРИМЕР 2

Ако се напише наредба којом се на место предвиђено за цео број смешта текст, компајлер ће пријавити грешку јер се текст не може доделити целобројној променљивој. IDE ће приказати поруку: „invalid conversion from 'const char*' to 'int'".

Након успешног компајлирања следи **извршавање програма**. У овој фази програм се покреће и приказује резултат на екрану. Ако се не понаша на очекивани начин, могуће је да постоји логичка грешка – програм се извршава без проблема, али не даје тачан резултат. Ове грешке најтеже је открити јер их компајлер не препознаје.

ПРИМЕР 3

Ако програм треба да израчуна производ, али се у коду употреби оператор + уместо *, резултат ће бити погрешан, без икаквог упозорења. Програм ће радити, али неће извршавати оно што се од њега захтева.

Дебаговање је процес којим се откривају и анализирају овакве логичке или скривене грешке. Користе се алати као што су тачке прекида (breakpoints) – места у коду на којима се извршавање привремено зауставља – и функција извршавања корак по корак (step-by-step), помоћу које се прате промене вредности променљивих. IDE окружења омогућавају визуелни приказ вредности и једноставно кретање кроз редове кода ради провере. Ова техника помаже ученицима да разумеју шта се тачно дешава у њиховом коду.

ЗАКЉУЧАК

Писање програма не завршава се кодирањем. Сваки програм мора бити пажљиво компајлиран, тестиран и, према потреби, дебагован. Грешке не треба доживљавати као неуспех, већ као природан део процеса. Са сваком исправљеном грешком научи се нешто ново. Овладавање дебаговањем значи стицање једне од најважнијих вештина у програмирању.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Наведи пример синтаксичке грешке!

2. Шта представља поруку за грешку од компајлера?

3. Опиши како се врши дебаговање корак по корак!

4. Који IDE алати се користе за дебаговање?

5. Зашто се неке грешке не откривају при компајлирању?

Запамти шта си научио!

- Писање кода захтева пажњу усмерену на синтаксу и логику.
- Грешке су уобичајен део процеса.
- Компајлер проверава синтаксу.
- Извршавање програма показује да ли је резултат тачан.
- Дебаговање је процес откривања и исправљања грешака.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Напиши кратак програм са намерно унетом логичком грешком и покушај да је пронађеш!
- Истражи како функционише дебаговање у програму Visual Studio Code!
- Научи како се поставља тачка прекида (breakpoint)!
- Преведи поруке о грешкама са енглеског на српски језик и објасни шта оне значе!

3.10 Исказ у програмирању

Већ знаш...

- препознаш основне елементе програмског окружења,
- разликујеш уређивач, компајлер и прозор за извршавање,
- шта значи да се добије резултат на екрану.

ЗАДАЦИ И ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Објасни шта значи компајлирање програма!

2. Које су главне врсте грешака у програмирању?

3. Зашто је важно дебаговање?

4. Како се користи breakpoint?

5. Шта значи логичка грешка?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објасниш шта је исказ у програмирању, да разликујеш исказ за приказ на екрану од других наредби,
- разумеш како програми комуницирају са корисником.

У свету програмирања свака наредба коју задајемо рачунару назива се исказом. Искази су основни градивни елементи сваког програма – они рачунару говоре шта тачно треба да уради. Помоћу њих изводе се различите радње: израчунавање, чување вредности или – што је најважније за ову лекцију – приказивање порука и резултата на екрану. Приказивање информација представља први облик интеракције између програма и корисника. Када напишемо програм, важно је

не само да он ради већ и да може да прикаже шта је урадио. Управо зато учење исказа за приказивање на екрану представља кључни корак у развијању логичког и комуникативног мишљења код ученика.

У сваком програму који комуницира са човеком постоји потреба за приказивањем информација – било да је реч о резултату израчунавања, упутству, упозорењу или поруци о успеху. У ту сврху користе се посебни типови исказа који омогућавају слање порука на екран. Захваљујући њима, програм постаје разумљив и користан човеку који га користи. Када се такав исказ изврши, на екрану се појављује тачно оно што је наведено – текст, број или њихова комбинација. Овај приказ представља визуелни доказ да се нешто догодило у програму. На пример, можемо приказати резултат неке математичке операције или једноставно исписати поздравну поруку.

Поред самосталних порука, често се приказују и вредности које су претходно израчунате или унете. У том случају програм не само да извршава тражено израчунавање већ и саопштава резултат кориснику. Ово је изузетно важно за проверу – када корисник види шта је приказано, може да процени да ли програм ради онако како би требало. На тај начин успоставља се интеракција – корисник уноси податке, програм их обрађује, а затим приказује резултат. Без ове последње фазе програм би остао „нем“, без могућности да покаже шта се у њему дешава.

Веома је важно разумети да редослед исказа у програму одређује редослед приказивања порука. Програми се извршавају одозго надоле, редом, осим ако није другачије наведено. То значи да, ако желимо да се најпре прикаже једна, а затим друга информација, искази морају бити правилно поређани. Ако се њихов редослед замени, промениће се и резултат приказан на екрану. На тај начин ученици уче како програм „размишља“ – његова логика следи се строго и прецизно, а свака промена утиче на излаз који видимо.

Понекад желимо да се поруке приказују у засебним редовима, једна испод друге. У ту сврху постоје механизми којима се означава прелазак у нови ред. Помоћу њих може се контролисати изглед текста на екрану – да ли ће бити исписан у једном реду или распоређен у више редова, да ли ће између делова постојати размак или ће се све приказати заједно. Ово је корисно за визуелно уређивање информација како би излаз био уредан, читљив и пријатан за гледање. Управо се кроз овакве детаље учи да програмирање није само логика већ и комуникација са човеком који посматра резултат.

Искази за приказивање међу првима се уче јер одмах дају видљив резултат. Они пружају осећај постигнућа – када се порука појави, ученик зна да програм ради. Помоћу њих започиње разумевање динамике програмског тока, јер су приказане поруке и резултати непосредан одраз онога што се дешава „у позадини“. Они омогућавају и повратну информацију – ако нешто није правилно приказано, то може указивати на логичку грешку. Зато ови искази нису само алат за приказивање већ и средство за проверу и побољшавање програма.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља исказ у програмирању?

2. Зашто је важан исказ за приказивање на екрану?

3. Како корисник сазнаје шта је програм урадио?

4. Шта се дешава ако се промени редослед исказа?

5. Осмисли сценарио са две поруке које морају бити приказане тачним редоследом!

6. Опиши како би изгледао излаз ако би све поруке биле приказане у истом реду!

7. Објасни зашто је приказивање на екрану важан део сваког алгоритма!

Запамти шта си научио!

- Искази су инструкције које програм извршава одређеним редоследом.
- Исказ за приказивање омогућава комуникацију са корисником.
- Програм приказује резултате или поруке на екрану.
- Важно је како су искази поређани јер од тога зависи какав ће бити излаз.
- Визуелно уређивање излаза помаже у разумевању онога што се дешава у програму.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи како се приказују поруке у мобилним апликацијама!
- Замисли програм који показује мени и објасни како би изгледао редослед приказивања!
- Напиши сценарио у коме приказана порука зависи од неке претходне вредности!
- Размисли како би изгледао један тест са више приказаних питања и резултата!

3.11 Псевдојазик

Већ знаш да...

- разликујеш бројчане и текстуалне вредности,
- изводиш једноставне математичке операције, препознајеш шта значи улаз и излаз података, задатак радиш поступно - једноставним логичким корацима, знаш да променљиве служе за чување података.

ЗАДАЧИ И ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Зашто је важно да приказана информација буде уредна?

2. Замисли програм који приказује поздравну поруку! Објасни шта представља сваки корак!

3. Осмисли ситуацију у којој треба да се прикаже резултат операције! Опиши логику!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- препознаш и запишеш исправну наредбу на псеудојезику; декларишеш променљиве и доделиш им вредности; извршиш унос и приказ података помоћу наредби унеси и прикажи; примениш аритметичке операције на псеудојезику; напишеш цео алгоритам у неколико корака.

Псеудојезик је поједностављен, описни облик програмирања који се користи као корак пре писања стварног кода. Помоћу њега учи се логика програмирања, без оптерећивања детаљима синтаксе. Он се не извршава непосредно на рачунару, већ служи за планирање и објашњавање корака које би програм следио. Посебно је користан приликом првог сусрета са програмирањем. Помоћу псеудојезика ученици стичу представу о томе како ради код „иза екрана“ – како се подаци уносе, обрађују и како се добија резултат.

Доделување вредност:

Прва и најосновнија операција која се користи у сваком програму јесте додела вредности. Упсеудојезику то се ради помоћу стрелице ←, која означава да вредност са десне стране треба доделити променљивој са леве стране. На пример:

Ова променљива цео број a ← 10

Ово значи да се креира променљива под називом a, која чува цео број и има почетну вредност 10. Ова наредба може да се користи за иницијализацију, али и за ажурирање вредности. На овај начин уводи се концепт чувања података, који представља основу сваког даљег израчунавања.

Унос података:

Потребно је да програм може да прима информације од корисника. У ту сврху користи се наредба „унеси“, која омогућава уношење вредности помоћу тастатуре. Тако програм постаје интерактиван. На пример:

унеси broj

Ово значи да ће корисник унети неку вредност, која ће затим бити сачувана у променљивој broj. Без улазних података програм би радио само са унапред дефинисаним вредностима, што би ограничило његову применљивост. Наредба „унеси“ представља корак ка динамичком понашању алгоритама.

Излаз података:

Следећи важан елемент јесте приказивање података. Помоћу наредбе „прикажи“ програм може да пошаље поруку или резултат кориснику. На пример:

прикажи „Вредност је: “, broj

Ова наредба ће приказати текст „Вредност је: “ заједно са тренутном вредношћу променљиве broj. Излаз је неопходан како би се видео резултат израчунавања или потврдило да све функционише онако како би требало. Без излаза корисник не би имао увид у рад програма.

Аритметичке операције:

Са додељеним вредностима и унетим подацима могу се обављати израчунавања. Псеудојезик подржава основне аритметичке операције, као што су +, −, *, /. Пример сабирања:

променљива цео број zbir ← a + b

Ово значи да се сабирају вредности променљивих a и b, а резултат се чува у променљивој zbir. Аритметичке операције представљају основу за математичке задатке, израчунавање оцена, бројање, статистику и још много тога. Помоћу њих програми добијају могућност да обрађују информације.

Комбиновањем уноса, излаза, доделе вредности и аритметичких операција добија се функционалан алгоритам. На пример:

прикажи „Унеси два броја:“

унеси x, y

променљива цео број proizvod ← x * y

прикажи “Производ је: “, proizvod

Овај алгоритам омогућава кориснику да унесе два броја, након чега ће их програм помножити и приказати резултат. Помоћу оваквих примера учи се целовита структура решавања проблема применом алгоритамског начина размишљања. Ова знања представљаће основу за наредне, сложеније теме, као што су услови, петље и функције.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта означава симбол ← у псеудојезику?

2. Која реч се користи за унос вредности помоћу тастатуре?

3. Објасни шта ради наредба „прикажи“!

4. Напиши алгоритам који ће приказати квадрат унетог броја!

5. Затражи од корисника да унесе три броја и прикажи њихов збир!

6. Направи алгоритам који ће израчунати вредност израза (x + y) / 2!

7. Напиши алгоритам који ће приказати: „Добро дошли, [име]“, при чему корисник уноси име.

Запамти шта си научио!

- У псеудојезику се користе једноставне и интуитивне наредбе.
- Променљиве се креирају помоћу речи „променљива“, а вредност им се додељује симболом ←.
- Наредбом „унеси“ примају се подаци од корисника, а наредбом „прикажи“ приказују се резултати.
- Могу се изводити основне математичке операције: +, –, *, /.
- Комбинација улаза, израчунавања и излаза чини функционалан алгоритам.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Ако желимо додатно да формализујемо логику, можемо увести псеудојезик са структурама као што су **ако ... онда, понављај** и условима типа **x > y**. Ове конструкције омогућавају доношење одлука и понављање радњи. Такође се могу користити формати псеудокода који су слични стварним програмским језицима, као што су Python или C++, али без строгих синтаксичких правила. На тај начин псеудојезик постаје мост између логичког размишљања и практичног програмирања.

3.12 Секвенца од искази

Већ знаш да...

- напишеш једноставни исказ за приказ на екрану,
- разумеш појмове псеудојезик и инструкција,
- следиш логику основног програмског тока.

ЗАДАЧИ И ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Како се записује множење две променљиве?

2. Зашто је важно да се комбинују улаз, аритметика и излаз?

3. Напиши псеудојезик који ће израчунати разлику два броја!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- креираш другачији тип променљивих.

У програмирању секвенца исказа представља групу инструкција које се извршавају утврђеним редоследом – једна за другом, без условљавања или прескакања. То је основни механизам за изградњу програма јер омогућава дефинисање јасних и логичних корака. Када се програм чита од почетка до краја, сваки исказ извршава одређену функцију, и то редоследом којим се појављује у коду. Овај след назива се секвенцијално извршавање, а саме инструкције чине секвенцијални блок. Секвенца је полазна тачка од које се програмска логика даље надограђује условима и петљама, па је њено правилно разумевање од суштинског значаја.

Сваки исказ у програму представља наредбу коју рачунар извршава. Када су ти искази поређани један испод другог, без додатних контролних структура као што су услови или понављања, они чине секвенцу. У оваквом блоку редослед је изузетно важан – рачунар извршава инструкције линеарно, одозго надоле.

Резултат једне наредбе може да утиче на следећу. Ако се нека наредба постави прерано, пре него што се припреме потребни подаци, може доћи до грешке или нежељеног резултата. Зато се препоручује да се програмирање започне писањем малих, логички повезаних секвенци инструкција.

ПРИМЕР 1

променљива цео број a ← 10

променљива цео број b ← 5

променљива цео број zbir ← a + b

прикажи „Збир је: “, zbir

У овом примеру сваки ред зависи од претходног. Ако се редослед промени, на пример, ако се наредба cout постави пре израза zbir ← a + b, доћи ће до грешке. Зато секвенцу треба пажљиво планирати.

Добро организована секвенца омогућава лако читање кода и предвидљиво извршавање. У образовном контексту то значи да ученици уче да размишљају корак по корак и логички, као и да разумеју како се прелази са једне операције на другу. На тај начин развија се не само програмерска већ и **аналитичка способност**. Ово је корисно не само за кодирање већ и за решавање проблема из свакодневног живота, у којима такође важи принцип логичког следа.

ПРИМЕР 2

прикажи “Унеси два цела броја: “

унеси x, y

променљиву цео број proizvod ← x * y

прикажи “Производ је:“, proizvod

У овој секвенци корисник најпре уноси вредности, затим се обавља израчунавање и на крају се приказује резултат. Сваки ред мора бити постављен одговарајућим редоследом како би се добио исправан резултат.

Поред тога, секвенца омогућава **модуларизацију логике** – то јест, груписање операција које имају заједничку намену. На тај начин могу се створити логички блокови који се лако разумеју и тестирају. Због тога програмери често организују програм у **функционалне сегменте**, од којих се сваки састоји од сопствене секвенце. Чак и када се пређе на сложеније структуре, као што су услови и петље, сваки њихов део биће састављен од секвенце инструкција.

ПРИМЕР 3

променљива цео број m ← 8
променљива цео број n ← 3
променљива цео број razlika ← m - n
прикажи „Разлика је:“, razlika

Поново, наредбе морају да се извршавају редом. Ако покушамо да користимо разлику пре него што се она израчуна, појавиће се грешка.

ЗАКЉУЧАК

Секвенца исказа представља основу сваког програма. Разумевање начина на који се инструкције извршавају одређеним редоследом кључни је корак у учењу програмирања. Без овог концепта није могуће прећи на сложеније програмске логике. Зато ученици морају да науче не само да пишу инструкције већ и да их логично и смислено организују у секвенцу. То ће им омогућити да стварају стабилне, функционалне и читљиве програме.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Напиши програм који учитава два цела броја и исписује њихов збир и производ!

2. Направи секвенцу у којој се најпре уносе вредности, затим се обавља неко израчунавање, а потом приказује резултат!

3. Промени редослед инструкција и упореди излаз!

4. Зашто је важно да се променљива не користи пре него што се дефинише?

Запамти шта си научио!

- Добро написана секвенца представља основу стабилног програма. Напиши програм који учитава два цела броја и исписује њихов збир и производ!
- Направи секвенцу у којој се најпре уносе вредности, затим се обавља неко израчунавање, а потом приказује резултат!
- Промени редослед инструкција и упореди излаз!
- Зашто је важно да се променљива не користи пре него што се дефинише?

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи како се секвенцијално извршавање примењује у роботици!
- Препознај секвенце у свакодневним задацима, као што су кување и одлазак пешке у школу!
- Покушај да напишеш програм који користи три различите секвенце!
- Испробај шта се дешава ако се секвенца прекине у средини!

3.13 Променливи и константи

Већ знаш да...

- препознаш редоследно извршавање,
- напишеш секвенцу исказа.

ЗАДАЧИ И ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Објасни шта је секвенца исказа!

2. Зашто је важан редослед инструкција?

3. Напиши три инструкције које су повезане логички!

4. Шта ће се догодити ако се разместе инструкције?

5. Које су предности секвенцијалног извршавања?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- креираш променљиве и константе,
- доделиш почетну вредност променљивих,
- разликујеш начин функционисања код променљивих и константи.

У сваком програму користе се информације које се мењају или остају исте. Да би рачунар могао да ради са овим информацијама, оне се чувају у такозваним променљивим или **константама**. Променљиве служе за чување података који се могу мењати током извршавања програма, док се константе користе за податке чије вредности не смеју да се мењају. Ове две структуре представљају основу сваке логике у програмирању и омогућавају флексибилно управљање информацијама.

Променљива је именовано место у меморији рачунара које се користи за чување вредности која се може мењати. Свака променљива има **име (идентификатор), тип података и вредност**. Име служи за препознавање променљиве, тип указује на то какве вредности она може да садржи, а сама вредност представља конкретан број или текст. Пре употребе променљива мора бити декларисана, односно рачунару треба

саопштити какав ће податак чувати. Поред тога, променљивој се одмах може доделити почетна вредност – то се назива **иницијализација.**

ПРИМЕР 1

```
int vozrast = 16;
```

У овом примеру дефинише се променљива под називом „vozrast“, која садржи цео број. Тип int указује на то да вредност сачувана у овој променљивој може бити цео број, односно број без децимала. Почетна вредност која се додељује променљивој јесте 16. Ова вредност назива се иницијална или почетна вредност и може се мењати током извршавања програма, у зависности од потреба задатка или интеракције са корисником. Када се променљива „vozrast“ једном декларише, она постаје доступна и може се користити у различитим деловима програма, свуда где је то потребно.

Након што је променљива дефинисана, она се може слободно користити за различите операције и израчунавања. На пример, ако се кориснику омогући да унесе нови узраст, вредност променљиве може се лако ажурирати унетом вредношћу. Променљива се такође може користити за математичке операције, као што су сабирање, одузимање или поређење са другим променљивим. Вредност коју променљива садржи може се у сваком тренутку приказати на екрану, што програм чини интерактивнијим и динамичнијим. Коришћењем оваквих променљивих програми постају флексибилнији и могу да се прилагођавају ситуацији или корисничком уносу. Управо зато променљиве представљају основни елемент програмирања јер омогућавају чување и мењање информација током извршавања програма

ПРИМЕР 2

```
int const broj = 8;
```

У овом примеру дефинише се константна променљива под називом „broj“, која садржи цео број. Тип ове променљиве је int, што значи да њена вредност мора бити цео број, док спецификатор const указује на то да се једном додељена вредност ове променљиве не може мењати током извршавања програма. Почетна вредност у овом случају је 8 и она остаје фиксна и непроменљива током целокупног извршавања програма. Употреба оваквих константи посебно је корисна када су вредности које треба користити у програму унапред познате и никада не треба да се мењају.

Константне променљиве веома су важне јер програм чине читљивијим, јаснијим и безбеднијим. Пошто се вредност константе не може променити након њене иницијализације, програмер може бити сигуран да неће доћи до случајне или ненамерне промене њене вредности, чиме се спречава могућност настанка грешака или неспоразума у коду. фиксне вредности као што су математичке константе (на

▸▸▸

пример, π или гравитацијска константа), али и за параметре који остају исти током целог програма, као број дана у недељи или број месеци у години. Коришћење оваквих константи доприноси и лакшој промени вредности у будућности, јер се декларишу само на једном месту у коду. Ово омогућава лакше одржавање и менаџирање програма, посебно ако је програм сложен и има много редова кода.

ЗАКЉУЧАК

Променљиве и константе представљају кључне градивне елементе сваког програма. Оне омогућавају чување података и управљање њима, без обзира на то да ли су променљиви или непроменљиви. Помоћу њих извршавају се бројне логичке операције, израчунавања и интеракције са корисником. Разумевање начина на који се променљиве и константе декларишу, користе и штите представља важан корак у процесу учења програмирања. Оне су алати помоћу којих програм „памти“ и обрађује информације.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Декларишти променљиву visina и додели јој вредност!

2. Креирај константу MAX_OCENKA и додели јој вредност 5!

3. Шта ће се догодити ако покушаш да промениш вредност константе?

4. Зашто је важно да променљиве имају описна имена?

Запамти шта си научио!

- Променљиве се користе за вредности које се мењају.
- Константе чувају вредности које се не мењају.
- Свака променљива има тип, име и вредност.
- Иницијализација је додељивање почетне вредности.
- Константе се декларишу помоћу const и не могу се мењати.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи како се променљиве користе у играма!
- Покушај да направиш програм који користи и променљиве и константе!
- Прочитај шта се дешава ако употребиш променљиву без иницијализације!
- Упореди како различити програмски језици (C++, Python и Java) користе константе!
- Напиши код у којем променљива више пута мења своју вредност!

3.14 Типови променливи

Већ знаш да...

- разликујеш променљиве константи,
- разликујеш текст и број у програму.

ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Шта је променљива?

2. Шта је иницијализација променљиве?

3. Када се користи константа уместо променљиве?

4. Да ли је могуће променити вредност константе?

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- креираш различите типове променљивих.

У компјутерском програмирању сваки податак који се користи мора припадати одређеном **типу**. Тип податка говори програму колико меморије треба да издвоји и на који начин да обради тај податак. На пример, цео број 5 и текст „5" изгледају исто на екрану, али представљају потпуно различите податке. Управо зато избор одговарајућег **типа променљиве** представља важан корак у писању исправног и функционалног програма. Основни типови података са којима се најчешће сусрећемо јесу: цели бројеви (int), децимални бројеви (double или float), знакови (char) и логичке вредности (bool).

ПРИМЕР 1

```
double prosek = 4.75;
```

Овде променљива prosek чува оцену ученика/ученице, која је децимални број. Ако би се користио тип int, вредност 4,75 била би сведена на 4. Овде није реч о заокруживању као у математици, већ о „одсецању" децималног дела. Заправо, у овом случају задржава се само цео број.

Још један занимљив тип је char, који се користи за појединачне знакове, као што су слова или специјални симболи. Овај тип је погодан за обраду иницијала, ознака или симбола. Декларише се навођењем знака између једноструких наводника ('A', '%', '#'). Са њим се не обављају математичке операције, већ се користи за симболичко претсављање.

ПРИМЕР 2

```
char bukva = 'A';
```

Тип података char служи за чување појединачних знакова, као што су слова, цифре или симболи. На пример, char bukva = 'A'; дефинише променљиву под називом „bukva", која чува вредност A. Знакови се у језику C++ увек записују између једноструких наводника. Овај тип користи се при раду са текстуалним подацима на нивоу појединачних знакова, као и за проверу и поређење конкретних знакова у програму.

ПРИМЕР 3

```
string ime = "Ana";
```

У овом примеру дефинише се променљива типа string (низ знакова) под називом „ime". Овај тип променљиве користи се за чување текстуалних података, односно речи, фраза или реченица. У овом конкретном случају променљивој „ime" додељује се почетна вредност „Ana", што значи да ће, када се ова променљива употреби током извршавања програма, она садржати управо тај текстуални податак. Тип string веома је користан када је потребно сачувати и обрадити текстуалне информације, као што су имена, адресе, поруке и други слични подаци састављени од једног или више знакова.

Након што се променљива „ime" декларише, њена вредност може се мењати према потреби, односно може јој се доделити други текст или се постојећи текст може изменити. На пример, ако корисник унесе ново име, вредност променљиве биће ажурирана новом информацијом. Поред тога, текстуалне променљиве омогућавају извођење различитих операција, као што су спајање више речи (конкатенација), поређење текстова, претраживање знакова у низу и многе друге радње. То омогућава да програми буду динамичнији и флексибилнији јер су текстуалне информације кључне за комуникацију између корисника и рачунара.

ПРИМЕР 4

```
bool aktiviran = true;
```

У овом примеру дефинише се логичка променљива типа bool под називом „aktiviran". Логичке променљиве типа bool могу имати само две вредности: true (тачно) или false (нетачно). У датом случају променљива је иницијализована вредношћу true, што значи да је услов који ова променљива представља тренутно испуњен или активан. Логичке променљиве имају важну улогу у контроли тока програма јер од њихове вредности зависи како ће се наставити извршавање инструкција.

Употреба логичких променљивих неопходна је у ситуацијама када треба проверити да ли је неки услов испуњен. На пример, променљива „aktiviran" може указивати на то да ли програма. Ако ова променљива има вредност true, програм ће

▸▸▸

наставити да извршава одређене операције. Ако је, међутим, њена вредност false, програм ће извршити неку другу операцију или прескочити одређени поступак. На овај начин логичке променљиве омогућавају програмеру да програм доноси одлуке у зависности од тренутног стања услова, што га чини интерактивним, динамичним и прилагодљивим различитим ситуацијама и потребама корисника.

ЗАКЉУЧАК

Типови променљивих обезбеђују структуру и логику у програмирању. Правилним избором типа постижу се тачност, оптимално коришћење меморије и разумљивост кода. Цели бројеви (int), децимални бројеви (float и double), знакови (char) и логичке вредности (bool) представљају основу готово сваког програма. Разумевање њихове намене и међусобних разлика од кључног је значаја за сваког почетника у програмирању.

ПИТАЊА ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Који тип променљиве ћеш користити за чување броја страница у књизи?
2. Како се int и double разликују по прецизности?
3. Шта ће се догодити ако се double декларише целим бројем?
4. Чему служи char? Наведи пример.
5. Када се користи тип bool?

Запамти шта си научио!

- Сваки податак има свој тип.
- int се користи за целе бројеве.
- double се користи за децималне бројеве.
- char чува један знак.
- bool чува логичку вредност: true или false.
- Правилан избор типа значи бољи програм.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи типове float, long и short!
- Прочитај шта се дешава приликом претварања вредности из једног типа у други!
- Напиши пример у којем тип bool управља током програма!
- Да ли char може да представља и број? Истражи!
- Како се вредност типа double претвара у тип int?

3.15 Додељивање вредности променљиве

Већ знаш да...

- разликујеш променљиве према њиховом типу,
- препознајеш цео број, децимални број, знак и логичку вредност,
- поведеш нову променљиву са одговарајућим именом и типом.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Декларишти променљив узраст типа int.!

2. Напиши променљиву типа double double за чување вредности тежине!

3. Креирај char променљиву за иницијал!

4. Дефиниши bool који ће проверити да ли је број позитиван!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- додељујеш вредности променљивима,
- препознајеш преклапање вредности.

У свету програмирања свака променљива има не само име и тип већ и вредност. Ова вредност може бити одређена приликом креирања променљиве или касније, током извршавања програма. Поступак којим се поставља или мења вредност променљиве назива се додељивање вредности. У ту сврху користи се оператор =. Иако изгледа као математички знак једнакости, у програмирању = значи „додели". То значи да се израз са десне стране израчунава, а резултат чува у променљивој са леве стране.

Вредност се променљивој може доделити на неколико начина. Први начин је иницијализација, односно додељивање почетне вредности приликом креирања променљиве. Овај начин је једноставан и олакшава читање кода. На пример, ако желимо одмах да поставимо почетну температуру, можемо написати:

ПРИМЕР 1

```
int temperatura = 22;
```

У овом случају променљива temperatura типа int добија вредност 22 приликом самог креирања. То се назива иницијализација и најчешће се примењује када је вредност унапред позната.

Када променљива треба да добије вредност након неког израчунавања, она се најпре може декларисати без вредности, а затим јој се вредност додељује помоћу израза. На пример, ако желимо да израчунамо просечну оцену из два предмета.

ПРИМЕР 2

Овде је променљива prosek у почетку само декларисана, али јој се касније додељује вредност добијена изразом. То показује да променљива може динамички да добија вредност, што је важно за интерактивне програме.

Поред тога, вредност једне променљиве може се више пута мењати током извршавања програма. То значи да су променљиве флексибилне и да се њихов садржај може ажурирати. Ово се назива преписивање вредности и представља уобичајен поступак, посебно при бројању, акумулацији вредности или раду са условима.

ПРИМЕР 3

```
int rezultat = 0;
rezultat = rezultat + 5;
rezultat = rezultat * 2;
```

Прво, rezultat добија вредност 0. Затим се њена вредност повећава за 5, а потом множи са 2. Коначна вредност је 10. Ово је јасан пример преписивања вредности, које омогућава изградњу сложеније логике у коду.

Важно је напоменути да се са леве стране оператора = увек мора налазити променљива – нешто што може да промени вредност. Са десне стране може се налазити било који израз: бројеви, друге променљиве, математичке операције, па чак и позиви функција. Због тога је оператор = један од најважнијих елемената сваког програмског језика.

ЗАКЉУЧАК

Додељивање вредности променљивој представља основни и незаменљив алат у програмирању. Оно омогућава флексибилност, израчунавање и чување података. Без ове операције променљиве би остале празне и бескорисне. Разумевање разлике између иницијализације, израчунавања и преписивања вредности кључно је за прецизан и стабилан рад програма.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта значи иницијализација променљиве?

2. Када се користи оператор =?

3. Шта ће се догодити ако се покуша да се додели вредност константе?

4. Објасни разлику између x = x + 1 и x = 1!

5. Може ли десна страна израза да укључује више променљивих?

Запамти шта си научио!

- не значи једнакост, већ додељивање.
- Вредност се додељује с десна на лево.
- Променљива може да се промени више пута.
- Додељивање је основа сваког рачунског алгоритма.
- Иницијализација омогућава чист и организован код.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи шта је „chain assignment“: a = b = c = 5!
- Шта се догађа ако се додељује израз са различитим типовима?
- Покушај да иницијализираш const променљиву – шта се догађа?
- Да ли је x = x + 1 легално математички? А програмски?
- Колико вредности може да има једна променљива у току програма?

3.16 Приказ вредности променљиве

Већ знаш да...

- креираш и додељујеш вредности променљивих,
- користиш основне типове: целе, децималне и знаковне,
- разликујеш текстуалне и нумеричке податке.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Декларишти и иницијализирај променљиву godini са вредношћу 18!

2. Креирај променљиву cena, додели јој вредност 120, па је увећај за 30!

3. Изрази формулу povrsina = dolzina * sirina!

4. Променљива zbir нека се добије као збир од broj1 и broj2!

5. Додели вредност променљивој након учитавања податка са тастатуре!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- приказујеш вредности променљивих на екрану,
- користиш основне команде за испис,
- форматираш излаз помоћу специјалних знакова,
- комбинујеш текст са променљивима у приказу,
- приказујеш више променљивих у једној наредби,
- разумеш како ради испис у новом реду или табулацијом.

Приказивање вредности променљиве представља једно од најважнијих средстава комуникације између програма и корисника. Помоћу њега резултати израчунавања, стање података или поруке о грешкама могу се јасно приказати на екрану. То се постиже употребом посебних наредби за испис, при чему се у програмском језику C++ најчешће користи cout. Иако је реч о једноставној операцији, правилно приказивање вредности има велики значај за читљивост, тачност и професионални изглед програма.

У програмима који се израђују у C++-у, на почетку стоји:

```
#include <iostream>
using namespace std;
```

Наредба **#include <iostream>** говори компајлеру да укључи стандардну библиотеку за улазно-излазне операције у језику C++, која се налази у датотеци заглавља iostream (input-output stream). Она омогућава коришћење основних објеката, као што су cin за унос података помоћу тастатуре, cout за приказивање на екрану и cerr за приказивање грешака. Без ове наредбе програм не би знао шта значе cout или cin, па би се током компајлирања појавиле грешке.

Наредба **using namespace std**; говори програму да ће се користити имена из простора имена std (standard), без потребе да се испред њих сваки пут пише std::. На пример, уместо std::cout, можемо користити само cout. То почетницима поједностављује синтаксу и чини код лакшим за читање, али је у већим пројектима боље користити std::cout како би се избегли сукоби са другим просторима имена.

Најосновнији начин приказивања вредности јесте употреба објекта cout. Назив cout означава character output (излаз знакова) и припада простору имена std, што значи да се користи као std::cout или само као cout ако је наведена наредба using namespace std;. Он омогућава програмеру да непосредно на екрану прикаже текст, бројеве или вредности променљивих. Наредба cout увек се користи са оператором <<, који усмерава податак ка екрану. На пример:

ПРИМЕР 1

n.cpp

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int broj = 7;
8      cout << "Vrednost broja je: " << broj << endl;
9      return 0;
10 }
11
```

Ова наредба ће на екрану приказати: „Вредност броја је: 7“. Примећује се да се текст и променљива могу комбиновати у истом реду, што омогућава динамичан и разумљив приказ.

У већини случајева потребно је приказати више вредности или организовати податке у редове или табелу. Зато се користе посебни знакови, као што су \n за прелазак у нови ред и \t за табулацију. Ови такозвани escape знакови не приказују се непосредно, већ утичу на изглед излаза. На пример:

ПРИМЕР 2

```cpp
#include <iostream>

using namespace std;

int main()
{
    double pi = 3.14159;
    cout << "Priblizna vrednost broja pi: " << pi << endl;
    return 0;
}
```

Излаз би био: Приближна вредност pi: 3.14159. У сложенијим ни случајевима, могу да се користе додатне библиотеке као <iomanip> како би се контролисао број децимала или за усаглашавање текста.

ПРИМЕР 3

```cpp
#include <iostream>

using namespace std;

int main()
{
    int a = 5, b = 10;
    cout << "a = " << a << "\tb = " << b << endl;
    return 0;
}
```

Излаз ће бити уређен у облику табеле: a = 5 b = 10. Ово показује да \t раздваја информације у облику табеле, док endl започиње нови ред, исто као и \n.

Испис се веома често користи за проверу вредности, дебаговање или обавештавање корисника о одређеним догађајима. У таквим ситуацијама од кључног је значаја да информације буду приказане јасно и сажето. Форматирање се може побољшати задавањем фиксне ширине, заокруживањем децималних бројева или поравнавањем података у колонама.

Све ове технике омогућавају јасан и уредан приказ информација, што је важно не само због естетике већ и због тачности и правовременог разумевања резултата програма. То је посебно значајно приликом анализе великих скупова података, рада са табелама или праћења логичких токова.

ЗАКЉУЧАК

Приказивање вредности променљиве неопходан је део сваког програма. Оно представља везу између корисника и машине. Његовом правилном употребом програм постаје читљив, функционалан и професионалан. Комбиновање променљивих, текста и посебних знакова омогућава флексибилан и прилагођен излаз, а овладавање овом вештином представља корак ближе развоју стварних апликација.

ПИТАЊА ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља функција cout и за шта се користи?

2. Који оператор се користи за приказ вредности?

3. У чему је разлика између \n и endl?

4. Како може да се прикаже вредност више променљивих у једној наредби?

5. Који је ефекат \t у испису?

Запамти шта си научио!

- Функција cout је основни начин за приказ вредности.
- Оператор << се користи за усмеравање ка екрану.
- Escape означава како \n и \t форматирају текст.
- Може да се комбинују текст и вредности.
- Приказ је важан за тачну комуникацију са корисником.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи библиотеку <iomanip> и функције setprecision и setw!
- Покушај да прикажеш резултате у табеларном облику!
- Користи испис за дебаговање – покажи вредности у сваком кораку!
- Направи програмски облик са линијама и заглављем!
- Прикажи текст са ћириличним знацима у C++!

3.17 Аритметичке операције и изрази

Већ знаш да...

- користиш променљиве и да им додељујеш вредности,
- приказујеш резултат на екрану,
- разликујеш целе и децималне бројеве,

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Декларишти променљиве ime и godini, прикажи их на екрану!
2. Користи \n за приказ вредности у новом реду!
3. Креирај променљиве за висину и тежину, и прикажи их уз објашњење!
4. Прикажи три броја за редом, који су раздвојени табулацијом!
5. Користи cout да прикажеш израз као 5 + 3 = 8!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- користиш аритметичке операције: сабирање, одузимање, множење, дељење,
- применуваш правилан редослед операција,
- користиш заграде за контролу израза,
- разумеш разлику између целобројног и децималног дељења,
- користиш оператор за остатак,
- решаваш изразе који комбинују више операција.

Аритметичке операције представљају основу готово сваког израчунавања у програмирању. Оне омогућавају рад са бројевима – било да је реч о сабирању, одузимању, множењу или дељењу – и примењују се у готово свим логичким решењима, од најједноставнијих до најсложенијих. У овој лекцији биће објашњено како се формирају и израчунавају изрази, који се оператори користе и како редослед операција утиче на резултат.

Аритметичке операције обухватају стандардне математичке симболе: + за сабирање, – за одузимање, * за множење, / за дељење и % за остатак при дељењу. Ови оператори користе се у комбинацији са вредностима и променљивим ради формирања израза. Основни израз може изгледати овако: a + b * c, где су a, b и c променљиве.

ПРИМЕР 1

```
1   #include <iostream>
2
3   using namespace std;
4
5   int main()
6   {
7       string ime;
8       int a = 5, b = 3, c = 2;
9       int rezultat = a + b * c;
10      cout << "Rezultat je: " << rezultat << endl;
11      return 0;
12  }
13
```

У овом случају, множење има приоритет над сабирањем, па се рачуна b * c = 6, а затим a + 6 = 11.

Увек када у једном изразу постоји више операција, мора се узети у обзир приоритет оператора. Правило гласи: множење и дељење имају виши приоритет од сабирања и одузимања. Ако желите да промените редослед израчунавања, користе се заграде.

ПРИМЕР 2

```
#include <iostream>

using namespace std;

int main()
{
    int a = 5, b = 3, c = 2;
    int rezultat = (a + b) * c;
    cout << "Rezultat je: " << rezultat << endl;
    return 0;
}
```

Овде, прво ће се извршити a + b = 8, а затим 8 * c = 16, због заграда.

Посебно је важно разумети разлику између целобројног и децималног дељења. Када се користе цели бројеви, резултат дељења увек је цео број, без децималног дела. Да би се добио прецизан децимални резултат, потребно је да бар један од бројева буде децималног типа (double или float).

ПРИМЕР 3

```
#include <iostream>

using namespace std;

int main()
{
    int a = 5, b = 2;
    double rezultat = (double)a / b;
    cout << "Decimalni rezultat: " << rezultat << endl;
    return 0;
}
```

Овде је а претворен у децимални број пре дељења, па ће резултат бити 2.5, а не 2.

Поред тога, оператор % користи се за добијање остатка при дељењу. На пример, израз 7 % 3 даје резултат 1, јер се број 7 при дељењу бројем 3 дели два пута, уз остатак 1. Овај оператор посебно је користан при решавању задатака повезаних са парношћу бројева, бројањем и провером дељивости.

ЗАКЉУЧАК

Аритметички изрази саставни су део логике сваког програма. Правилно разумевање оператора, њиховог приоритета и употребе заграда омогућава добијање тачних и предвидљивих резултата. Способност комбиновања операција и контролисања редоследа њиховог извршавања кључна је за решавање задатака и развој стварних апликација.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Које аритметичке операторе познајеш?

2. Који оператор има највиши приоритет?

3. Када се користе заграде у изразима?

4. Објасни разлику између a / b и (double)a / b!

5. Шта ће бити резултат од 8 % 3?

Запамти шта си научио!

- Основни оператори су: +, -, *, /, %.
- Приоритет операција утиче на резултат.
- Заграде се користе да се промени редослед.
- Целобројно дељење не враћа децимале.
- Оператор % враћа остатак.
- Тачност зависи од типа променљивих.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи функције pow(), sqrt(), fabs() од <cmath>!
- Напиши израз који израчунава средњу вредност три броја!
- Реши изразе са више заграда и операција!
- Направи миникалкулатор који користи свих пет операција!
- Анализирај како грешке у редоследу утичу на тачност резултата!

3.18 Уноси податке

Већ знаш да...

- креираш променљиве,
- додељујеш вредности променљивој,
- приказујеш резултат на екрану.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Израчунај резултат од 4 + 2 * 3 и објасни редослед!

2. Напиши израз који ће сабрати x и y, а затим поделити са z!

3. Израчунај остатак дељења 25 са 6!

4. Претвори a = 7 у децимални и подели са 2!

5. Напиши израз где је a = (b + c) * d!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- користиш наредбе за унос података помоћу тастатуре,
- препознаш различите типове унетих података,
- спроведеш основну проверу исправности уноса,
- разликујеш текстуални и нумерички унос,
- унесеш више података одједном,
- отклониш грешке при уносу.

У сваком интерактивном програму постоји потреба да корисник унесе податке. Без тих података програм не би могао да буде флексибилан нити користан. Уношење информација представља полазну тачку сваког процеса обраде, без обзира на то да ли је реч о бројевима, тексту или њиховој комбинацији. Овим поступком програм добија приступ вредностима које је дефинисао корисник, а које се затим могу користити за израчунавања, доношење одлука на основу услова и различите логичке токове. У овој лекцији научићемо како се подаци уносе и обрађују, као и како се отклањају могуће грешке при уносу.

Основни начин уношења података помоћу тастатуре подразумева употребу посебних функција или оператора који читају вредности и уписују их у променљиве. У многим програмским језицима, као што је C++, у ту сврху користи се објекат cin. Његовом употребом програм добија могућност да „ослушкује“ унос са тастатуре и да унету вредност сачува у одређеној променљивој. То је важно за развој интерактивних апликација које реагују на унете податке.

ПРИМЕР 1

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int broj;
8      cout << "Unesi jedan broj: ";
9      cin >> broj;
10     cout << "Uneli ste broj: " << broj << endl;
11     return 0;
12 }
13
```

У овом примеру програм омогућава кориснику да унесе цео број, који се затим поново приказује као потврда уноса.

Важно је разумети да тип података који се уноси мора одговарати типу променљиве у којој се чува. Ако корисник унесе текст када се очекује број, доћи ће до грешке. То наглашава потребу за основном провером – валидацијом уноса. Одговарајући механизми валидације зависе од програмског језика и сложености апликације, али се на почетном нивоу програм може осмислити тако да захтева поновни унос ако добије неважећу вредност.

ПРИМЕР 2

```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  int main()
7  {
8      string ime;
9      cout << "Unesi svoje puno ime: ";
10     getline(cin >> ws, ime);  // ws uklanja moguci prethodni newline
11     cout << "Tvoje ime je: " << ime << endl;
12
13     return 0;
```

Помоћу Co getline() се омогућава унос целог реда текста, заједно са празним местима између речи.

ЗАКЉУЧАК

Уношење података представља основу сваког интерактивног програма. Употребом објеката као што је cin, програм комуницира са корисником и реагује у складу са унетим вредностима. Разумевање типова података, валидације и начина читања више података од суштинског је значаја за писање поузданих и корисних апликација. Правилно управљање улазним информацијама чини програм стабилним и отпорним на грешке корисника.

ПИТАЊА ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта се догађа ако се у cin унесе текст уместо броја?

2. Како се уносе више вредности одједном?

3. Зашто је важна валидација уноса?

4. Који оператор се користи за уношење целог броја?

5. Која је улога getline()?

Запамти шта си научио!

- За уношење података користи се cin.
- Вредности морају одговарати типу променљиве.
- Валидација је потребна како би се избегле грешке.
- Помоћу функције getline() чита се цео ред текста.
- При стандардном уносу текста cin не чита празна места, па се за реченице користи getline().

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Унеси два броја и израчунај њихов збир!

2. Затражи од корисника да унесе број између 10 и 100. Понављај унос све док не унесе исправан број!

3. Унеси име и презиме ученика и прикажи их на екрану!

4. Напиши програм који захтева унос броја и проверава да ли је он паран!

5. Затражи унос три цела броја и прикажи њихову просечну вредност!

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- Истражи како се cin.fail() користи за проверу неисправног уноса!
- Напиши програм који омогућава унос и проверу лозинке!
- Направи мини-формулар за унос имена, узраста и адресе!
- Примени валидацију текстуалног уноса (на пример, најмање три знака)!
- Експериментиши са уносом децималних бројева типа double!

3.19 Поредбене операције

Већ знаш да...

- напишеш једноставне C++ програме,
- користиш променљиве,
- прикажеш вредности на екрану,
- напишеш основну математичку операцију.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Напиши програм који ће израчунати обим правоугаоника на основу дужина његових страница унетих помоћу тастатуре.

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- користиш упоредне операторе у C++,
- разликујеш различите типове поредбених релација,
- процениш да ли је израз истинит или неистинит,
- комбинујеш поредбене операторе са променљивима,
- користиш резултат поређења у програму.

Напиши програм који ће израчунати обим правоугаоника на основу дужина његових страница унетих помоћу тастатуре. Операције поређења у програмирању представљају основу за доношење одлука. Помоћу њих може се утврдити да ли су два броја једнака, да ли је један број већи или мањи од другог, као и да ли се вредности разликују. У језику C++ постоје шест основних оператора поређења: == (једнако), != (није једнако), < (мање), > (веће), <= (мање или једнако) и >= (веће или једнако). Ови оператори увек враћају Булову вредност, односно true (тачно) или false (нетачно), која се затим може користити у условним конструкцијама.

ПРИМЕР 1

```cpp
#include <iostream>
using namespace std;

int main() {
    int a = 5, b = 10;
    bool rezultat = a < b;
    cout << "Rezultat izraza a < b je: " << rezultat << endl;
    return 0;
}
```

Када се у језику C++ пореде вредности, њихов тип има важну улогу. Ако се пореде цели бројеви, поређење је непосредно. Међутим, приликом поређења децималних бројева или знакова треба водити рачуна о прецизности и ASCII вредностима. При поређењу текстуалних података (string) користи се посебна логика заснована на редоследу знакова. На тај начин могу се писати програми који пореде имена, оцене или друге податке из стварног живота..

ПРИМЕР 2

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    string ime1 = "Ana";
    string ime2 = "Bojan";
    cout << "Da li je ime1 < ime2? " << (ime1 < ime2) << endl;
    return 0;
}
```

Веома често резултат операције поређења чува се у променљивој типа bool, која се затим користи у другим деловима програма. На пример, на основу резултата поређења може се исписати различита порука, променити ток извршавања програма или извршити одређена радња. Булове вредности могу се комбиновати и са логичким операторима ради формирања сложенијих услова, што представља основу за следећу лекцију.

ПРИМЕР 3

```cpp
#include <iostream>
using namespace std;

int main() {
    int godina = 2025;
    bool prestupna = (godina % 4 == 0);
    cout << "Godina je prestupna? " << prestupna << endl;
    return 0;
}
```

Важно је вежбати са различитим вредностима и операторима како би се разумела логика поређења. Повећавањем сложености услова припремамо се за наредне кораке у програмирању – употребу условних конструкција (if/else), циклуса и логичких оператора. У почетној фази најбоље је увежбавати једноставна поређења бројева и стрингова како би се усвојило правилно писање оператора и разумели очекивани резултати.

ЗАКЉУЧАК

Оператори поређења неопходни су део сваког програма који треба да донесе одлуку. Они омогућавају проверу односа између вредности и представљају основу условних конструкција. Вежбањем и радом на практичним примерима стиче се разумевање начина на који се у програмирању доносе логичке одлуке.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Представи све операторе поређења реченицама на примерима из стварног живота!

2. Вежбај са бројевима од -100 до 100, користећи <, >, ==, !=!

3. Провери шта се дешава када упоређујеш стрингове са истим првим словима!

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта враћа оператор == у језику C++?

2. Који оператор се користи за проверу да ли се две вредности разликују?

3. Шта је променљива типа bool?

4. Коју вредност враћа операција поређења?

5. Напиши програм који ће проверити да ли је број 15 већи од броја 10!

6. Напиши програм који ће проверити да ли су два имена једнака!

7. Напиши програм који ће исписати „ДА“ ако је број мањи од 100!

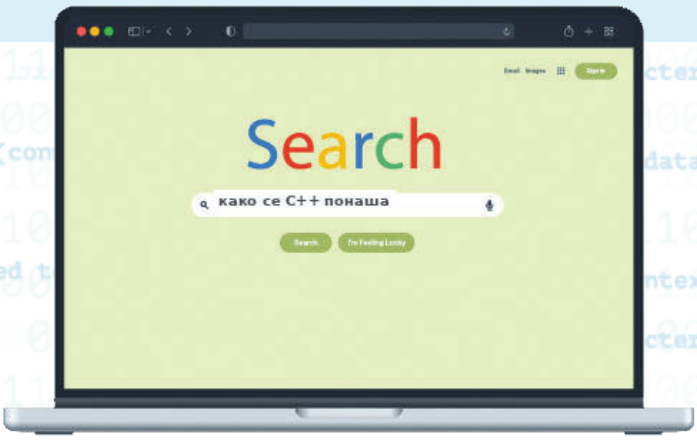
8. Напиши програм који ће упоредити два стринга и исписати који је од њих лексикографски „мањи“!

Запамти шта си научио!

- Оператори поређења користе се за поређење вредности.
- Сваки израз поређења враћа вредност true или false.
- Резултат се може сачувати у променљивој типа bool.
- Ови оператори представљају основу условних конструкција.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се C++ понаша приликом поређења различитих типова података (на пример, int са double)! Покушај да напишеш програм који комбинује поређења са логичким операторима, као што су && (и) и || (или), које ћемо изучавати у следећој лекцији!



3.20 Логички изрази

Већ знаш да...

- користиш поредбене операторе (==, !=, <, >, <=, >=),
- провериш да ли је израз истинит или неистинит,
- пишеш просте услове у програму,
- добијеш булову вредност поређења.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Како изгледа оператор за „мање или једнако“?

2. Напиши програм који ће проверити да ли је неки број паран!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- комбинујеш више услова користећи логичке операторе,
- објасниш шта су логички изрази и како функционишу,
- користиш операторе &&, || и ! у језику C++,
- разумеш правила приоритета операција,
- препознаш када се примењује кратко израчунавање (short-circuit evaluation)..

У програмирању се често сусрећемо са ситуацијама у којима је потребно истовремено проверити више услова. На пример, да ли је неки број већи од 10 и мањи од 20 или да ли је корисник унео тачну лозинку или тачан ПИН код. У таквим случајевима једноставна операција поређења није довољна. Уместо тога, услови се комбинују помоћу такозваних логичких оператора. Они омогућавају формирање логичких израза који враћају вредност тачно или нетачно на основу више поређења.

Логички оператори представљају основни алат у сваком програмском језику када се ради са условима. У језику C++ постоје три главна логичка оператора: && за логичку конјункцију (и), || за логичку дисјункцију (или) и ! за негацију (не). Ови оператори користе се за комбиновање једноставних услова у сложеније логичке изразе. На

пример, ако желимо да проверимо да ли је број x између 10 и 20, написаћемо x > 10 && x < 20. Овај израз ће вратити вредност true само ако су оба поређења тачна. Ако је бар једно од њих нетачно, цео израз ће имати вредност false.

ПРИМЕР 1

```
#include <iostream>
using namespace std;

int main() {
    int broj = 15;
    if (broj > 10 && broj < 20) {
        cout << "Broj je izmedju 10 i 20." << endl;
    }
    return 0;
}
```

Када се услови комбинују, важно је водити рачуна о приоритету оператора. У језику C++ логички оператор ! има највиши приоритет, затим следи &&, а на крају ||. То значи да ће се, ако се у једном изразу користе сва три оператора, најпре извршити негација, затим конјункција, па дисјункција. Да би се избегле нејасноће, често се користе заграде. На пример, израз !(x < 10 || x > 20) вратиће вредност true само ако је број x у опсегу од 10 до 20, укључујући и граничне вредности. Заграде омогућавају прецизну контролу редоследа израчунавања.

ПРИМЕР 2

```
#include <iostream>
using namespace std;

int main() {
    int x = 25;
    if (!(x < 10 || x > 20)) {
        cout << "x je izmedju 10 i 20." << endl;
    } else {
        cout << "x je izvan opsega." << endl;
    }
    return 0;
}
```

Једна важна особина логичких оператора у језику C++ јесте кратко израчунавање (short-circuit evaluation). То значи да се, ако је први услов код оператора && нетачан, други услов уопште не проверава јер је цео израз већ нетачан. Слично томе, ако је први услов код оператора || тачан, други се не проверава јер је довољно да један услов буде тачан како би цео израз имао вредност true. Ово не само да повећава ефикасност програма већ омогућава и писање услова који би у супротном могли да изазову грешке. Пример је провера да ли се вредност променљиве разликује од нуле пре извршавања дељења.

ПРИМЕР 3

```
#include <iostream>
using namespace std;

int main() {
    int broj = 0;
    if (broj != 0 && 100 / broj > 1) {
        cout << "Kolicnik je veci od 1." << endl;
    } else {
        cout << "Uslov nije ispunjen." << endl;
    }
    return 0;
}
```

Комбиновање логичких израза омогућава писање услова који су блиски природном језику. На пример, може се проверити да ли је корисник пунолетан и да ли има личну карту, односно да ли има дозволу или пратњу. Ови примери се у програмима представљају помоћу логичких оператора, што код чини јаснијим. Поред тога, употреба логичких израза у петљама и условним конструкцијама кључна је за стварање стварних и функционалних програма. Помоћу њих остварује се логика на којој се заснива свака аутоматизација.

ЗАКЉУЧАК

Логички изрази неопходни су за прецизну и условну контролу извршавања програма. Они омогућавају комбиновање више услова и рационално доношење одлука у коду. Правилна употреба оператора &&, || и !, као и вођење рачуна о њиховом приоритету и кратком израчунавању, представљају основу поузданих и ефикасних програма.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Истражи како се логички оператори комбинују са операторима поређења и како се користе у структурама if и while! Напиши програм у којем се примењују услови за пријављивање корисника (унос корисничког имена и лозинке) и анализирај шта се дешава ако један део услова није тачан!

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Која су три логичка оператора у C++?

2. Шта представља кратко померање (short-circuit evaluation)?

3. Који је приоритетни редослед логичких оператора?

4. Зашто се користи ! оператор?

5. Напиши програм који проверава да ли је број позитиван и паран!

6. Напиши програм који проверава да ли корисник има 18 или више година или има дозволу!

7. Напиши програм који проверава да ли је стринг празен или не и да ли има више од 3 слова!

8. Напиши програм која са && и || комбинује 3 услова за број!

Запамти шта си научио!

- Логички оператори &&, || и ! комбинују више услова.
- && враћа true само ако су оба израза истинита.
- || враћа true ако је барем један израз истинит.
- ! негира истинитост услова.
- Примењује се кратко израчунавање ефикасности.

3.21 Структура избора од две могућности

Већ знаш да...

- користиш поредбене операторе,
- испишеш текст на екрану,
- вршиш аритметичке операције.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Напиши програм који користи ! за негацију услова!

2. Провери шта се догађа када се комбинују више && и || без заграда!

3. Напиши своје примере из стварних ситуација (нпр. услов за упис у школу)!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- разликујеш шта представља избор између две могућности,
- препознајеш када треба користити структуру избора у програмирању,
- разумеш како логички услов утиче на извршавање кода,
- састављаш примере са структуром if-else у програмском језику C++,
- анализираш резултате различитих логичких услова.

У свакодневном животу често се доносе одлуке између две могућности. Такве ситуације срећу се и у свету програмирања. Структура избора између две могућности омогућава рачунару да донесе одлуку на основу неког услова и изврши један од два могућа блока наредби. Ова лекција разгледа основну логичку структуру која омогућа-

ва програмеру да имплементира одлуке. Она се назива if–else структура и представља део условног управљања током програма. Коришћење таквих услова чини код флексибилнијим и погоднијим за решавање реалних проблема.

Структура избора између две могућности један је од најједноставнијих, али веома важних контролних механизама у програмирању. Она омогућава програму да изабере између две алтернативе. То се остварује помоћу кључне речи if, којом се проверава да ли је неки логички услов испуњен. Ако је услов истинит, извршава се један блок наредби; у супротном, извршава се други блок, означен кључном речју else. Овакав начин контроле омогућава програму да реагује различито у зависности од унетих података или ситуација. Одлуке у програмима заснивају се на изразима за поређење и логичким изразима, о којима је било речи у претходној лекцији.

ПРИМЕР 1

```
#include <iostream>
using namespace std;

int main() {
    int broj;
    cout << "Unesi jedan broj: ";
    cin >> broj;

    if (broj > 0)
        cout << "Broj je pozitivan." << endl;
    else
        cout << "Broj nije pozitivan." << endl;

    return 0;
}
```

У овом примеру приказује се избор између две могућности: да ли је број већи од нуле или није. На основу резултата услова исписује се одговарајућа порука. Поред једноставних услова за поређење, у структури избора између две могућности може се комбиновати више услова коришћењем логичких оператора. Ова могућност чини услов сложенијим и омогућава прецизнију контролу одлука које програм доноси. На пример, може се проверити да ли је кориснички унос у задатом опсегу или да ли су истовремено испуњена два независна услова.

ПРИМЕР 2

```cpp
#include <iostream>
using namespace std;

int main() {
    int poeni;
    cout << "Unesi poene (0-100): ";
    cin >> poeni;

    if (poeni >= 51 && poeni <= 100)
        cout << "Polozeno." << endl;
    else
        cout << "Nije polozeno." << endl;

    return 0;
}
```

У примеру се проверава да ли је број унетих поена у оквиру минималног прага потребног за полагање. Обрати пажњу на то да је за одређивање опсега коришћен логички оператор &&.

Често се поставља питање да ли се може комбиновати више if-else израза. Одговор је потврдан – када је потребно доносити сложене одлуке, угнежђују се if структуре или се користи структура else if. Ипак, увек је добро задржати јасну и читљиву структуру кода, јер то омогућава лакше одржавање и разумевање логике програма.

ПРИМЕР 3

```cpp
#include <iostream>
using namespace std;

int main() {
    int broj;
    cout << "Unesi ceo broj: ";
    cin >> broj;

    if (broj % 2 == 0)
        cout << "Broj je paran." << endl;
    else
        cout << "Broj je neparan." << endl;

    return 0;
}
```

Програм из примера проверава да ли је број паран или непаран. Ова врста провере честа је у програмирању када се анализира врста броја или одређује да ли треба извршити одређену операцију.

Већ знаш да...

- Израчунаш аритметичке изразе и да користиш променљиве;
- користиш команду cin и cout;
- разликујеш поредбене операторе и логичке вредности.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља структуру if–else?

2. Који део кода се извршава ако услов није испуњен?

3. Како се користе логички оператори у if–else?

4. Шта значи broj % 2 == 0?

5. Када је боље да се користи else if уместо више if?

6. Напиши програм који ће проверити да ли је унети број негативан или не!

7. Напиши програм који ће проверити да ли је ученик/ца положио/ла испит ако има преко 60 поена!

8. Напиши програм који ће проверити да ли је температура у нормали (између 36 и 37,5 степени)!

9. Напиши програм који ће проверити да ли је слово самогласник!

10. Напиши програм који ће проверити да ли је година преступна!

Запамти шта си научио!

- Структура избора од две могућности (if–else) омогућава извршавање различитог кода у зависности од логичког услова. Она је основни механизам за контролу тока програма и користи се за изражавање одлука. Помоћу ње програм се може прилагодити различитим ситуацијама и унетим вредностима. Коришћење оператора за поређење и логичких израза омогућава креирање сложених услова. Примена структуре if-else чини код динамичнијим и функционалнијим у решавању стварних проблема.

3.22 Задаци за вежбање структуре if else

ЛАКИ ЗАДАЦИ (1 – 7)

1. Напиши програм који ће проверити да ли је унети број већи од 100!
2. Напиши програм који ће проверити да ли је дати број негативан!
3. Напиши програм који ће проверити да ли је унети број дељив са 5!
4. Напиши програм који ће проверити да ли корисник има навршених 18 година!
5. Напиши програм који ће проверити да ли су два унета цела броја једнака!
6. Напиши програм који ће исписати „Данас је викенд“ ако корисник унесе „субота“ или „недеља“, а у супротном „Радни дан“!
7. Напиши програм који ће проверити да ли је ученик/ученица добио/добила оцену 5 ако има више од 90 поена!

СРЕДЊЕ ТЕШКИ ЗАДАЦИ (8–14)

8. Напиши програм који ће одредити да ли је унети број дељив и са 3 и са 7!
9. Напиши програм који ће проверити да ли се број који је корисник унео налази у интервалу од 10 до 99!
10. Напиши програм који ће проверити да ли је корисник унео исправну лозинку (претпоставимо да је лозинка „tajna123“)!
11. Напиши програм који ће проверити да ли одређена особа има право гласа ако има најмање 18 година и држављанин је!
12. Напиши програм који ће исписати „паран и позитиван“ ако је број позитиван и паран, а у супротном „не испуњава услове“!
13. Напиши програм који ће проверити да ли три странице могу да образују троугао!
14. Напиши програм који ће приказати „одлично“ ако је број поена већи од 90, „добро“ ако је између 70 и 90, а у супротном „довољно“! (Користи структуру if-else само за два исхода, а трећи нека буде подразумевани исход.)редње тешки

ТЕЖИ ЗАДАЦИ (15 – 20)

15. Напиши програм који ће одредити који је од два унета броја већи!
16. Напиши програм који ће проверити да ли је дата температура у границама нормалне телесне температуре (од 36,0 до 37,5 степени)!
17. Напиши програм који ће проверити да ли је унето слово велико или мало!
18. Напиши програм који ће проверити да ли је дата година преступна!
19. Напиши програм који ће проверити да ли је унети број троцифрен и паран!
20. Напиши програм који ће проверити да ли су унети корисничко име и лозинка исправни (на пример: username = „admin“, password = „1234“)!

3.23 Внесување податоци

Већ знаш да...

- градиш једноставне условне исказе,
- користиш if и else за доношење одлука,
- креираш основне програме са улазом и

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Како изгледа израз за проверу, да ли је број у оптицају између 10 и 20?
2. Препиши примере и промени вредности да добијеш различит излаз!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- препознајеш шта значи да један исказ буде „угнежђен“ у други,
- разликујеш једноставне од угњеждених услова,
- користиш угњеждење за решавање сложенијих проблема,
- напишеш програм са вишекратним угњежђењем,
- разумеш где треба пажљиво да се користи угњежђење.

У процесу креирања условних структура у програмирању често се јављају ситуације у којима је потребно извршити проверу унутар друге провере. Ова појава, названа угнежђивање исказа, представља моћан концепт који омогућава креирање сложених логичких одлука. Угнежђени исказ је услов који се налази унутар другог услова, што омогућава програмирање помоћу вишеслојне логике. Овакве структуре често се примењују приликом моделовања стварних проблема у којима одређене радње треба предузети само ако је више услова испуњено одређеним редоследом.

Угнежђени услови могу бити у облику if структуре унутар друге if структуре или комбинације структура if и else if, при чему једна структура садржи другу. Угнежђени услови користе се када једна одлука зависи од претходно донете одлуке. У практичним примерима често је потребно прво проверити да ли је неки број позитиван, а затим да ли је паран. Ове две провере нису независне, јер друга провера нема смисла ако број није позитиван. Управо се овакве ситуације решавају угнежђивањем.

Почиње се једним условом, а унутар његовог тела поставља се други услов. На тај начин други услов проверава се само ако је први истинит, чиме се избегавају непотребне или нелогичне провере. Овакав начин организовања логике доприноси јаснијем коду и бољој контроли тока програма.

```cpp
#include <iostream>
using namespace std;

int main() {
    int broj;
    cin >> broj;
    if (broj > 0) {
        if (broj % 2 == 0) {
            cout << "Broj je pozitivan i paran." << endl;
        }
    }

    return 0;
}
```

Угнежђени услови нису ограничени само на два нивоа. Они се могу организовати и у више слојева, нарочито када се анализирају различита својства или пореди више променљивих. Међутим, што је више угнежђених услова, то код постаје тежи за читање и одржавање. Препоручљиво је умерено користити угнежђивање и примењивати га само када је логички оправдано. Угнежђивање се примењује када је редослед провера важан – неке услове треба проверити само ако су претходни услови испуњени. На пример, ако треба проверити да ли ученик има довољан број поена за полагање, а затим и довољан број часова, логично је да се други услов провери само ако је први испуњен.

```cpp
#include <iostream>
using namespace std;

int main() {
    int poeni, prisustvo;
    cin >> poeni >> prisustvo;
    if (poeni >= 50) {
        if (prisustvo >= 75) {
            cout << "Ucenik je polozio." << endl;
        }
    }

    return 0;
}
```

У стварним проблемским ситуацијама често је потребно направити избор између различитих сценарија. Тада се угнежђени искази могу комбиновати и са else гранама, што омогућава различито понашање програма у зависности од тога који су услови испуњени. Ово додатно повећава могућности угнежђивања, али захтева пажљиво организовање кода како не би дошло до логичких грешака. Структура if-else често се може комбиновати са већим бројем угнежђених услова, чиме се добија логика у облику стабла. Овакве ситуације типичне су приликом одређивања цене на основу више критеријума, као што су старосна група и чланство.

```cpp
#include <iostream>
using namespace std;

int main() {
    int godine;
    bool clan;
    cin >> godine >> clan;
    if (godine >= 18) {
        if (clan) {
            cout << "Cena je 100 dinara." << endl;
        } else {
            cout << "Cena je 150 dinara." << endl;
        }
    } else {
        cout << "Cena je 50 dinara." << endl;
    }

    return 0;
}
```

Поред предности, угнежђивање доноси и неколико изазова. Прекомерна употреба може учинити код тешко читљивим и склоним грешкама. Зато добра програмерска пракса препоручује да се угнежђивање користи само када је логички неопходно и када се не може заменити једноставнијом структуром. Ако код постане предугачак или тежак за праћење, може се размотрити издвајање делова логике у посебне функције. То омогућава поновну употребу кода и бољу организацију програма. Такође, треба водити рачуна о прегледности – правилно увлачити редове и користити коментаре који објашњавају где се који услов завршава.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља угнежђени исказ?

2. Када се угнежђена if структура користи у стварним ситуацијама?

3. Који су ризици прекомерног угнежђивања?

4. Како можемо учинити код прегледнијим приликом угнежђивања?

5. Напиши програм који ће проверити да ли је број дељив са 3, а затим да ли је паран!

6. Напиши програм који за ученика/ученицу уноси број поена и податке о присуству и одређује да ли је положио/положила!

7. Напиши програм који на основу старосне групе и чланства израчунава цену улазнице!

8. Напиши програм који проверава да ли ученик/ученица има одличан успех и да ли је добио/добила награде!

Запамти шта си научио!

- Угнежђени искази су услови постављени унутар других услова.
- Користе се када логика захтева да се провере извршавају само под одређеним условима.
- Прекомерно угнежђивање отежава читање и одржавање кода.
- Увек води рачуна о томе да буде јасно ком делу структуре сваки услов припада.
- Угнежђивање омогућава сложено одлучивање засновано на више повезаних критеријума.

3.24 Задаци за вежбање угнежђене структуре if else

ЛАКИ ЗАДАЦИ

1. Напиши програм који ће проверити да ли је унети број позитиван и дељив са 5.

2. Напиши програм који ће проверити да ли је број дељив са 10, и ако јесте, да ли је већи од 100.

3. Напиши програм који ће проверити да ли је број непаран, и ако јесте, да ли је једноцифрен.

4. Напиши програм који ће проверити да ли је број дељив са 4, а затим да ли је дељив и са 6.

5. Напиши програм који ће проверити да ли је број позитиван и дељив са 3.

6. Напиши програм који ће проверити да ли је слово самогласник, и ако јесте, да ли је велико слово.

СРЕДЊЕ ТЕШКИ ЗАДАЦИ

7. Напиши програм који ће проверити да ли је температура испод 0 и, ако јесте, да ли пада киша!

8. Напиши програм који ће проверити да ли је особа пунолетна и, ако јесте, да ли има возачку дозволу.

9. Напиши програм који ће проверити да ли је година преступна, а затим да ли је парна.

10. Напиши програм који ће проверити да ли име почиње великим словом, а затим да ли има више од пет слова.

11. Напиши програм који ће проверити да ли је ученик/ученица уписан/уписана у средњу школу, а затим да ли има просечну оцену већу од 4.

12. Напиши програм који ће проверити да ли је унети број двоцифрен и, ако јесте, да ли је његова последња цифра парна.

13. Напиши програм који ће проверити да ли је дан субота и, ако јесте, да ли је температура виша од 25 степени.

ТЕШКИ ЗАДАЦИ

14. Одељењски старешина жели да провери ко од ученика заслужује посебну похвалницу. За сваког ученика/ученицу познати су број поена освојених на тестовима и број урађених домаћих задатака. Напиши програм који ће проверити да ли ученик/ученица има више од 80 поена, а затим да ли је урадио/урадила све домаће задатке!

15. На крају полугодишта одељењски старешина жели да утврди који ученици могу да добију позитивну напомену о ангажовању. За сваког ученика/ученицу познати су број поена освојених из предмета и број неоправданих изостанака. Напиши програм који ће проверити да ли ученик/ученица има више од 60 поена и, ако има, да ли има мање од пет неоправданих изостанака!

16. У канцеларији за пријаву возачких испита за сваку особу проверава се да ли испуњава потребне услове. За сваку особу познати су њене године и податак о томе да ли има личну карту. Напиши програм који ће проверити да ли особа има 18 или више година, а затим да ли има личну карту!

3.25 Структура за избор између више могућности

Већ знаш да...

- разликујеш између услова са једном и више опција,
- користиш структуре за избор од две могућности (if-else),
- препознајеш логичке изразе у програмима,
- примењујеш поредбене операторе,
- решаваш једноставне програмске проблеме са условима.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Како се повезују if и else у вишеструким слојевима?

2. Напиши програм који ће проверавати да ли је унети број позитиван и већи од 100!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- разликујеш избор између више могућности од двоструке условне структуре,
- примењујеш структуру if-else if-else,
- препознајеш када је прикладно користити структуру switch,
- примењујеш вишеструке услове у програму,
- пишеш програм са више од две алтернативе,
- избегаваш логичке грешке приликом гранања.

Када решавамо проблеме из свакодневног живота или пишемо програм, често се суочавамо са ситуацијама у којима треба да изаберемо између више могућности. На пример, ако је температура испод 0 °C, облачи се зимска јакна; ако је између 0 и 20 °C – јесењи капут, а ако је изнад 20 °C – танка кошуља. То представља избор између више могућности. У програмирању се овакво понашање моделује помоћу структуре избора између више могућности, која представља логичан наставак структуре избора између две могућности. За разлику од структуре if-else, која омогућава само две путање извршавања, структура if-else if-else или switch омогућава обраду више алтернатива.

Овакав тип логике од кључног је значаја у програмирању и моделовању различитих сценарија. У овој лекцији биће објашњено како и када треба користити ове структуре, како избећи грешке приликом њихове употребе и како помоћу примера развити осећај за правилну примену разгранатих програмских токова. У стварним програмским ситуацијама често је потребно проверити више различитих услова, од којих само један треба да буде испуњен. У таквим случајевима користи се структура if-else if-else. Она омогућава редоследно проверавање услова, при чему програм улази у први блок чији је услов истинит, док се остали блокови занемарују. Важно је напоменути да се ова структура користи када се услови међусобно искључују, односно када треба извршити само једну грану. Овакав приступ обезбеђује јасан и уредан код, а истовремено повећава читљивост програма и олакшава његово одржавање.

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int broj;
6      cout << "Unesi jedan broj: ";
7      cin >> broj;
8
9      if (broj > 0)
10         cout << "Broj je pozitivan." << endl;
11     else if (broj < 0)
12         cout << "Broj je negativan." << endl;
13     else
14         cout << "Broj je nula." << endl;
15
16     return 0;
17 }
18
```

У примеру се користи структура if-else if-else како би се проверило да ли је број позитиван, негативан или једнак нули. Само један од ових услова може бити испуњен, а програм бира одговарајућу грану и занемарује преостале. Иако је структура if-else if-else моћна, понекад се провере обављају на основу вредности исте променљиве. У таквим случајевима једноставнија и ефикаснија алтернатива јесте структура switch. Она омогућава избор блока кода у зависности од вредности целог броја или знака. Кључни елемент ове структуре је case, који представља једну могућност, док се

default користи као последња опција ако ни један case не одговара. Ова структура побољшава читљивост, посебно када је реч о много услова који се проверавају на истој променљивој.

```cpp
#include <iostream>
using namespace std;

int main() {
    int dan;
    cout << "Unesi broj od 1 do 7: ";
    cin >> dan;

    switch (dan) {
        case 1: cout << "Ponedeljak"; break;
        case 2: cout << "Utorak"; break;
        case 3: cout << "Sreda"; break;
        case 4: cout << "Cetvrtak"; break;
        case 5: cout << "Petak"; break;
        case 6: cout << "Subota"; break;
        case 7: cout << "Nedelja"; break;
        default: cout << "Nema takvog dana!";
    }

    return 0;
}
```

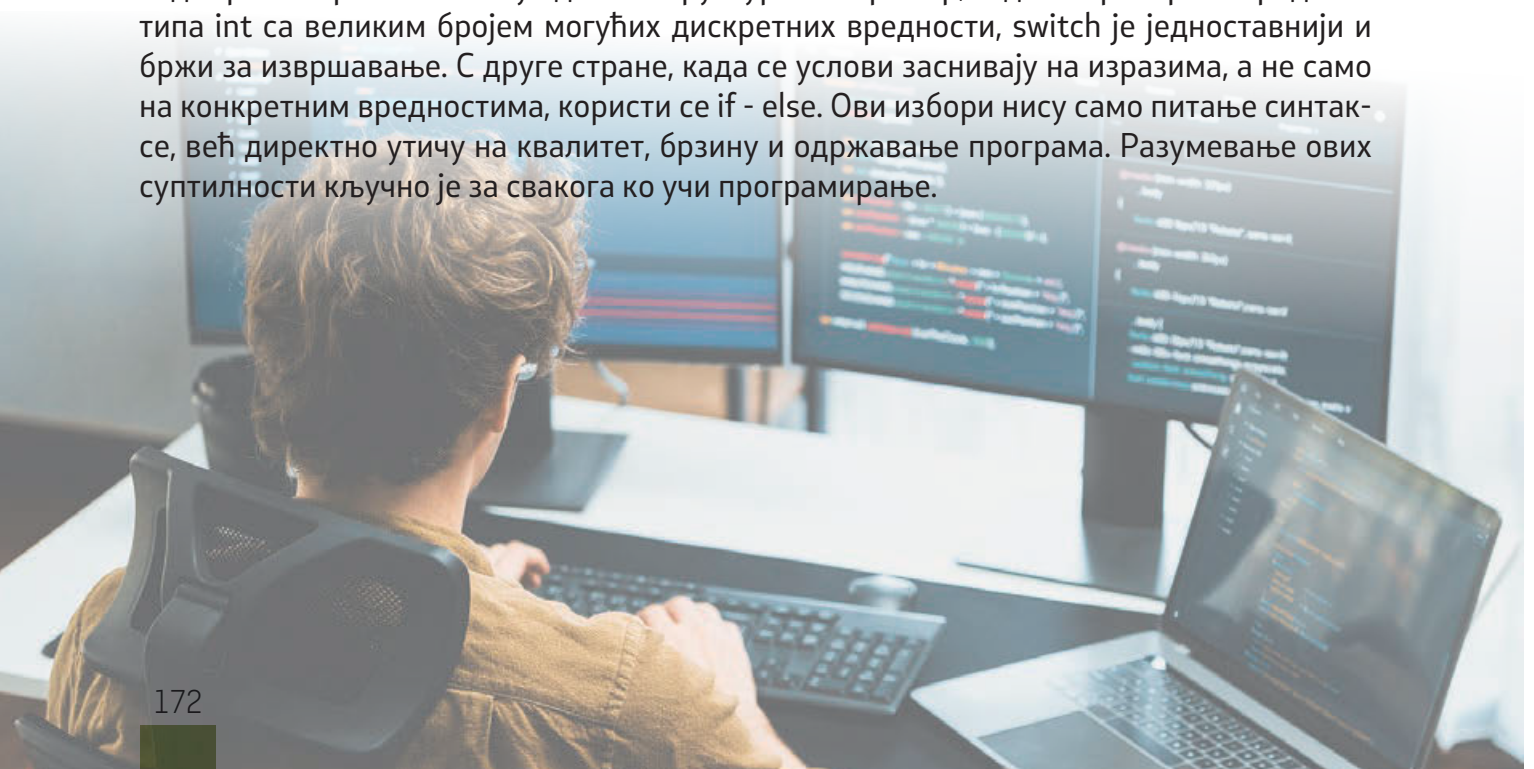
Овај пример користи switch за избор дана у недељи. Ово је добар пример када се избор врши према вредности једне променљиве и за сваку вредност се предузима одговарајућа радња.

У случајевима када се проверава велики број услова, важно је водити рачуна о редоследу услова и избегавати њихово преклапање. Прво треба проверити најограниченије (најстроже) услове, а затим шире. Такође, честа грешка је изостављање else, због чега може доћи до непредвиђеног извршавања више блокова. Због тога се препоручује коришћење добро документованог и читљивог кода. Поред тога, променљиве укључене у услове треба изабрати тако да обезбеђују логичку доследност и лако отклањање грешака. Пажљивим распоређивањем услова смањује се ризик од логичких грешака и добија јаснији програмски ток.

```cpp
#include <iostream>
using namespace std;

int main() {
    int poeni;
    cout << "Unesi broj poena (0-100): ";
    cin >> poeni;

    if (poeni >= 90)
        cout << "Ocena: 5" << endl;
    else if (poeni >= 75)
        cout << "Ocena: 4" << endl;
    else if (poeni >= 60)
        cout << "Ocena: 3" << endl;
    else if (poeni >= 50)
        cout << "Ocena: 2" << endl;
    else
        cout << "Ocena: 1" << endl;

    return 0;
}
```

Пример показује избор између више оцена на основу броја поена. Важно је уочити да су услови распоређени строгим редоследом: од највиших ка најнижим. На тај начин омогућава се тачна категоризација.

Поред правилне употребе if - else if - else и switch, од суштинског је значаја знати када треба користити сваку од ових структура. На пример, када се проверава вредност типа int са великим бројем могућих дискретних вредности, switch је једноставнији и бржи за извршавање. С друге стране, када се услови заснивају на изразима, а не само на конкретним вредностима, користи се if - else. Ови избори нису само питање синтаксе, већ директно утичу на квалитет, брзину и одржавање програма. Разумевање ових суптилности кључно је за свакога ко учи програмирање.



ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља структура if - else if - else и када се користи?

2. Објасни разлику између if и switch структура!

3. Зашто је важно да се услови распореде од најстрожих ка најопштим?

4. Када се препоручује коришћење switch уместо if?

5. Напиши програм који ће приказати сваку сезону на основу унетог броја месеца!

6. Напиши програм који ће дати опис ученика према броју поена (1 – 100)!

7. Напиши програм који ће извршити проверу оцене и приказати поруку („Одлично“, „Добро“ итд.)!

8. Напиши програм који ће приказати врсту транспорта према унетом типу (1 – Аутомобил, 2 – Авион, 3 – Воз)!

Запамти шта си научио:

- Структура if - else if - else се користи када треба да се изабере само једна од више могућности.
- Структура switch омогућава избор према тачној вредности променљиве.
- Услови увек треба да се постављају пажљиво и логично.
- У структури if-else if-else извршава се само један блок кода.
- У структури switch сваки case мора да се заврши наредбом break како би се спречило извршавање следећег case блока.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

Поред основних структура, у напредном програмирању користе се и угнежђени switch изрази, као и комбинације if услова са логичким операторима (&&, ||). У неким програмским језицима постоје и конструкције као што су match или pattern matching, које омогућавају сложенији избор грана. Такође, у функционалном програмирању избори се често моделују као структуре података, што омогућава виши степен апстракције.

3.26 Задаци за вежбање структуре избора између више могућности

Лесни задачи

1. Напиши програм који ће проверити да ли је унети број позитиван, негативан или нула!
2. Напиши програм који ће приказати дан у недељи на основу унетог броја од 1 до 7!
3. Напиши програм који ће проверити да ли ученик има оцену 1, 2, 3, 4 или 5 и исписати одговарајући опис!
4. Напиши програм који ће приказати тип одеће у зависности од унете температуре (у °C)!
5. Напиши програм који ће исписати назив месеца на основу унетог броја од 1 до 12!
6. Напиши програм који ће исписати поруку на основу унетог степена образовања (1 – основно, 2 – средње, 3 – високо)!
7. Напиши програм који ће описати ниво воде у резервоару на основу процента (низак, средњи, висок)!
8. Напиши програм који ће одредити временску зону на основу броја часова у односу на GMT!
9. Напиши програм који ће дати препоруку за пиће у зависности од узраста (испод 14 – сок, до 18 – безалкохолно пиће, изнад 18 – кафа)!
10. Напиши програм који ће приказати тип рачуна (1 – струја, 2 – вода, 3 – телефон)!

Средно тешки задачи

11. Напиши програм који ће израчунати цену превоза у зависности од удаљености (испод 10 km, до 50 km, изнад 50 km)!
12. Напиши програм који ће класификовати музички жанр на основу унетог броја (1 – рок, 2 – џез, 3 – поп, 4 – класика)!
13. Напиши програм који ће одредити опис квалитета ваздуха на основу AQI вредности (0–50 – одличан, 51–100 – добар итд.)!
14. Напиши програм који ће исписати тип корисника на основу улоге (админ, гост, регистрован)!
15. Напиши програм који ће одредити боју сигналног светла на основу унетог броја (1 – црвено, 2 – жуто, 3 – зелено)!
16. Напиши програм који ће израчунати порез на основу категорије прихода (ниска, средња, висока)!
17. У школској лабораторији тестирају се таблети за електронске уџбенике. За сваки таблет мери се проценат напуњености батерије (од 0 до 100). Техничар жели да се аутоматски приказује стање батерије:
од 0 до 20% – „критично“,
од 21 до 60% – „средње“,
од 61 до 100% – „пуно“.
Напиши програм који ће прочитати проценат напуњености батерије и исписати одговарајућу поруку о стању батерије према овим интервалима!

3.27 Циклус while

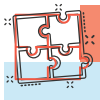
Већ знаш да...

- разликујеш логичке и поредбене изразе,
- користиш структуре избора у C++,
- разумеш шта значи да се направи поређење између две вредности,
- препознајеш када је потребно да се понови неки поступак.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Које су могуће грешке при коришћењу вишеструких услова?

2. Напиши програм који ће према унетом броју дана од 1 до 7 приказати име дана!



У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- разликујеш циклус од других структура у програмирању,
- објашњаваш шта представља циклус и када се користи,
- препознајеш различите типове циклуса у програмском језику,
- разумеш зашто и када се користи понављање наредби,
- пишеш једноставне програме са структурама за понављање,
- анализираш да ли ће се циклус извршити једном, више пута или ниједном.

У многим ситуацијама приликом решавања задатака потребно је поновити одређене активности више пута. Било да је реч о израчунавању збира елемената у низу, приказивању менија или провери лозинке, неопходно је да се неке наредбе изврше више пута. У програмирању се то остварује помоћу такозваних циклуса, који омогућавају да се делови програма извршавају изнова, све док је испуњен одређени услов. Циклуси представљају једну од основних и најважнијих структура за контролу тока програма. Разумевање њихове логике и правилна употреба представљају важан корак у овладавању програмирањем. Они омогућавају писање флексибилнијих, краћих и ефикаснијих програма.

У програмирању структура која се назива циклус представља начин да се исти блок кода изврши више пута, све док је одређени услов истинит. За разлику од секвенцијалног извршавања, при којем се свака наредба извршава једном и задатим редоследом, циклус омогућава да се део програма „врати уназад“ и поново изврши исти код.

Ово понављање може бити унапред одређено (на пример, 10 пута) или условљено – све док се не достигне неко стање. Постоји неколико врста циклуса, од којих сваки има своју намену и карактеристике, али сви имају заједничку идеју: понављање наредби како би се избегло њихово вишеструко ручно исписивање у коду. Структура циклуса while изгледа овако:

```
while (услов) {
// команде које се извршавају
}
```

ПРИМЕР 1

```
#include <iostream>
using namespace std;

int main() {
    int i = 0;
    while (i < 5) {
        cout << "Zdravo!" << endl;
        i++;
    }
    return 0;
}
```

Претпостави да је потребно исписати „Здраво!“ пет пута. Без циклуса било би потребно написати пет cout наредби. Међутим, помоћу циклуса то се може урадити много елегантније и ефикасније. У овом примеру променљива i повећава се при сваком извршавању циклуса. Чим достигне вредност 5, услов i < 5 постаје неистинит и циклус се зауставља. Једна од најважнијих предности коришћења циклуса у програмирању јесте аутоматизација задатака који се понављају. Уместо да програмер ручно копира и понавља код, циклуси омогућавају да се исти блок кода употреби више пута уз минималне измене. То не само да чини програм краћим и лакшим за одржавање већ и смањује могућност настанка грешака. При томе се свака итерација циклуса обично контролише помоћу променљиве која се назива бројач, што омогућава примену додатне логике при сваком понављању. Бројач се може користити за индексирање елемената, сабирање вредности, проверу услова и многе друге функције. Када је добро осмишљена, употреба циклуса значајно побољшава структуру и читљивост кода.

ПРИМЕР 2

Желимо да израчунамо збир првих 10 природних бројева (од 1 до 10).

```
#include <iostream>
using namespace std;

int main() {
    int zbir = 0;
    for (int i = 1; i <= 10; i++) {
        zbir += i;
    }
    cout << "Zbir prvih 10 brojeva je: " << zbir << endl;
    return 0;
}
```

Овде циклус for аутоматски иницијализује бројач i, проверава услов и ажурира његову вредност. У свакој итерацији вредност променљиве i додаје се променљивој zbir. Након завршетка циклуса, променљива zbir садржи укупан збир. Циклуси не омогућавају само понављање већ стварају и услове за динамичко понашање програма. У зависности од корисничког уноса, резултата неког израчунавања или другог спољашњег утицаја, циклус може наставити да се извршава или се може прекинути. Такви циклуси називају се условни циклуси, а најчешће се користе структуре while и do-while. У циклусу while услов се проверава пре сваког извршавања, док циклус do-while најмање једном извршава тело циклуса пре провере услова. Овакви циклуси често се користе када није унапред познат број потребних итерација – на пример, док корисник не унесе исправну вредност или док се не испуне одређени динамички критеријуми. То програму омогућава да флексибилно реагује на различите сценарије без понављања истог кода. Програмери често комбинују циклусе са условима и операцијама над подацима како би изградили сложене алгоритме, од симулација до анализе података. Вештина избора одговарајуће врсте циклуса, постављања исправног услова и избегавања бесконачних циклуса представља један од темеља доброг програмирања. Понекад само мала грешка у услову или ажурирању бројача може довести до тога да се програм никада не заврши. Зато су циклуси алат који захтева дисциплину и пажњу.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта представља циклус у програмирању?

2. Које врсте циклуса познајеш и по чему се разликују?

3. Зашто се у циклусима користе бројачи?

4. Шта ће се догодити ако услов у циклусу while никада не буде испуњен?

5. Напиши програм који ће исписати све парне бројеве од 1 до 50!

6. Напиши програм који ће израчунати збир свих бројева од 1 до N, при чему се N уноси помоћу тастатуре!

7. Напиши програм који ће од корисника тражити да уноси број све док не унесе позитиван број!

8. Напиши програм који ће израчунати производ свих бројева од 1 до 10!

Запамти шта си научио!

- Циклуси омогућавају понављање кода.
- Користе се три основна типа: for, while и do-while.
- Циклуси су ефикасан начин за аутоматизацију.
- Бројачи и услови контролишу број понављања.
- Погрешно постављени циклус може да доведе до бесконечног извршавања.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

У такмичарском програмирању и задацима типа „Дабар“ циклуси се користе за решавање проблема који захтевају анализу или обраду великих серија података. Примери су бројање колико се пута одређено слово понавља у датом тексту или симулација кретања кроз лавиринт.

Комбиновањем циклуса и услова решавају се многи алгоритамски задаци, укључујући сортирање, претраживање и анализу матрица. Што ученик раније савлада логику циклуса, то ће брже моћи да приступи напреднијим концептима, као што су рекурзија, структуре података и оптимизација.

3.28 Задаци за вежбање структуре за понављање while

ЛАКИ ЗАДАЦИ

1. Напиши програм који ће одштампати бројеве од 1 до 15!
2. Напиши програм који ће одштампати бројеве од 10 до 1 у обрнутом редоследу!
3. Напиши програм који ће одштампати све парне бројеве од 20 до 40!
4. Напиши програм који ће одштампати сваки трећи број почињући од 5 до 30!
5. Напиши програм који ће одштампати све бројеве од 1 до 100 дељиви са 10!
6. Напиши програм који ће одштампати фразу „Вежбам while циклуси“ тачно 6 пута!
7. Напиши програм који ће приказати збир бројева од 30 до 40!

СРЕДЊЕ ТЕШКИ ЗАДАЦИ

8. Напиши програм који ће израчунати од 1 до N (унети из тастатуре) који су дељиви са 3!
9. Напиши програм који тражи од корисника да унесе 10 цела броја и прикаже највећи од њих!
10. Напиши програм који ће избројати колико бројева од 1 до 50 је дељиво са 7!
11. Напиши програм који ће израчунати збир свих непарних бројева од 1 до 100!
12. Напиши програм који ће избројати колико цифара има унети број!
13. Напиши програм који ће одштампати збир свих бројева између два унета броја A и B!
14. Напиши програм који ће проверити да ли је унети број дељив и са 2 и са 5!

ТЕШКИ ЗАДАЦИ

15. У математичком клубу ученици уче о „савршеним бројевима“ – бројевима код којих је збир свих позитивних делилаца, не рачунајући сам број, једнак том броју. На пример, за број 6 важи: 1 + 2 + 3 = 6. Напиши програм који ће учитати један цео број и проверити да ли је савршен!
16. Напиши програм који ће пронаћи најмању цифру у унетом броју!
17. Напиши програм који ће израчунати збир квадрата бројева од 1 до N!
18. Напиши програм који ће пребројати колико је бројева од 1 до 100 истовремено дељиво са 3 и са 4!
19. Напиши програм који ће пребројати све палиндромске бројеве од 10 до 999!
20. Приликом провере безбедносног кода дозвољени су само бројеви састављени искључиво од парних цифара (0, 2, 4, 6, 8). Ако се у броју појави барем једна непарна цифра, код је неважећи!

3.29 Циклус са условом – структура do while

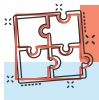
Већ знаш да...

- разликујеш основне типове циклуса у програмирању,
- препознајеш ситуације када треба да се користи понављање инструкција,
- разумеш значење услова у контроли тока програма,
- прочиташ и разумеш једноставни циклус написан у C++.

ЗАДАЧИ И ПИТАЊА ЗА ПОНАВЉАЊЕ:

1. Да ли сваки циклус може да се напише са while?

2. Напиши програм који ће одштампати све бројеве од 100 до 1 обрнутим редоследом!



У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- објашњаваш како функционише циклус do-while,
- препознајеш ситуације у којима је прикладно користити циклус do-while,
- пишеш програм који користи циклус do-while за понављање све док се не испуни услов,
- избегаваш бесконачне циклусе правилним формулисањем услова,
- укључујеш променљиве, услове и ажурирање вредности у циклус.

У програмирању се често јављају ситуације у којима одређени блок кода мора да се изврши најмање једном, без обзира на то да ли је услов истинит. У таквим случајевима користи се циклус do-while, који има специфичну структуру – прво се извршава тело циклуса, а затим се проверава услов. Ако услов остане истинит, циклус се понавља. Овај приступ је користан када програм захтева почетно извршавање, као што су уно-

шење броја од стране корисника, избор опције из менија или покретање неке радње која се мора извршити најмање једном. Структура циклуса do-while изгледа овако:

```
do {
// команде које се извршавају
} while (услов);
```

ПРИМЕР 1

Следећи пример показује како се циклус do-while користи да би се обезбедио унос позитивног броја. Циклус ће се понављати све док корисник не унесе исправну вредност:

```
#include <iostream>
using namespace std;

int main() {
    int broj;
    do {
        cout << "Unesi pozitivan broj: ";
        cin >> broj;
    } while (broj <= 0);

    cout << "Validan broj je: " << broj << endl;
    return 0;
}
```

У овом случају корисник има могућност да унесе број најмање једном. Ако унос није исправан, односно ако број није позитиван, програм ће поново тражити унос све док се не унесе исправна вредност. Овај приступ користи се и за безбедну интеракцију са корисником када је потребно проверити исправност унетих података.

do while циклус најчешће се примењује када се ради о **менаџирању интеракције са корисником**, на пример у менијима, када корисник треба да изабере опцију, а затим да му се да могућност да изабере поново док не изабере „излаз". За разлику од while циклуса, који може да се не изврши ако услов није испуњен од самог почетка, do while је гаранција да ће се логика у телу циклуса извршити најмање једном. Та предност постаје суштинска када први позив према кориснику или према некој функцији **мора** да се изврши, без обзира на логичко стање услова. На тај начин се обезбеђује предвидљиво и контролисано понашање програма.

ПРИМЕР 2

У следећем примеру приказан је мени са три опције. Корисник може да бира све док не унесе опцију 3 – „излаз“:

```
#include <iostream>
using namespace std;

int main() {
    int izbor;
    do {
        cout << "=== Meni ===" << endl;
        cout << "1. Unesi podatke" << endl;
        cout << "2. Prikazi podatke" << endl;
        cout << "3. Izlaz" << endl;
        cout << "Tvoj izbor: ";
        cin >> izbor;

        if (izbor == 1)
            cout << "Podaci se unose..." << endl;
        else if (izbor == 2)
            cout << "Podaci se prikazuju..." << endl;
        else if (izbor != 3)
            cout << "Nepoznata opcija!" << endl;

    } while (izbor != 3);

    cout << "Program je zavrsen." << endl;
    return 0;
}
```

Овај пример приказује типичну примену циклуса do-while у интерактивним програмима. Прво се приказује мени, затим се чека унос корисника, а поступак се понавља све док се не изабере опција за излаз. Ово је уобичајен начин реализације интерактивних конзолних апликација са јасном и безбедном логиком.

Главна карактеристика по којој се циклус do-while издваја јесте редослед извршавања: тело циклуса извршава се пре провере услова, док се код других циклуса услов проверава пре сваке итерације. То значи да се циклус do-while увек извршава најмање једном, без обзира на то да ли је услов истинит од самог почетка. Код циклуса while и for може се догодити да се циклус уопште не изврши ако услов није испуњен од самог почетка.

Поред тога, треба водити рачуна о могућности настанка бесконачног циклуса ако услов никада не постане неистинит. Зато се препоручује да у телу циклуса do-while постоји јасна логика која доводи до прекида, као што је приказано у претходним примерима са уносом, избором или случајним генерисањем. Са становишта читљивости и одржавања кода, циклус do-while треба користити само када постоји стварна потреба за почетним извршавањем.

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Који је главни структурни принцип по којем се циклус do-while разликује од циклуса while?

2. Када је најприкладније употребити циклус do-while уместо циклуса while или for?

3. Да ли је могуће да се циклус do-while ниједном не изврши? Објасни зашто!

4. Зашто се циклус do-while користи у интерактивним менијима?

5. Напиши програм који ће од корисника тражити да унесе број већи од 100.

6. Ако унесе мањи број, програм ће поново тражити унос користећи циклус do-while!

7. Напиши програм који ће симулирати бацање две коцкице све док се на обе не добије исти број!

8. Напиши програм који ће питати корисника да ли жели да настави рад у програму

9. („да“ или „не“), све док не унесе „не“!

10. Напиши програм који ће сабирати бројеве које корисник уноси све док не унесе нулу, а затим приказати њихов збир!

Запамти шта си научио!

- do while циклус се користи када желимо да се блок наредби изврши најмање једном.
- Услов за настављање се проверава по извршавању циклуса.
- Примењује се при уносу података, интерактивних менија, случајним симулацијама, и иницијалним проверама.
- Увек мора да постоји јасна логика која ће довести до прекида циклуса како не би дошло до бесконачног извршавања.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

У неким програмским језицима као што је Python, do while циклус не постоји као уграђена конструкција. У таквим случајевима, његова логика се реализује помоћу while True: и уграђеног услова за прекид помоћу наредбе break. Додатно, напредне симулације, као што су симулације игара на срећу или менаџери дијалога у видео-играма, такође користе логику do while циклуса. У оваквим системима, програмери често комбинују do while са графичким интерфејсима како би омогућили поновљиву интеракцију са корисником.

3.30 Задаци за вежбање структуре за понављање do while

ЛАКИ ЗАДАЦИ

1. Напиши програм који ће исписати све непарне бројеве од 1 до 30!
2. Напиши програм који ће исписати бројеве од 10 до 1 опадајућим редоследом!
3. Напиши програм који ће израчунати збир свих парних бројева од 1 до 100!
4. Напиши програм који ће од корисника тражити да уноси бројеве све док не унесе 0!
5. Напиши програм који ће приказивати поруку „Поново унеси!" све док корисник не унесе број већи од 100!
6. Напиши програм који ће исписати све бројеве дељиве са 5 између 1 и 50!
7. Напиши програм који ће израчунати збир свих бројева дељивих са 3 између 1 и 50!
8. Напиши програм који ће израчунати збир цифара унетог позитивног броја!

СРЕДЊЕ ТЕШКИ ЗАДАЦИ

9. Напиши програм који ће одредити да ли је унети број прост!
10. Напиши програм који ће приказати цифре унетог броја обрнутим редоследом!
11. Напиши програм који ће израчунати производ свих непарних бројева између 1 и 15!
12. Напиши програм који ће пронаћи највећи број у низу бројева које корисник уноси све док не унесе -1!
13. Напиши програм који ће пребројати колико цифара има унети број!
14. Напиши програм који ће тражити унос бројева и израчунавати њихову средњу вредност све док корисник не унесе негативан број!

ТЕШКИ ЗАДАЦИ

15. У једној нумеричкој загонетки тражи се „тајни број" скривен између 100 и 1.000. О њему знамо две ствари: број мора бити дељив са 17, а збир његових цифара мора бити **15.** Напиши програм који ће пронаћи и исписати први број између 100 и 1.000, посматрано растућим редоследом, који је дељив са 17 и има збир цифара једнак 15!
16. Напиши програм који ће израчунати факторијел унетог броја!
17. Напиши програм који ће израчунати збир квадрата првих N природних бројева, при чему корисник уноси вредност N!
18. Ученици уносе низ целих бројева у рачунар како би га анализирали. Уношење бројева завршава се уносом броја 0, који служи само као ознака краја и не сматра се делом низа. Потребно је пронаћи други највећи број међу свим унетим различитим бројевима. Напиши програм који ће уносити целе бројеве један по један, све док се не унесе 0 као ознака краја, а затим одредити и исписати други највећи број међу унетим бројевима!

3.31 Циклус со for структура

Већ знаш да...

- разликујеш основне контролне структуре као услов и циклус,
- препознаш када програм треба да понови одређену акцију,
- користиш do while за најмање једно извршавање,
- следиш логику редоследног извршавања у програму.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Како можеш да избегнеш бесконачан do while циклус?

2. Напиши програм који ће тражити лозинку од корисника све док не унесе тачно одређену лозинку („tajna123“)!

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- користиш for циклус за бројање почетне и крајње вредности,
- објасниш како функционира иницијализација, услов и ажурирање у for-у,
- пишеш програме са фиксним бројевома понављања,
- упоређујеш for, while и do while циклусе,
- решаваш реалне задатке понављањем активности.

У програмирању, потреба за понављањем одређених инструкција јавља се свакодневно. Ова понављања се најчешће појављују у ситуацијама **када се тачно зна колико пута** треба извршити неку радњу. Тада је најпогоднија контролна структура која омогућава бројање понављања. Таква структура се у C++-у (и у многим другим програмским језицима) назива for циклус. Ова структура је компактна, лака за читање и јасно приказује логику почетне вредности, услова за прекид и корака (ажурирања). Управо због ових карактеристика, **for циклус** се често користи при обради низова, табела, уносу података и извођењу прорачуна у

тачно одређеном броју корака. У овој лекцији биће обрађени његов облик, начин извршавања, као и конкретни примери из стварног живота и програмских ситуација.

For циклус у C++-у је конструисан тако да садржи три основна дела: иницијализацију, услов за наставак и ажурирање вредности, који су сви смештени у заградама циклуса. Структура изгледа овако:

```
for (иницијализација; услов; ажурирање) {
// инструкције које се понављају
}
```

У делу „иницијализација" задају се почетне вредности променљивих. Ако је потребно иницијализовати више променљивих, оне се одвајају запетом. Исто важи и за део „ажурирање", у којем се најчешће налазе итератори – ако је потребно извршити више ажурирања, она се такође одвајају запетом. Овај облик омогућава да сви кључни делови циклуса буду обједињени на једном месту, чиме се повећава читљивост кода. Иницијализацијом се обично поставља почетна вредност бројача (на пример, int i = 1;), условом се одређује до када ће се циклус понављати (на пример, i <= 5;), а ажурирањем се мењају вредности при свакој итерацији (на пример, i++).

ПРИМЕР 1

Штампање бројева од 1 до 5

У овом примеру циклус почиње вредношћу i = 1 и понавља се све док је i мање или једнако 5. После сваке итерације вредност i повећава се за 1. На тај начин на екрану се приказују бројеви од 1 до 5. У многим ситуацијама програм захтева бројач који се креће у супротном смеру – од више ка нижој вредности. То може бити потребно при **одбројавању, приказивању елемената обрнутим редоследом или претраживању до испуњења одређеног услова при крају низа.**

Циклус for са обрнутим бројачем дефинише се већом почетном вредношћу, условом према којем се циклус завршава када бројач падне испод одређене вредности и ажурирањем којим се бројач смањује. Овакав начин примене је интуитиван и показује флексибилност циклуса for, који се може прилагодити логици проблема

```
1   #include <iostream>
2   using namespace std;
3
4   int main() {
5       for (int i = 1; i <= 5; i++)
6       {
7           cout << i << endl;
8       }
9       return 0;
10  }
11
```

ПРИМЕР 2

Одбројавање од 10 до 1

Овде циклус почиње вредношћу i = 10 и смањује се наредбом i-- све док је i веће или једнако 1. На екрану ће бити исписани бројеви 10, 9, 8, ..., 1. Овакво одбројавање користи се, на пример, у тајмерима, алармима и интерактивним играма када треба приказати колико је времена преостало.

Поред тога, циклус for може се користити за сабирање, множење или израчунавање збирних вредности. То је нарочито корисно при раду са подацима које је корисник унео или приликом извођења прорачуна са одређеним бројем итерација. Уместо да само исписује вредности, циклус може да обрађује вредност бројача и чува добијени резултат у променљивој. Овај приступ често се примењује у задацима који обухватају израчунавање збира, просечне вредности, факторијела, па чак и проналажење максималне или минималне вредности.

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      for (int i = 10; i >= 1; i--)
6      {
7          cout << i << " ";
8      }
9      return 0;
10 }
11
```

ПРИМЕР 3

Рачунање збира првих 100 природних бројева

У овом примеру збир почиње од 0, а у свакој итерацији додаје му се тренутна вредност променљиве i. Након завршетка циклуса, променљива sum садржаће коначну вредност: 5050. Овакви задаци срећу се и на почетним нивоима програмирања и у математичким моделима.

У практичним програмским решењима циклус for често се користи за приступ елементима низа или серије података. Овај приступ омогућава прецизно и контролисано кретање кроз сваки елемент низа, са тачно одређеним бројем понављања. Структура циклуса for овде је идеална јер користи величину низа за постављање услова за завршетак циклуса и аутоматски ажурира индекс при свакој итерацији. Оваква употреба циклуса од основног је значаја за обраду података, статистику, аутоматизацију прорачуна и многе друге алгоритамске задатке.

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int zbir = 0;
6      for (int i = 1; i <= 100; i++)
7      {
8          zbir += i;
9      }
10     cout << "Zbir je: " << zbir << end
11     return 0;
12 }
13
```

ПИТАЊА И ЗАДАЦИ ЗА ПОНАВЉАЊЕ ЗНАЊА

1. Шта је for циклус и када се користи?

2. Која је разлика између for и while циклуса у односу на синтаксу и примену?

3. Како изгледа структура for циклуса и шта представљају његова три дела?

4. Шта ће се догодити ако се у for циклусу пропусти део за ажурирање?

5. Објасни шта ће исписати следећи програм:

6. for (int i = 1; i <= 5; i++) { cout << i * i << " ";

7. }

8. Напиши програм који ће исписати све парне бројеве од 2 до 20 користећи for циклус!

9. Напиши програм који ће израчунати производ првих 10 природних бројева!

10. Напиши програм који ће исписати збир свих непарних бројева између 1 и 99!

11. Напиш програм који ће приказати абецеду од A до Z користећи for циклус!

12. Напиши програм који ће одбројавати од 50 до 1 и исписати „ГОТОВО!" на крају!

Запамти шта си научио!

- For циклус је структура за понављање познатим бројем итерација.
- Састоји се од иницијализације, услова и ажурирања.
- Идеалан је за бројачке задатке, као бројење, рачунање и обрада низова.
- Може да се користи растућим или опадајућим редоследом.
- Примењује се у многим практичним задацима у програмирању и анализи података.

ЗА ОНЕ КОЈИ ЖЕЛЕ ДА ЗНАЈУ ВИШЕ

- У програмском језику C++ постоје и угнежђени циклуси for, у којима је један циклус уграђен у други. Користе се за обраду матрица и табела.
- иклус for може се комбиновати са условним исказима, као што је if, ради извршавања одређених прорачуна.
- У напредном програмирању користи се итерација помоћу итератора у структурама као што су вектори (vector), мапе (map) и скупови (set).
- Можете користити и циклусе for засноване на опсегу (range-based for) у програмском језику C++11 и новијим верзијама.

3.32 Задаци за вежбање структуре за понављање for

ЛАКИ ЗАДАЦИ

1. Напиши програм који ће исписати све бројеве од 1 до 20!
2. Напиши програм који ће исписати све непарне бројеве од 1 до 50!
3. Напиши програм који ће исписати збир бројева од 1 до 100!
4. Напиши програм који ће исписати бројеве од 10 до 1 обрнутим редоследом!
5. Напиши програм који ће исписати производ бројева од 1 до 5!
6. Напиши програм који ће исписати све бројеве дељиве са 3 од 1 до 30!
7. Напиши програм који ће исписати квадрате свих бројева од 1 до 10!

СРЕДЊЕ ТЕШКИ ЗАДАЦИ

8. Напиши програм који ће исписати збир свих парних бројева од 1 до 100!
9. Напиши програм који ће исписати све бројеве од 1 до N, при чему N уноси корисник, који су дељиви са 5!
10. Напиши програм који ће исписати збир свих бројева од 1 до N који нису дељиви са 2!
11. Напиши програм који ће за сваки број од 1 до 10 исписати његов квадрат и куб!
12. Напиши програм који ће пребројати колико је бројева од 1 до 100 дељиво са 7!
13. Напиши програм који ће израчунати факторијел броја N који уноси корисник!
14. Напиши програм који ће исписати аритметички низ са почетним чланом 3, разликом 5 и укупно 10 чланова!

ТЕШКИ ЗАДАЦИ

15. Напиши програм који ће исписати колико се пута цифра 3 појављује у бројевима од 1 до 100!
16. Напиши програм који ће исписати најмањи и највећи од N бројева унетих помоћу тастатуре!
17. Напиши програм који ће израчунати средњу вредност N унетих бројева!
18. Напиши програм који ће исписати све бројеве од 100 до 999 чији је збир цифара једнак 15!
19. Напиши програм који ће исписати све бројеве од 1 до 100 чија је цифра десетица 2!
20. Напиши програм који ће исписати бројеве од 1 до 100, при чему ће уместо бројева дељивих са 3 исписати „Fizz“, уместо бројева дељивих са 5 „Buzz“, а уместо бројева дељивих и са 3 и са 5 „FizzBuzz“!

ЗАКЉУЧАК

У теми „Програмирање у C++-у“ заокружује се целина у којој се ученицима постављају темељи за даље изучавање информатике и алгоритамског размишљања. На почетку се разматрају логички и алгоритамски задаци, чиме се развија навика да се проблеми анализирају корак по корак и претварају у јасна упутства. Затим се уводе основни информатички концепти и бинарни бројеви, како би се разумело како се подаци представљају и обрађују у рачунару, а увод у кодирање и енкрипцију указује и на важност безбедног руковања информацијама.

Даље се пажња усмерава на појам програмирања, програмске језике и преводиоце, као и на разлике међу језицима. C++ се поставља као главни алат за учење, а истовремено се сагледава и место других језика, као што су Scratch, Java, Python, PHP и Lisp. Кроз интегрисано развојно окружење приказује се како се теоријски концепти претварају у практичан рад: програми се пишу, извршавају и отклањају им се грешке. Кроз псеудокод и секвенцу исказа учи се како се решење најпре описује разумљивим „људским“ језиком, а затим претвара у конкретне C++ наредбе.

У следећем делу уводе се променљиве и константе, њихови типови и правила за додељивање и приказивање вредности, чиме се ствара представа о томе како се подаци чувају у меморији. Аритметичким операцијама и изразима, уносом података, операторима поређења и логичким операторима постављају се математичка и логичка основа програма. На тој основи гради се контрола тока: структура избора између две или више могућности, угнежђени услови и три основна типа циклуса (while, do while и for), што омогућава решавање сложенијих задатака са понављањем и гранањем извршавања.

Бројним задацима за вежбање после сваке целине учвршћује се стечено знање, подстиче самостално размишљање и гради сигурност у коришћењу језика C++. На крају теме ученик треба да разуме логику програма, осмисли алгоритам, запише га у псеудокоду и претвори у исправан C++ код који чита, обрађује и приказује податке. Тако се поставља чврста основа на којој се у наредним темама и годинама могу надограђивати напреднији концепти и технике програмирања.

ПОНАВЉАЊЕ ТЕМЕ 3

1. Објасни својим речима шта је алгоритам и по чему се разликује од обичног описног поступка у свакодневном животу (на пример, рецепта за припрему јела)!
2. Осмисли ситуацију из свакодневног живота (одлазак у школу, припрема за тест, куповина) и запиши је као јасан низ корака! Подвуци који кораци представљају улаз, који обраду, а који излаз!
3. Састави логички задатак са три услова (на пример, о распореду људи, предмета или боја) и напиши најмање два могућа решења!
4. Претвори децимални број 37 у бинарни бројевни систем! Опиши поступак корак по корак!
5. Дат је бинарни број 101101_2. Израчунај његову децималну вредност и објасни како је добијен резултат!
6. Наведи два примера из свакодневног живота у којима се користи кодирање информација, а који нису повезани са рачунарима, и објасни шта се кодира у сваком примеру!
7. Објасни шта је енкрипција (шифровање) и наведи пример у којем би било важно да порука буде шифрована!
8. Објасни својим речима шта значи да је програм детерминистички и зашто је то важно приликом решавања задатака!
9. Напиши три сличности и три разлике између програмског и природног језика (као што су српски или енглески)!
10. Објасни разлику између компајлера и интерпретера! Наведи пример ситуације у којој би ти било важно да знаш који се тип користи!
11. Опиши улогу интегрисаног развојног окружења (IDE) приликом писања програма! Наведи најмање три функције које програмеру олакшавају рад!
12. Напиши алгоритам у псеудојезику који учитава три цела броја и приказује њихову аритметичку средину!
13. Напиши алгоритам у псеудојезику који за унете дужину и ширину правоугаоника израчунава и приказује његову површину и обим!
14. За сваку од следећих променљивих одреди одговарајући тип податка (целобројни, децимални или текстуални) и објасни зашто: узраст ученика, температура ваздуха, цена улазнице и назив града!
15. Запиши у псеудојезику израз који израчунава вредност $3 \cdot x^2 - 2 \cdot x + 5$ за дато x! Објасни како би се овај израз израчунавао корак по корак!
16. Осмисли један пример израза за поређење и један пример логичког израза! Објасни у којим ситуацијама ће њихова вредност бити „истинита“, а у којим „неистинита“!
17. Напиши услов у псеудојезику, користећи структуру if-else, који проверава да ли је ученик положио предмет ако је праг за полагање 50 поена! У зависности од резултата прикажи одговарајућу поруку!
18. Објасни разлику између циклуса while и do-while! Наведи по један пример и опиши када је прикладније користити први, а када други циклус!
19. Напиши алгоритам са циклусом for који у једном реду на екрану приказује све парне бројеве од 1 до 50!
20. Осмисли проблем из свакодневног живота који захтева употребу услова и циклуса (на пример, израчунавање укупне цене са попустом приликом куповине више производа)! Опиши логику решења као низ корака који би се могли претворити у програм!


BOLD
ALPHABET
A
A
CONCE
esc
tab
caps lock
shift
fn
control
option
command

ТЕМА 4: Рад са текстом

Данас се рачунари свакодневно користе за израду и уређивање текстуалних докумената. Најчешће коришћени уређивачи текста су MS Word и Writer из пакета LibreOffice, као и онлајн верзија програма Word и Google Docs. Ови програми су једноставни за употребу и нуде бројне могућности за уређивање текста у различитим форматима и стиловима. У овом уџбенику биће објашњене функционалности прва два уређивача, које су у великој мери сличне функционалностима онлајн уређивача.

Нови појмови

- Наслов, поднаслов, садржај, индекс, чаробњак за обрасце, образац, поље за контролу текста, дугме за избор, падајући мени.

Већ знаш да...

- вршиш промену фонта, боје и стила слова;
- одредиш проред пасуса и растојање између више пасуса;
- пишеш текст у колонама;
- додајеш слике, фотографије, табеле, графике и друге графичке елементе

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

1. Направи плакат за концерт твоје омиљене групе према следећим упутствима:

- Место и време одржавања напиши ћириличним словима и величине 20 тачака;
- Да буду постављени централно и у горњем делу плаката;
- Име групе и њених чланова напиши латиничним словима. Место одреди по сопственом избору тако да привуче више посетилаца;
- Постави одговарајући оквир око назива групе како би се он посебно истакао;
- Назив организатора напиши ћириличним словима и плавом бојом;
- Додај графичке елементе по сопственом избору.

2. Напиши кратак сценарио од десетак редова за видео-игру! Текст напиши словима величине 12 тачака и подеси проред на 1,15. Наслов уреди словима величине 14 тачака и постави га центрирано. Подели текст на два пасуса са међусобним размаком од 6 тачака. Маргине странице подеси на 2 cm. У заглављу напиши назив видео-игре, а у подножју своје име и презиме.

У САДРЖАЈИМА КОЈИ СЛЕДЕ НАУЧИШ ДА:

- креираш и уређујеш текстуални документ различитим стиловима;
- уређујеш садржај у документима према одређеним захтевима;
- креираш индексе у документу;
- користиш шаблоне и формуларе.
- примењујеш заштиту докумената.

4.1 Рад са стиловима

Колико начина уређивања текста знаш?
А Којим наредбама је уређен текст наниже?

„У данашњем **дигиталном свету**, способност за јасну и професионалну **комуникацију** је моћна вештина – **започнимо** изучавањем правила којим ваше идеје постају утицајне ПОРУКЕ.“

Уређивање текстова стиловима у Microsoft Word

4.1.1 Разумевање стилова у текстуалном документу

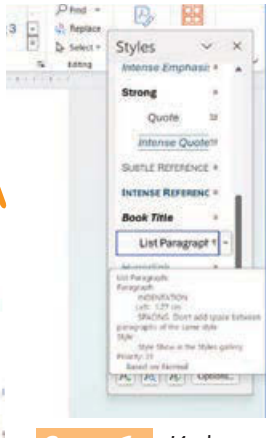
У програмима за обраду текста стил представља скуп подешавања која се примењују на текст, укључујући врсту и величину фонта, боју, поравнање пасуса, размак између редова и друга својства. Употреба стилова омогућава уредан изглед документа, аутоматско креирање садржаја (табеле садржаја), брзо мењање изгледа текста и доследност у целом документу. Истовремено, стилови омогућавају уједначено форматирање докумената, једноставно мењање формата, бољу приступачност

Постоје три основне врсте стилова:

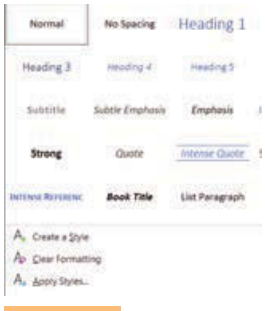
- **Стилови пасуса** – примењују се на целим пасусима (на пр. Heading 1 (Наслов 1), Normal (Нормалан или Основни);
- **Стилови карактера** – примењују се на појединачним речима или фразама (на пр. Emphasis (Наглашено), Strong (Јако наглашено);
- **Табеларни стилови** – примењују се на целим табелама.

4.1.2 Примена постојећих стилова

Microsoft Word нуди широк избор претходно дефинисаних стилова као што су Наслов 1, Наслов 2, Цитат и други. Ови стилови могу да се примене селектирањем текста и избором одговарајућег стила са табулатора Home > Styles.



Слика 1 *Информације О стиловима*



Слика 2 *Врсте стилова*

ПРИМЕР

- **Heading 1** – користи се за главне наслове секција.
- **Heading 2** – за поднаслове, подтеме.
- **Normal** – за обичан текст.
- **Quote** – за истицање цитата или посебног текста.
- **Title и Subtitle** – за главни наслов и поднаслов документа.

Практична примена стилова:

1. Означи пасус који желиш да форматираш!

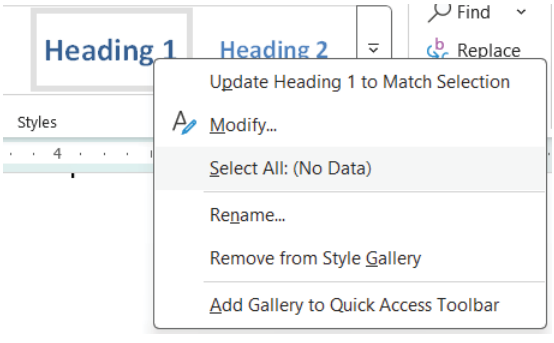
2. Изабери стил из панела Styles!

3. Приметићеш да ће се изабрани Стил аутоматски применити на цео пасус.

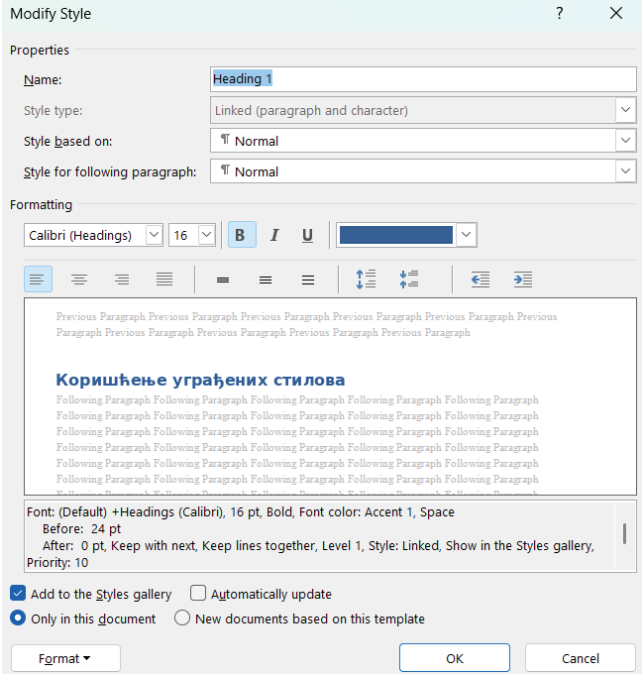
4.1.3 Уређивање постојећих стилова

Када желиш да извршиш мале промене у изгледу стила (на пример, други фонт или боја), није потребно да форматираш сваки пасус ручно.

Довољно је да уредиш сам стил:



Слика 3 *Уређивање постојећег стила*



Слика 4 *Прозор за промену стила*

Кораци за уређивање:

1. Десни клик на стил који желиш да измениш (на пр. Heading 1).

2. Изабери Modify...

3. У прозору ће се појавити, изврши жељене промене: фонт, боја, величина, растојање, поравнање.

4. Потврдом на OK, сви делови који користе тај стил ажурираће се аутоматски.

Ова функција је посебно корисна за документе са више страна где је потребна брза промена дизајна.

4.1.4 Креирање сопствених стилова

Ако документ тражи специфичан изглед који није заступљен у постојећим стиловима, можеш да креираш свој стил.

Кораци:

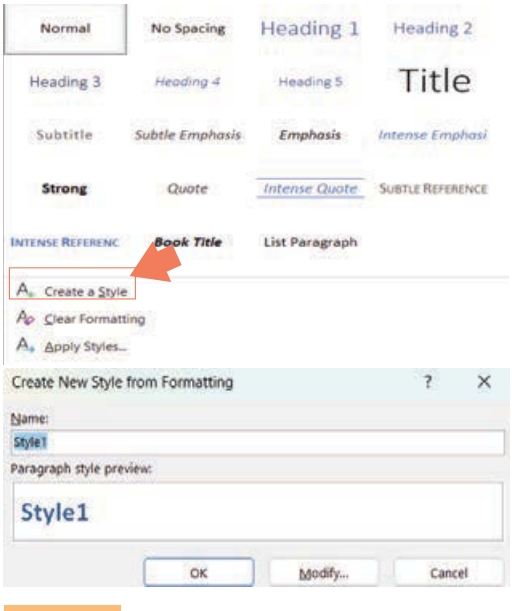
1. Форматирај текст онако како желиш да изгледа (тип фонта, величина, боја, растојање).

2. Означи текст и кликни на Styles > Create a Style > New Style.

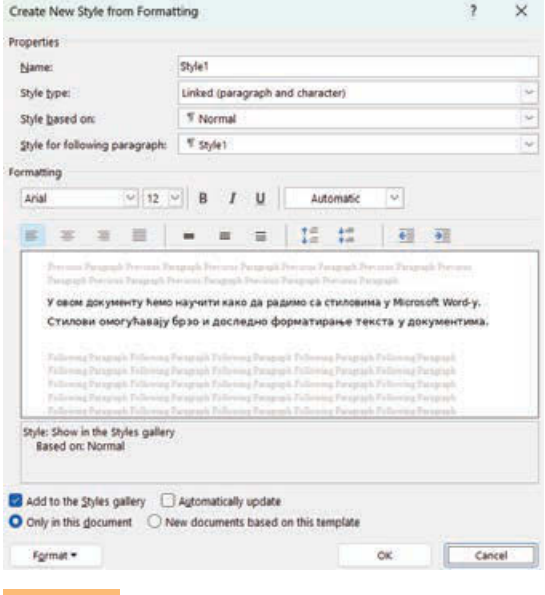
3. Додели име стилу (на пр. „Поднаслов – сиви“).

4. Потврди са OK и стил ће се појавити у листи доступних стилова.

Креирање сопствених стилова је корисно кад радиш на документима са специфичним стандардима или при креирању докумената за штампу, презентацију или објављивање.



Слика 5 *Прозор за креирање новог стила*



Слика 6 *Прозор за креирање новог стила*

У поље Name уписује се назив новог стила, након чега ће се он појавити на постојећој листи стилова.

Стил се касније може изменити у складу са одређеним захтевима кликом на дугме Modify.

- У пољу Style бира се врста стила (за пасус, знакове, табелу или листу);
- У пољу Based on бира се постојећи уграђени формат на којем ће се стил заснивати;
- У пољу Style for following paragraph бира се стил следећег пасуса;
- У пољу Formatting бирају се врста, величина, боја и стил фонта;
- Избором опције Add to Quick Style List стил ће се приказати у галерији и на листи стилова.

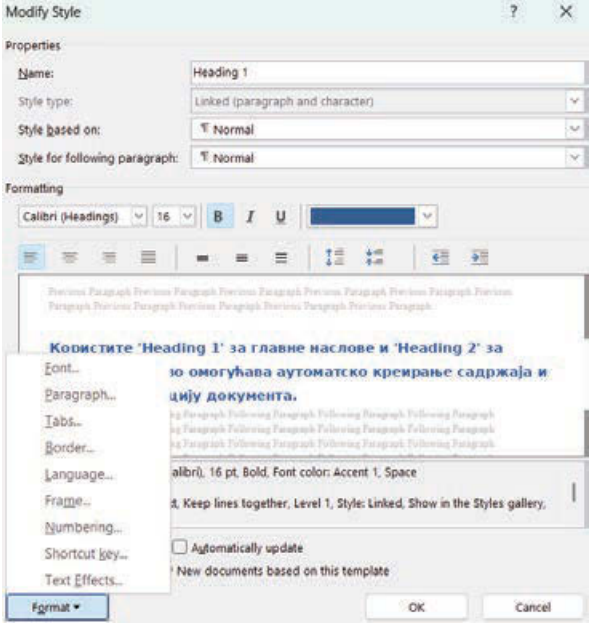
4.1.5 Наслов и поднаслов

У сваком професионално структурираном документу наслов је први елемент који привлачи пажњу. Обично се форматира стилом Title и обухвата:

- фонт велике величине;
- центрирано поравнање;
- посебну боју или ефекат.

Поднаслов подржава наслов са додатним објашњењем или детаљима. Форматира се стилом Subtitle, обично је мање величине, али сличним фонтом и стилом.

Исто тако, можеш да користиш 'Heading 1' за главне наслове и 'Heading 2' за поднаслове. Ово омогућава аутоматско креирање садржаја и бољу организацију документа.



Слика 7 *Прозор за уређивање наслова и поднаслова*

ПРИМЕР

Напиши текст са насловом „Развој дигиталне писмености код ученика“ и поднасловом „Управљање, приступ и алати у ери вештачке интелигенције“. Уреди га применом научених техника за обликовање стила и наслова.

4.1.6 Зашто треба користити стилове?

Коришћење стилова носи више предности:

- аутоматско генерисање садржаја (табела садржаја),
- конзистентност у изгледу,
- једноставно преуређивање,
- професионални изглед,
- боља навигација (посебно при употреби Navigation Panel).

Примена стилова је неопходна посебно у документима типа извештаја, есеја, дипломских радова, образовних приручника или публикација.

Савети за доследно форматирање:

- **Користите стилове уместо ручног форматирања.**
- **Избегавајте мешање различитих врста и величина фонтова.**
- **Користите теме за уједначен изглед документа.**

Примери стилова у различитим документима:

- **Извештаји:** Наслови, поднаслови, текст, цитати.
- **Есеји:** Наслов, увод, главни део, закључак.
- **Писма:** Заглавље, поздрав, садржај, потпис.

4.1.7 ЗАКЉУЧАК

Рад са стиловима у програму Microsoft Word представља важну вештину за свакога ко креира документе са више одељака или документе намењене формалној употреби. Поред примене већ дефинисаних стилова, корисник може да их мења или да креира нове, прилагођене стилове у складу са конкретним потребама. Разумевање појмова наслова и поднаслова, као и њихова правилна примена, додатно обогаћује документ и чини га разумљивијим и визуелно привлачнијим. Употребом стилова документи постају организованији и лакши за читање.

4.1.8 Практична вежба: Рад са стиловима у Microsoft Word

Циљ вежбе:

- да се примењују постојећи стилови;
- да се уреде стилови према потребама;
- да се креирају нови стилови;
- да се правилно користе наслов и поднаслов.

1. Припрема документа

1. Отвори нови Word документ.
2. Унеси следећи текст, без да примењујеш никакво форматирање:

Историја вештачке интелигенције

Вештачката интелигенција (ВИ) је грана компјутерских наука која има за циљ направи системе који могу да размишљају, уче и решавају проблеме као људи.

Подобласт: Машинско учење

Машинско учење је метод анализе података који аутоматизира грађење аналитичких модела.

Подобласт: Дубинско учење

Дубинско учење користи неуронске мреже са више слојева за анализу комплексних података као што су слике и звук.

Закључак

Вештачка интелигенција се развија брзо и има огроман потенцијал да трансформира друштво.

2. Примена постојећих стилова

1. Означи фразу Историја вештачке интелигенције → примени Title стил.
2. Означи речи Подобласт: Машинско учење, Подобласт: Дубинско учење и Закључак → примени Heading 2 стил.
3. За остатак текста у пасусима користи Normal стил.

3. Уређивање постојећег стила

1. Десни клик на Heading 2 → Modify...
2. Промене:

Боја текста: тамноплава, Величина: 14

Додај Bold,
Поравнај лево.
3. Потврди са ОК.

4. Креирање сопственог стила

1. Означи пасус који почиње са „Вештачка интелигенција (ВИ)...“
2. У табулатору Styles, изабери Create a Style → New Style.
3. Име: Увод
4. Изабери:
   - Фонт: Georgia, 12 тачака
   - Италик
   - Простор између редова: 1,5
   - Justify поравнање
5. Потврди са ОК и примени стил.

5. Додатни задатак (по избору)

Додај аутоматску **табелу садржаја** на почетку документа преко: References > Table of Contents > Automatic Table 1

ПРАКТИЧНИ ЗАДАЧИ:

1. Отвори нови Word документ и напиши кратак текст са три наслова и неколико параграфа. Примени стилове „Heading 1“, „Heading 2“ и „Normal“!

2. Креирај нови стил са именом „Мој стил“ који користи плави фонт, величину 14 pt и поравнање по средини! Примени на неком параграфу!

3. Уреди стил „Heading 1“, тако што ћеш користити црвену боју и Bold фонт! Примени га на свим главним насловима у документу!

4. Генериши аутоматски садржај базиран на стиловима које си применио!

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Која картица у Microsoft Word-у садржи панел са стиловима?

а) Insert.

б) Home.

в) Layout.

г) Review.

2. Шта се дешава када примениш стил „Heading 1“ на текст?

а) Текст се претвара у табелу.

б) Примењује се форматирање за главни наслов.

в) Текст се брише.

г) Додаје се слика.

3. Који је исправан начин за уређивање постојећег стила?

а) Двоструки клик на стил.

б) Десни клик > Modify.

в) Клик на Insert > Style.

г) Клик на File > Options.

4. Шта омогућава коришћење стилова у документу?

а) Само форматирање.

б) Доследан изглед и аутоматски садржај.

в) Више грешака.

г) Само промену боје.

5. Објасни шта је стил у Microsoft Word-у и зашто је користан!

6. Како можеш да креираш сопствени стил у Word-у?

7. Које су предности коришћења стилова за наслове и поднаслове?

8. Шта се дешава ако промениш стил који је већ примењен на више места у документу?

4.2 Садржај и индекси

Када радимо на дужим документима, као што су реферати, есеји, пројекти и књиге, веома је важно да они буду прегледни и да омогућавају лако сналажење. У ту сврху Microsoft Word нуди алатке за креирање садржаја (табеле садржаја) и индекса, који се аутоматски генеришу и ажурирају на основу стилова и ознака у тексту.

4.2.1 Шта је садржај (табела садржаја)?

Садржај је списак свих главних делова и поднаслова у документу и обично се поставља на његов почетак. Он служи као водич читаоцу и омогућава брзо прелазак на жељени део документа.

Како се креира садржај?

Да би се креирао садржај, потребно је најпре применити стилове наслова („Heading 1“, „Heading 2“ итд.) на све наслове и поднаслове у документу.

Следећим корацима можеш да креираш садржај:

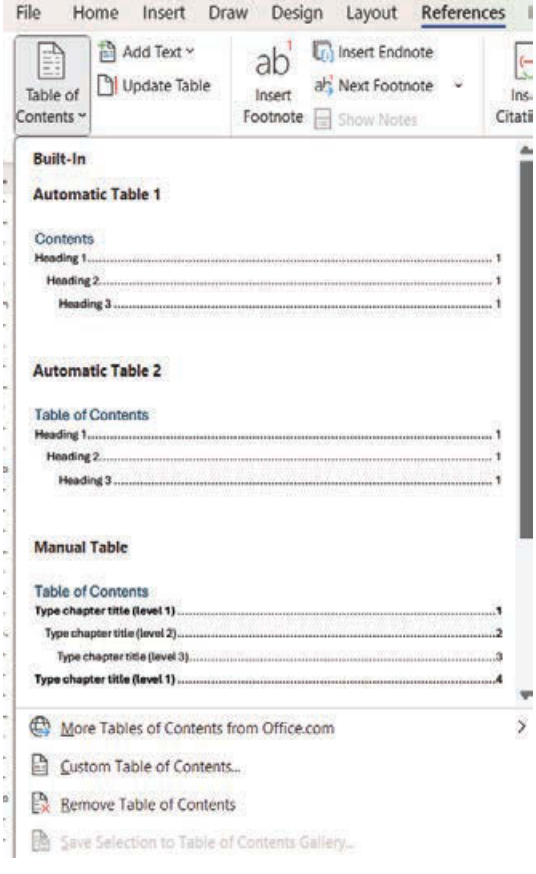
1. Постави курсер тамо где желиш да се прикаже садржај (на пр. на почетку документа).

2. Са траке алата изабери: References (Референце) → Table of Contents (Садржај).

3. Изабери стил садржаја са понуђене листе (Automatic Table 1 или 2).

4. Word аутоматски генерише садржај свих елемената форматираних стиловима наслова.

Генерисани садржај ће садржати бројеве страна и хипервеза са којима може једноставно да се навигује кроз документ до тражених наслова.



Слика 7 *Креирање садржаја у документу*

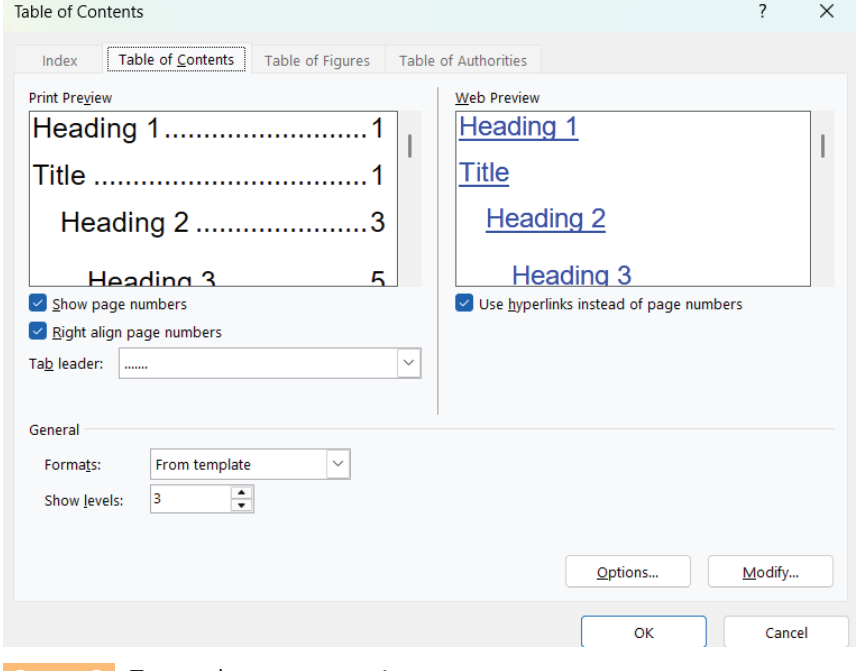
Савет:

Доследно користи стилове наслова у целом документу како би уређивање садржаја било једноставније. Странице такође треба да буду нумерисане. При генерисању садржаја програм проналази наслове и додаје их у табелу садржаја са одговарајућим бројем странице.

4.2.2 Прилагођавање садржаја

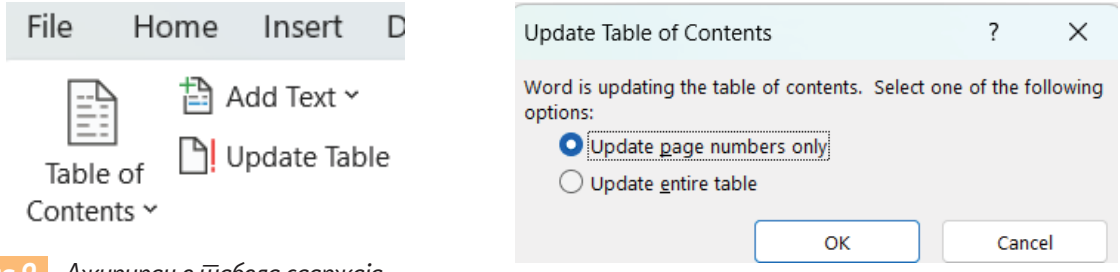
Садржај се може прилагодити потребама на следеће начине:

- укључивањем више нивоа поднаслова (нпр. Heading 3);
- изглед се може променити поновним избором картице References (Референце) → Table of Contents (Садржај), а затим опције Custom Table of Contents, након чега се појављује прозор као на слици 8;
- опцијом Show page numbers одређује се да ли ће бити приказани бројеви страница;
- опцијом Right align page numbers одређује се да ли ће бројеви страница бити поравнати са десне стране;
- у листи Tab leader бира се стил линије водиље до бројева страница (празан простор, линија, тачке или цртице);
- у пољу Formats бира се један од формата табеле садржаја;
- у пољу Show levels одређује се до ког нивоа ће наслови бити приказани;
- ако ће документ бити објављен на веб-локацији, елементе садржаја треба уредити као хипервезе. У ту сврху означава се опција Use hyperlinks instead of page numbers;
- на крају се изабрана подешавања потврђују кликом на дугме OK.



Слика 8 *Прилагођавање садржаја у документу*

Табела садржаја може да се ажурира и при промени у тексту опцијом Update Table. Бира се Update page numbers only уколико се ажурира само број страна или Update entire table којом се врши потпуна промена.



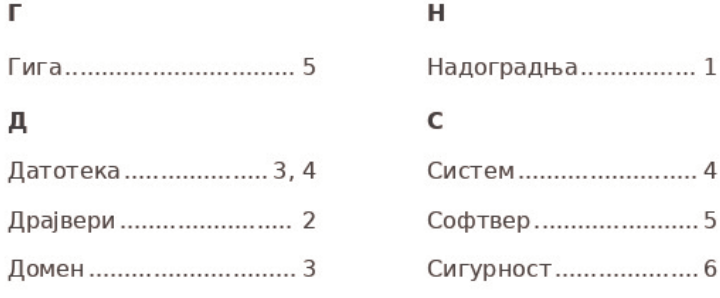
Слика 9 *Ажурирање табела садржаја*

Добро организовани документ треба да има:

- Јасну структуру: наслов → поднаслов → текст.
- Хијерархију информација: главна тема (Heading 1), подтеме (Heading 2), под-под-теме (Heading 3).
- Употреба празних простора, набрајање и слике када је потребно.

4.2.3 Шта је индекс?

Индекс је списак кључних речи или појмова из документа, приказаних на крају заједно са странама где се појављују. Уобичајено се користи у књигама, енциклопедијама или научним радовима.



Слика 10 *Индекс*

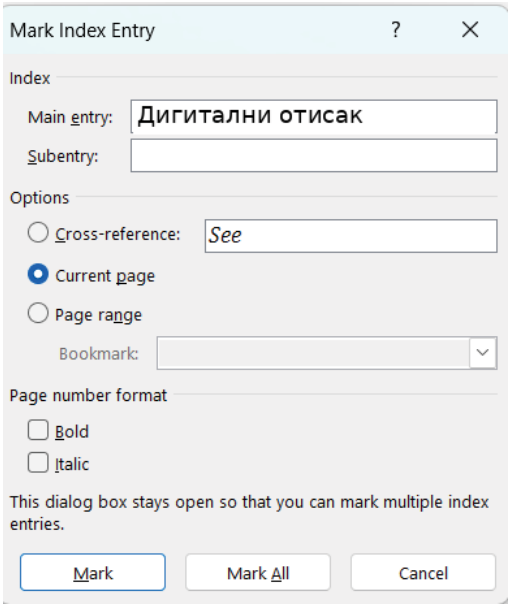
Како се додаје списак или табела са индексом?

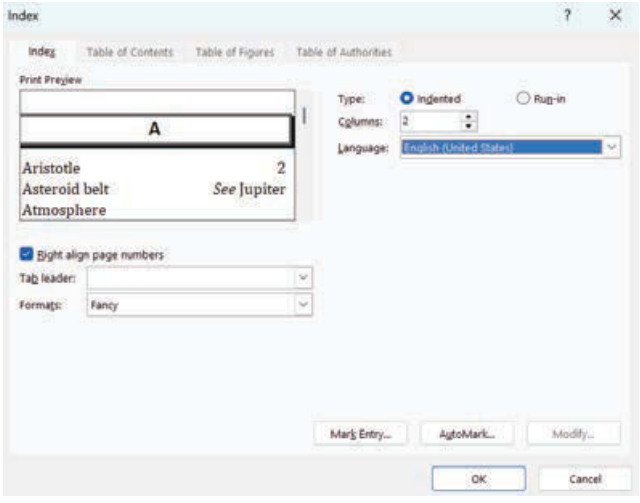
- Означи реч или појам који треба да буде део индекса;
- Изабери картицу References, а затим опцију Mark Entry;
- У пољу Main entry појавиће се означена реч;
- Кликни на Mark или Mark All ако се реч појављује више пута. MS Word ће у документ додати посебно поље XE (елемент индекса). Ово поље се неће приказивати приликом штампања документа, као ни ознака пасуса ¶;

```
Дигитални·отисак{·XE·"Дигитални·отисак"·}¶
```

- Затим постави курсор на крај документа.
- Изабери References → Insert Index.
- Изабери формат у којем ће се приказивати, тип вођице, број колона и потврди избор кликом на OK.

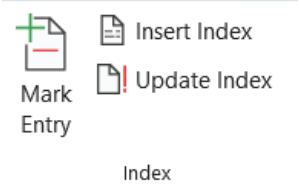
Индекс ће се приказати азбучно, бројевима на странама за сваки термин.





Слика 11 *Креирање индекса*

Уколико се додају нове речи индекс може да се ажурира избором опције Update Index.



Да би се уклонила ознака неке речи или појма, потребно је означити поље XE заједно са заградама {} и речју у њима, а затим притиснути тастер Delete.

4.2.4 Практична активност

Задатак: 1

1. Напиши кратак текст са три наслова и поднаслова.

2. Примени стилове (Heading 1 и 2).

3. Додај табелу садржаја на почетку.

4. Означи три речи као индекс и додај индекс на крају.

5. Направи измену у документу и ажурирај и садржај и индекс.

Задатак: 2

1. Креирај нови документ у Microsoft Word са насловом: „Дигитална писменост и коришћење ИКТ у образовању“

2Структуриши документ са следећим деловима (секцијама):

↪ Наслов документа:

- „Дигитална писменост и коришћење ИКТ у образовању“ → Heading 1
- Увод → Heading 1

- Шта представља дигиталну писменост? → Heading 2
- ИКТ-алати који се користе у школама → Heading 2
- Предности и изазови при коришћењу технологија у настави → Heading 2
- Закључак → Heading 1

Сваки део треба да садржи четири до пет реченица текста.

1. Примени одговарајуће стилове на сваком наслову (Heading 1, Heading 2).
2. Додај аутоматску табелу садржаја на почетку документа:
   - References → Table of Contents → Automatic Table.
3. Означи за индекс следеће речи:
   - технологија, ученик, дигитална писменост
   - За сваку реч: References → Mark Entry → Mark All
4. Додај индекс на крају документа:
   - References → Insert Index
5. Сачувај документ под именом: „Индексиран_документ_ТвојеИме.docx“

Савет

Кад додаш нови наслов или поднаслов у тексту, не заборави да ажурираш табелу садржаја: Клик на садржај → Update Table → Update entiretable

ЗАКЉУЧАК

Рад са садржајем и индексима у програму Microsoft Word омогућава бољу организацију, прегледност и професионални изглед текстуалних докумената. Употребом стилова наслова можемо брзо да креирамо аутоматску табелу садржаја, која се ажурира у складу са изменама у документу. Индекс, с друге стране, помаже читаоцима да лако пронађу кључне речи и теме. Ове алатке омогућавају ефикаснији рад са текстуалним документима у свакодневном пословању

ПРАШАЊА ЗА ПРОВЕРКА НА ЗНАЕЊЕТО

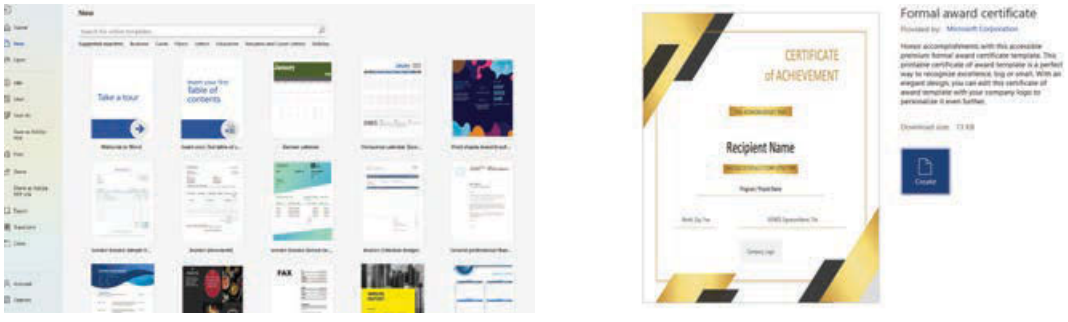
1. Како се креира табела садржаја у програму Word?
2. Зашто је важно користити стилове наслова?
3. Како се реч означава за индекс?
4. Када и како се ажурирају табела садржаја и списак индекса?
5. Шта треба да урадиш пре него што креираш индекс?
6. У ком делу документа најчешће се налази списак индекса?

4.3 Шаблони и формулари

Када радимо са документима који се често понављају — као што су пријаве, захтеви, уговори или извештаји — можемо користити шаблоне. Шаблони (templates) су унапред припремљени документи са одређеним форматирањем, структуром и садржајем који се може прилагодити. Поред тога, Word омогућава креирање формулара (образаца) који се користе за унос података на стандардизован начин.

4.3.1 Шаблон

Приликом отварања новог документа MS Word нуди велики број категорија готових шаблона, на пример за писма, позивнице, сертификате, насловне странице семинарских радова и друго. Шаблон се бира помоћу опција File → New, након чега се са десне стране приказује збирка категорија. Помоћу опције Search могу се претраживати шаблони на интернету. Кликом на одређену категорију приказује се већи број шаблона. Потребно је изабрати жељени шаблон и кликнути на дугме Create. Отвориће се нови документ са унапред дефинисаним стиловима, који се може уређивати према сопственим потребама или одређеним захтевима. Шаблон има другачију екстензију од обичног документа – .dotx – и чува се у посебној фасцикли Documents → Custom Office Templates.



Слика 12 *Креирање шаблона (MS Word)*

Кораци: како се користи шаблон?

1. Отвори Word → File → New.
2. У прозору изабери шаблон за агенду радионице прве помоћи.
3. Кликни Create и прилагоди документ према својим потребама.
4. Кад завршиш са уређивањем и писањем, сачувај документ под именом „Агенда за радионицу“.

Чекори: Како се креира сопствени шаблон?

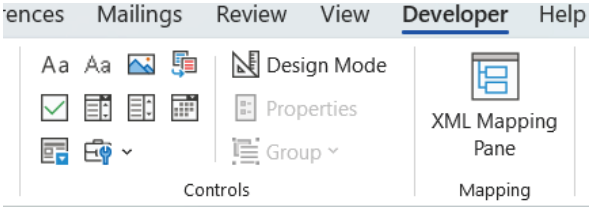
1. Направи нови документ.
2. Додај текст, заглавље, форматирање и елементе који се понављају.

3. Изабери: File → Save As.
4. Под Save as type изабери: Word Template (.dotx).
5. Додели име (на пр. „Мој шаблон за извештај“) и сачувај га.
6. Нови документ аутоматски се чува у директоријуму (фасцикли) Custom Office Templates.

4.3.2 Формулар

Формулар је документ који садржи поља за унос података, као што су имена, датуми, адресе и одговори на питања. Користи се за прикупљање података креирањем упитника, анкета, пријава и слично. Садржи унапред дефинисана поља, као што су поље за унос текста, дугме за избор, поља за потврду и падајући менији у које се могу уносити подаци. Такође садржи и текст који се не може мењати, на пример питања из анкете или захтеве у пријави („Унеси датум рођења“, „Име и презиме“, „Омиљена музичка група“ итд.).

Формулар се креира отварањем новог документа и избором картице Developer. Затим се у групи Controls бирају поља која формулар треба да садржи. Ако картица Developer није приказана на траци са картицама, можеш је додати избором File → Options, а затим Customize Ribbon. У прозору са десне стране означи Developer и кликни на OK.



Слика 13 *Креирање поља у формулару (MS Word)*

1. Поље за контролу текста (Text Control Field)

Ово поље омогућава корисницима да унесу текст у формулар. На пример, име или адреса уносе се у контролно поље за текст. Оно се додаје избором Developer → Controls → Rich Text Content Control или кликом на дугме Aa.

2. Дугме за избор (Option Button или Radio Button)

Ово је дугме које се користи када треба да се изабере само једна од више опција. Често се користи у упитницима (на пример: пол: → мушки → женски). Додаје се кликом на дугме.

3. Падајући мени (Drop-down list)

Падајући мени је листа избора која се отвара кад се кликне на стрелицу. Корисник може да изабере једну од унапред дефинисаних опција (на пример: градови, занимање, одељење). Додаје се кликом на дугме.

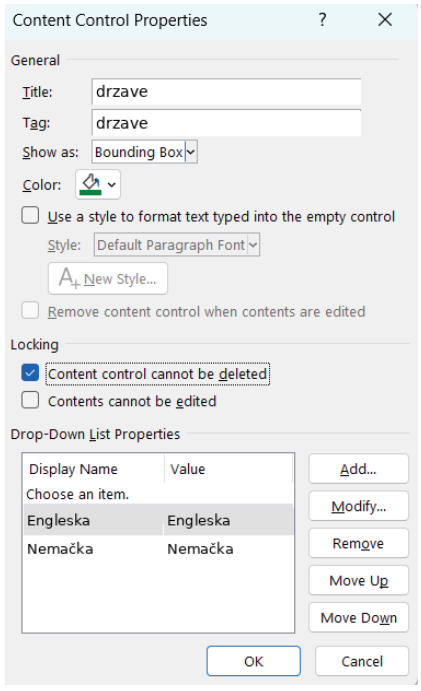
4. Поље за селектирање (Check Box)

Користи се за избор од више опција (на пр. интереси: читање, спорт, музика). Додаје се кликом на дугме.

5. Падајући мени (Drop-down List)

Падајући мени са унапред дефинисаним опцијама и корисницима бира најбољу понуду. Додаје се кликом на дугме.

Односно **Drop-Down List Content Control**, затим изабери **Properties** да додаш опције са кликом на дугме Add (слика 13). Можеш да додаш и име падајућег менија. Унете опције можеш да мењаш помоћу дугмета Modify, да их уклониш са Remove и да мењаш редослед са Move up и Move down. Селектирај опцију Content control cannot be deleted да корисник не би избрисао поље. Уколико је селектирана опција Content control cannot be edited, корисник не може да изабере ништа.



Слика 14 *Додавање опција у формулар (MS Word)*

6. 7. Поље за избор датума

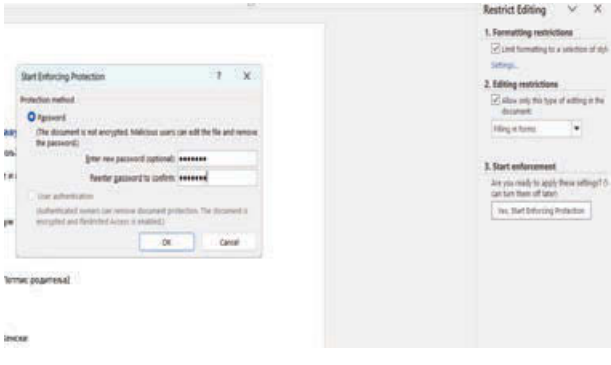
У њега се уносе датуми избора, на пример, одржавање састанка, избор дана за екскурзију, итд.

Закључавање формулара

Када дизајнираш формулар: изабери Developer > Restrict Editing

У делу „Editing restrictions“, изабери Filling in forms, чиме ће корисници само моћи да пишу одговоре и затим кликни Yes, Start Enforcing Protection да поставиш лозинку. Ово ће омогућити корисницима да попуњавају формулар, али да не мењају дизајн.

Слика 15 *Закључивање формулара без лозинке (MS Word)*



Слика 16 *Закључивање формулара са лозинком (MS Word)*

Чување као шаблон

Изабери File > Save As затим изабери тип документа Word Template (*.dotx) и кликни Save. Ово ће ти омогућити да више пута користиш формулар.

ЗАКЉУЧАК

Употреба шаблона и формулара у програму Word олакшава креирање стандардизованих докумената. Шаблони штеде време и обезбеђују доследан изглед, док формулари пружају структуру за прецизан и уредан унос информација. Ученик који савлада ове алатке биће оспособљен да израђује документе на напредном нивоу, прилагођене стварним образовним и административним ситуацијама.

ПРАКТИЧАН ЗАДАТАК

Креирање обрасца за пријаву ученика у школску активност

1. Отвори нови документ у Word.
2. Унеси заглавље: „Образац за пријаву за школску екскурзију.“
3. Укључи следећа поља:
   - Име и презиме → Text Field
   - Одељење → Drop-down list (од I-1 до IV-5)
   - Пол → Option Buttons (Мушки / Женски)
   - Датум рођења → Text Field
   - Потпис родитеља → Text Field
4. Активирај заштиту са Restrict Editing, тако да само поља могу да се попуњавају.
5. Сачувај документ као шаблон: Save as → Word Template (.dotx)

ПРАКТИЧАН ЗАДАТАК

1. Отвори нови документ у Microsoft Word.
2. Активирај траку „Developer“ преко File > Options > Customize Ribbon и означи „Developer“.
3. Креирај формулар за пријаву догађаја са следећим елементима: Формулар треба да садржи:
   - Текстуална поља за: Име, Презиме, Е-пошту
   - Падајући мени за избор града (на пр. Скопље, Битољ, Тетово)
   - Дугме за избор (Radio Buttons) за пол: Мушки / Женски
   - Поље за коментари (Text Area)
   - Поље за избор (Check Box) за интересе: Технологија, Уметност, Спорт

ПРАШАЊА ЗА ПРОВЕРКА НА ЗНАЕЊЕТО

1. Шта је шаблон и за шта се користи?

2. По чему се формулар разликује од обичног документа?

3. Где се налазе контроле за текст и падајући мени?

4. Како се чува свој шаблон?

5. У чему је разлика између дугмета за избор и падајућег менија?

6. Где се налазе алати за креирање формулара у Word?

а) File.

б) Developer. в) Insert.

г) View.

7. Које поље се користи за унос текста корисника?

а) Drop-down.

б) Text Control. в) Check Box. г) Table.

8. Које дугме се користи када треба да се изабере само једна опција?

а) Check Box.

б) Drop-down.

в) Radio Button.

г) Text Field.

9. Шта омогућава падајући мени? а) Унос слободног текста.

б) Избор понуђених опција.

в) Уметање слика.

г) Штампање.

10. Да ли можемо да закључимо формулар да може да се попуњава, али не и да се уређује?

а) Не.

б) Да.

11. Објасни улогу траке Developer у Word!

12. Који типови контроле могу да се користе у једном формулару?

13. Опиши како се закључава формулар за попуњавање!

4.4 Заштита докумената

Заштита Word докумената важна је за спречавање неовлашћеног читања, уређивања или брисања садржаја. Microsoft Word нуди неколико начина за заштиту докумената.

Врсте заштите

1. Само за читање (Read-only):

Овај режим омогућава да се документ отвори, али не и да се уређује без дозволе власника. Корисник ће добити обавештење да је документ само за читање.

2. Заштита помоћу лозинке (Password protection):

Може да се постави лозинка за отварање или уређивање документа. Без тачне лозинке, документ не може да се отвори или да се измени.

3. Ограничавање уређивања (Editing restrictions):

Омогућава да се дозволи само одређени тип уређивања, као попуњавање формулара, коментара или праћење промена.

Кораци за постављање заштите

1. Отвори документ у Microsoft Word.

2. Иди на картицу 'File' > 'Info'.

3. Кликни на 'Protect Document'.

4. Изабери једну опцију:
   - 'Always Open Read-Only'
   - 'Encrypt with Password'
   - 'Restrict Editing'

5. Прати инструкције за постављање лозинке или ограничења.

6. Сачувај документ.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

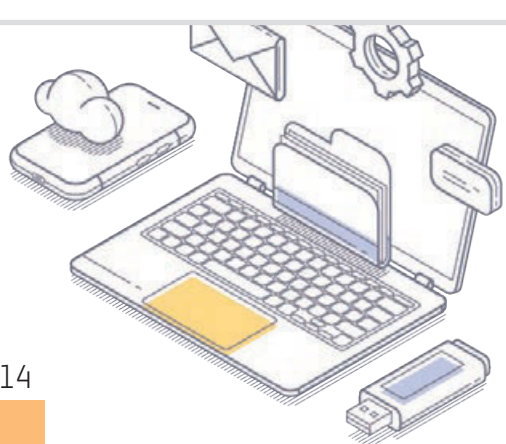
1. **Да ли може да се постави лозинка за отварање Word документа? (Да/Не)**
2. **Да ли режим ‘Read-only’ дозвољава уређивање? (Да/Не)**
3. **Да ли може да се дозволи само попуњавање формулара у документу? (Да/Не)**
4. **Објасни како се поставља лозинка за документ у Word!**
5. **Које су предности ограничавања уређивања?**
6. **У којим ситуацијама би користио режим ‘Read-only’?**

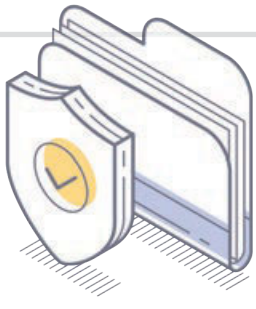
ПРАКТИЧАН ЗАДАТАК

1. **Креирај документ под насловом ‹Мој пројекат›. Напиши кратак текст. Затим постави заштиту на документ тако што ће:**
2. **бити само за читање.**
3. **имати лозинку за отварање.**
4. **дозвољавати само попуњавање формулара.**
5. **сачувај документ и покушај да га отвориш без лозинке.**

ЗАКЉУЧАК

Употребом алатки за заштиту у програму Word можемо очувати поверљивост и интегритет својих докумената. Важно је знати како се постављају различите врсте заштите и како се примењују у практичним ситуацијама.





Шта сам научио

Рад са документима у програму Microsoft Word не своди се само на писање текста – он обухвата вештине уређивања, организације, аутоматизације и заштите садржаја. Током ове теме ученици су се упознали са неколико кључних алатки и концепата.

Стилови

Употребом стилова документи добијају професионалан изглед и уједначеност. Стилови омогућавају брзо форматирање наслова, поднаслова и текста, као и аутоматско креирање садржаја.

Садржај и индекси

Аутоматски садржај (Table of Contents) и индекси корисни су за навигацију и организацију већих докумената. Они се заснивају на стиловима и омогућавају лако претраживање и већу прегледност.

Шаблони

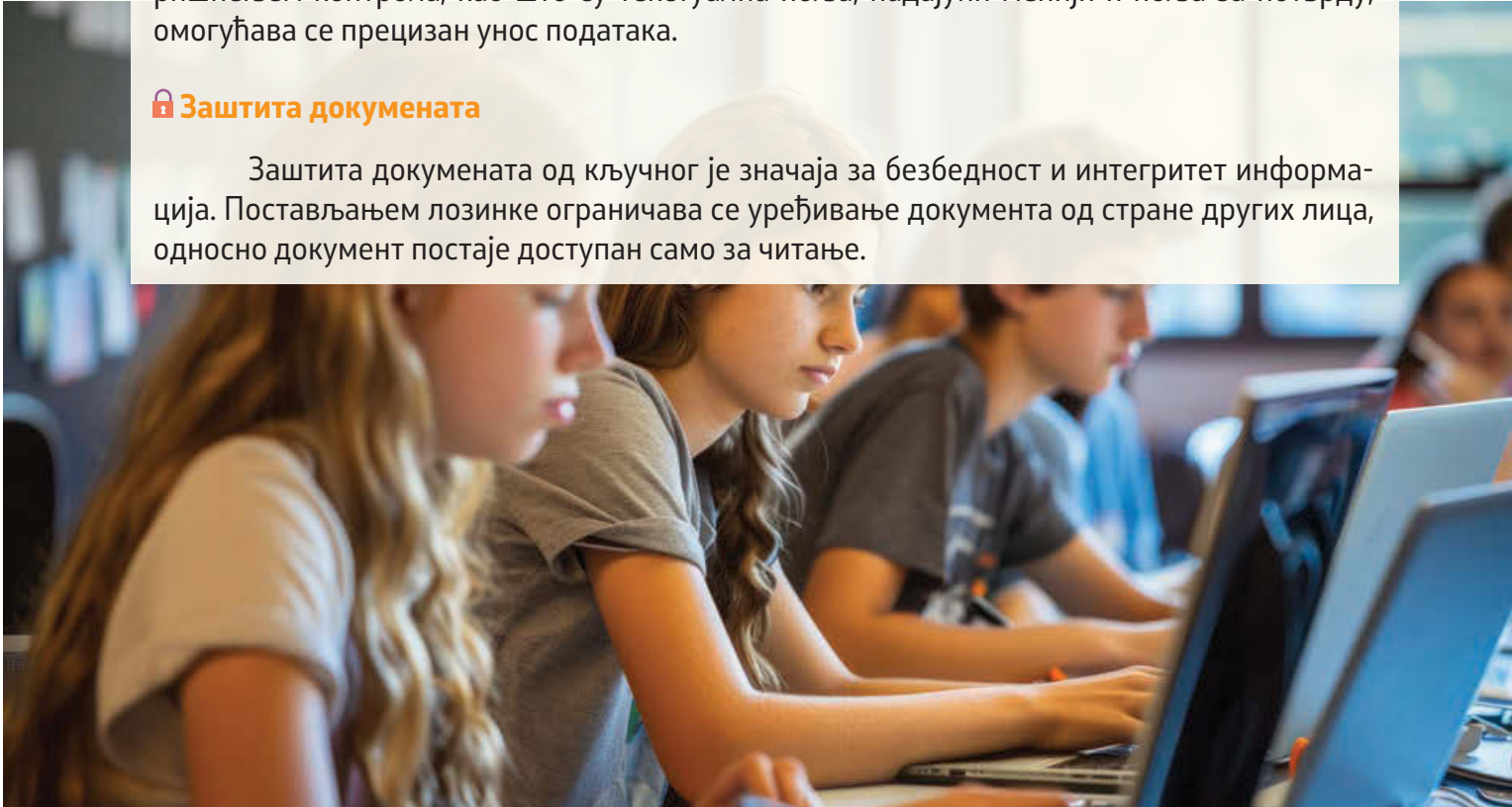
Шаблони (templates) су унапред дефинисани формати који штеде време и обезбеђују уједначен изглед докумената. Користе се за различите намене – извештаје, писма, формуларе и друго

Формулари

Формулари су интерактивни документи који садрже поља за попуњавање. Коришћењем контрола, као што су текстуална поља, падајући менији и поља за потврду, омогућава се прецизан унос података.

Заштита докумената

Заштита докумената од кључног је значаја за безбедност и интегритет информација. Постављањем лозинке ограничава се уређивање документа од стране других лица, односно документ постаје доступан само за читање.



ПОНАВЉАЊЕ ЗА ТЕМА 4

1. Шта представља „стил“ у текстуалном документу? Наведи најмање два примера (нпр. Heading 1, Normal)!

2. Објасни зашто је корисно примењивати стилове уместо ручног форматирања текста!

3. Како би уредио документ ако наставник захтева да сви наслови буду обликовани истим стилом? Опиши поступак!

4. У документу имаш листу појмова. Како би креирао индекс који води до сваке кључне речи?

5. Упореди могућности шаблона (template) и формулара (form). По чему су слични, а по чему се разликују?

6. Анализирај следећу ситуацију: у једном документу употребљени су различити фонтови, боје текста и величине слова. Како би организовао стилове да би документ изгледао уредно?

7. Процени да ли је безбедно послати документ електронском поштом без заштите ако садржи осетљиве информације! Објасни своју одлуку!

8. Процени који је начин заштите делотворнији у конкретном случају: лозинка, ограничење уређивања или режим само за читање!

9. Направи кратак документ користећи три различита стила – за наслов, поднаслов и текст –и објасни зашто си изабрао те стилове!

10. Направи сопствени шаблон за школски извештај! Укључи место за наслов, име аутора, датум и индекс! Опиши како би га користио!

ТЕМА 5: Програм за табеларна израчунавања

Програми за табеларна израчунавања представљају моћне дигиталне алатке које омогућавају систематско организовање, анализу и визуелизацију података. Захваљујући својој флексибилности и функционалности, имају широку примену у различитим областима свакодневног и професионалног живота — укључујући образовање, пословање, научна истраживања и администрацију. Омогућавају једноставну обраду информација, креирање формула и аутоматизована израчунавања, као и визуелно приказивање података помоћу графикона и табела. Поред математичких операција, подржавају и напредне функције, као што су сортирање, филтрирање, условно форматирање и анализа података, чиме значајно доприносе ефикаснијем раду и доношењу одлука заснованих на информацијама. Најчешће коришћени програми за табеларна израчунавања су Microsoft Excel, Google Sheets и LibreOffice Calc. У овом уџбенику биће објашњене функционалности програма Microsoft Excel, који је део пакета Microsoft 365.

Нови појмови

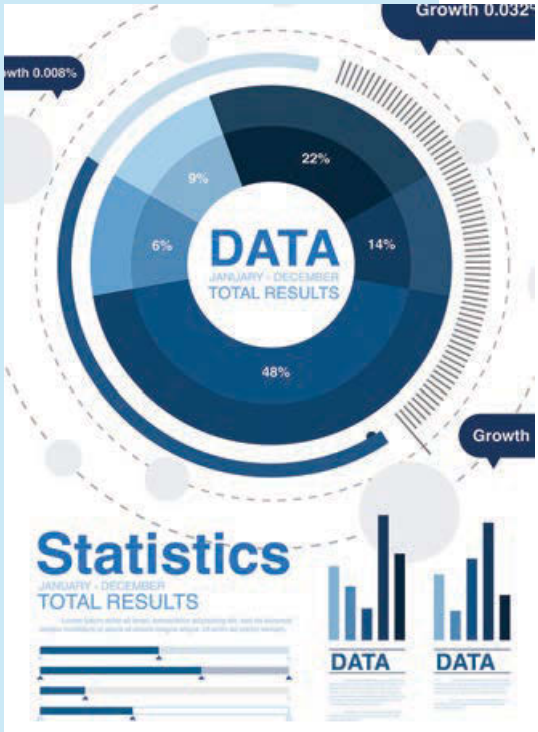
- табела, редови, колоне, ћелије, радни лист, додавање, брисање, преименовање, премештање и копирање радног листа, формула, функција, графикон, врсте графикона, сортирање података, филтрирање података, пивот табела, заштита ћелија, заштита радне свеске.

Већ знаш да...

- ✓ уносиш податке различитог формата;
- ✓ креираш и уређујеш једноставне табеле;
- ✓ креираш једноставне формуле за табеларна израчунавања користећи траку за формуле;
- ✓ користиш једноставне функције за израчунавање података у табели преко одговарајућег менија;
- ✓ креираш графиконе различитог типа;
- ✓ филтрираш податке у табели;
- ✓ сортираш податке у табели.

УНОСИШ ПОДАТКЕ РАЗЛИЧИТОГ ФОРМАТА;

- креираш и уређујеш табелу у програму за табеларна израчунавања;
- креираш базу података;
- примењујеш формуле и функције на напредном нивоу за израчунавање података у табели;
- бираш и уређујеш разне типове графикона, према датим критеријумима;
- ређаш (сортираш) и филтрираш податке из табеле, према датим критеријумима;
- користиш условно форматирање у креирању интерактивне табеле;
- користиш апсолутно и релативно адресирање;
- уређујеш пивот табеле;
- примењујеш заштиту података у табели.



ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

Замисли да си део тима који организује музички фестивал. Биће бендова, добре музике, укусне хране и много забаве. Твој задатак је да припремиш табелу у програму Excel и израчунаш колико ће новца бити потребно за све припреме – ангажовање бендова, озвучење, рекламирање, храну за волонтере и све остало што је неопходно да би фестивал био одлично организован!

5.1 Табела као база података

База података је организована збирка података којима се лако приступа и који се могу обрађивати, ажурирати и анализирати. Уместо да се подаци чувају на различитим местима или у несређеним списковима, база омогућава систематско складиштење информација, чиме се повећавају ефикасност и сигурност њихове употребе. Базе података користе се у готово свим областима савремене технологије — од малих личних апликација до сложених система у банкарству, здравству, образовању и електронској трговини. Основне карактеристике базе података су: организованост, структура, повезаност података, могућност претраживања, заштита и очување интегритета података.

Табеле представљају основни и најчешће коришћени елемент база података. Оне служе као основна структура за складиштење информација. Свака табела састоји се од редова и колона. Ћелија представља поље, ред (или запис) представља податке о једном елементу, док колона представља једну од његових карактеристика. Назив колоне истовремено је и назив поља. На пример, табела са подацима о ученицима може имати колоне „Име“, „Презиме“, „Одељење“ и „Број изостанака“, док сваки ред представља једног ученика са конкретним вредностима у тим пољима. Треба водити рачуна о томе да у табели не буде празних редова и колона. . Табеле могу да садрже различите врсте података: текст (као што су имена и адресе), бројеве (као што су оцене и цене), датуме, логичке вредности (да/не), па чак и повезане информације из других табела. Ова разноврсност омогућава да табеле буду флексибилне и применљиве у различитим ситуацијама.

На пример, у апликацији за онлајн продавницу једна табела може да садржи податке о производима, друга о корисницима, а трећа о поруџбинама. Све табеле повезане су заједничким вредностима (кључевима), чиме се ствара сложена мрежа података у релационим базама. У овом уџбенику детаљније ћемо се бавити радом са табелама и прорачунима. Начин креирања релација у бази података учићемо следеће школске године.

Табеле су кључне за управљање информацијама јер обезбеђују јасну структуру, омогућавају брзо претраживање и филтрирање и осигуравају доследност података. Без њих би база била само несређен скуп података без смисла и контроле. Зато су разумевање табела и њихова правилна употреба основа за свакога ко жели да научи како функционишу базе података и како се примењују у стварном свету.

ДА СЕ ПОДСЕТИМО:

Табела

Табела је структура за уређено приказивање података, састављена од редова и колона. Користи се за систематско приказивање и анализу информација. Табеле могу да садрже текст, бројеве, формуле или друге врсте података..

Редови

Редови у табели су хоризонталне линије које се пружају слева надесно. Сваки ред представља један запис или целину повезаних података. На пример, један ред може да садржи све податке о одређеном производу или ученику. Редови су означени бројевима (1, 2, 3, ...) и сваки нови запис обично се уноси у нови ред.

Колоне

Колоне су вертикалне линије означене словима у горњем делу радног листа (A, B, C, ...). Свака колона садржи податке исте врсте. На пример, једна колона може бити намењена именима, друга оценама, трећа датумима итд.

Ћелије

Ћелија је пресек једног реда и једне колоне и представља основну јединицу у коју се уносе подаци. Свака ћелија има своју адресу одређену колоном и редом. На пример, A1 је ћелија која се налази у првој колони и првом реду, док се C3 налази у трећој колони и трећем реду.

.

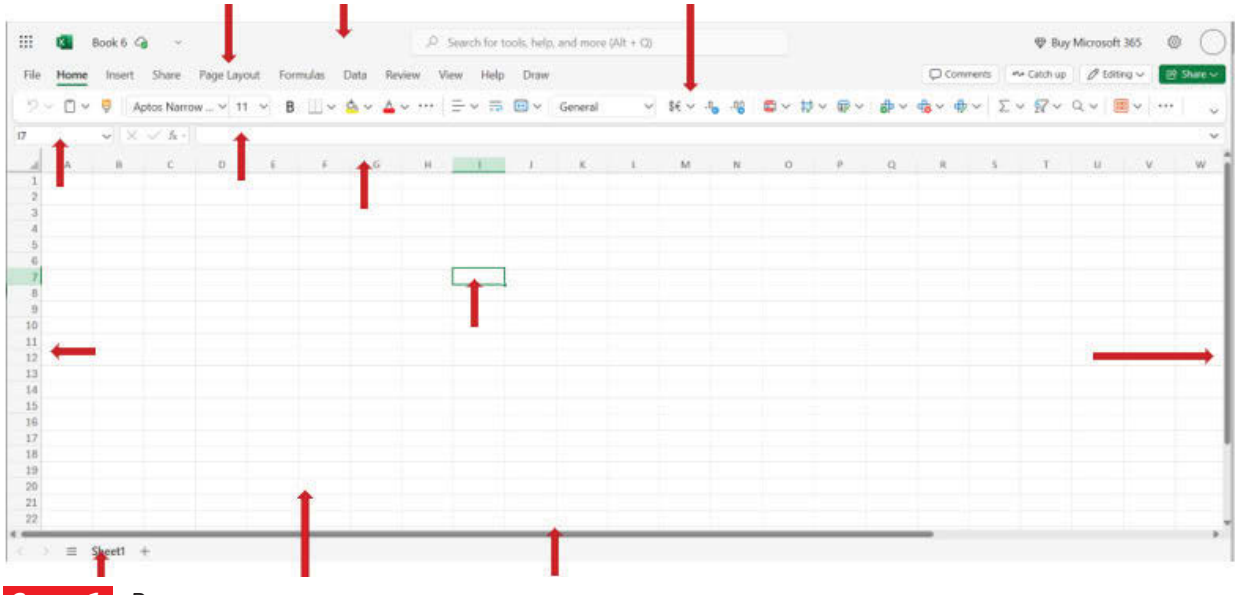
Радна књига

Документ израђен у програму за табеларна израчунавања назива се радна свеска (или радна књига) – Workbook. Радна свеска састоји се од радних листова – Worksheet.

Радни лист је једна страница у радној свесци. Сваки радни лист садржи мрежу састављену од редова и колона и користи се у различите сврхе – од једноставних спискова до сложених израчунавања и анализа.

Свака радна свеска може да садржи један или више радних листова, који су означени картицама на дну прозора (Sheet1, Sheet2...). У њих се уносе, обрађују и чувају подаци, као што су бројеви, текст и формуле. Радне листове можеш додавати и брисати. Подаци у њима могу бити потпуно независни једни од других, али могу бити и међусобно повезани.

Који су елементи радног прозора програма за табеларна израчунавања? Именуј означене делове на слици 1.!



Слика 1 *Радни прозор*

Појмови: Насловна трака, трака са алаткама, радни лист, вертикални клизач, активна ћелија, ознака колоне, ознака реда, радна табела, хоризонтални клизач, трака за формуле, трака менија (наредби), адреса активне ћелије / опсега / именованог објекта.

5.1.1 Операције са радним листовима

Када радиш на већим пројектима или задацима који садрже велику количину података, уношење свега у исти радни лист може створити неред и отежати прегледност. Уместо тога, податке је боље организовати у засебне радне листове, при чему ће сваки лист имати јасну намену или садржај, а сви ће припадати истом документу.

Оваквим начином организовања рад постаје једноставнији и ефикаснији – добијаш бољи преглед података, лакше управљаш информацијама и можеш да се усредсредиш само на део који обрађујеш, а да притом не изгубиш увид у целину. Истовремено се смањује ризик од грешака и омогућава прецизнија анализа.

Рад са више радних листова захтева обављање различитих радњи, као што су њихово додавање, премештање, преименовање, копирање и брисање. Ове радње називају се операцијама над радним листовима.

ДА СЕ ПОДСЕТИМО!

Додавање новог радног листа

- Кликни на икону плус (+) поред последњег радног листа (доле лево). Нови лист ће се појавити одмах поред активног листа и имаће назив „Sheet2“, „Sheet3“ итд.

Брисање радног листа

- Кликни десним тастером миша на назив листа који желиш да избришеш (у доњем делу прозора, међу картицама). Изабери Delete из менија и потврди избор кликом на OK.

Преименовање радног листа

- двапута кликни на име листа, упиши ноу назив и притисни Enter или
- десни клик на лист > Rename > упиши ново име > потврди избор са OK.

Премештање радног листа

- Кликни и задржи лист који желиш да преместиш, затим вуци (drag and drop) лево или десно. Када дођеш до жељене позиције отпусти клик.

Копирање радног листа

- Десни клик на радни лист > из прозора који се појављује бира се Copy Sheet > кликни где желиш да копираш лист> Paste Sheet (Excel 365 online).
- Десни клик на лист > Move or Copy > у прозору који се појављује изабери где желиш да копираш лист > чекирај поље Create a copy > кликни OK.

Копија ће имати исто име као оригинал, али са додатом бројком (на пр. „Sheet1 (2)“).

5.1.2 Уређивање табеле

Често се дешава да садржај у ћелијама буде предугачак или преширок да би се приказао у целости. У таквим случајевима, веома је важно да умеш да прилагодиш димензије редова и колона својим потребама.

У Excel-у постоји неколико могућности за промену димензија редова и колона.

Ширина колона може да се промени на неколико начина:

- показивачем се долази до десне ивице колоне (између слова колона). Када курсор добије изглед  левим кликом се повлачи у жељеном смеру;
- изабере се Home > Format > Column Width... У поље Column Width уписује се жељена вредност;
- десни клик на назив колоне и избор Column Width из менија.
- Висина редова може да се промени на неколико начина:
- показивачем се долази до доње ивице реда. Када курсор добије изглед  левим кликом се повлачи у жељеном смеру;
- изабере се Home > Format > Row Height... У поље Row Height уписује се жељена вредност;
- кликни десним тастером миша на ознаку реда и из менија изабери Row Height.

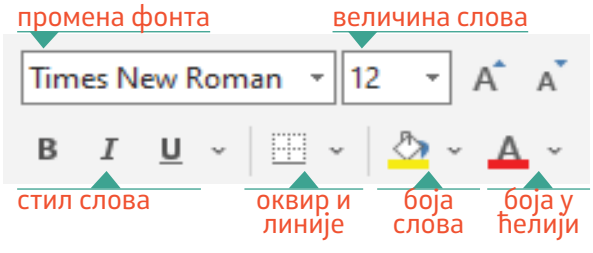
Ширина више колона или висина више редова може се прилагодити истовремено. Најпре означи све колоне или редове чије димензије желиш да промениш, а затим примени исти поступак као приликом промене димензије појединачне колоне или реда.

НАПОМЕНА:

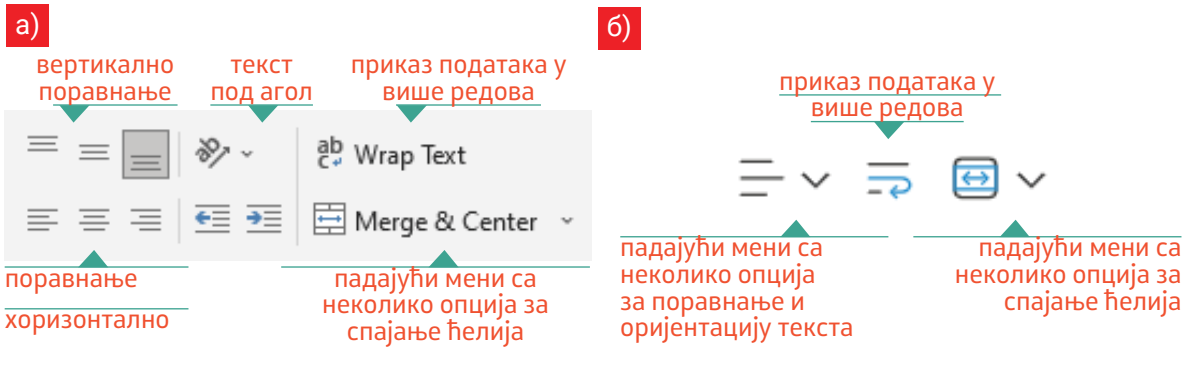
Ако се у ћелији са нумеричким податком појави знак тараба (#), то значи да у ћелији нема довољно „места“ за податак и да је потребно проширити колону.

5.1.3 Уређивање ћелија

Да би подаци били јасни, прегледни и функционални, често је потребно уређивати ћелије. То обухвата промену формата, фонта и његовог изгледа, поравнавање података у ћелијама, додавање боја и граница, спајање више ћелија и слично. Правилно уређене ћелије чине табеле лакшим за читање и анализу.



Слика 2 *Брзи алати за форматирање ћелија*



Слика 3 *Алатке за уређивање текста и ћелија (а) десктоп варијанта (б) Excel 365 онлајн*

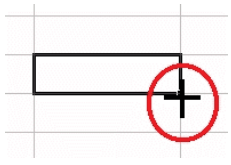
Табелу, односно ћелије, можеш уређивати и у прозору Format Cells: на картици Number можеш изабрати и подесити тип података; на картици Alignment можеш поравнати податке; на картици Font можеш изабрати и уредити фонт; на картици Border можеш изабрати стил, дебљину и боју ивица ћелија.

Напомена: Форматирање ћелија помоћу готовог формата обавља се избором опције Cell Styles на картици Home. Форматирање табела помоћу готовог формата обавља се избором опције Format as Table на картици Home.

5.1.4 Аутоматско попуњавање

Аутоматско попуњавање је веома корисна функција која омогућава аутоматско попуњавање низа ћелија на основу обрасца из претходних ћелија. Његова примена олакшава рад и штеди време приликом израде табела.

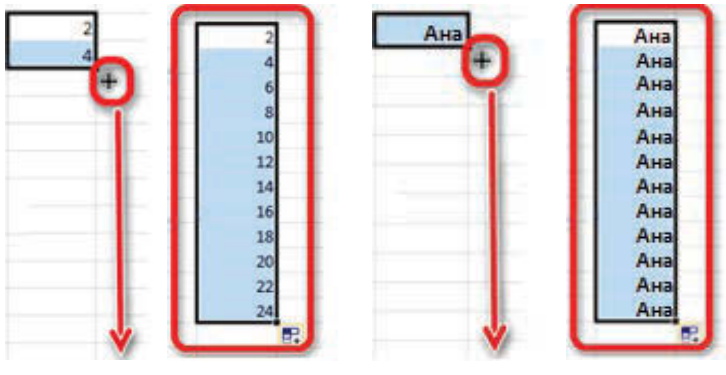
Аутоматско попуњавање најједноставније се обавља превлачењем ручице за аутоматско попуњавање у жељеном смеру. Ручица за аутоматско попуњавање је мали црни квадратић који се налази у доњем десном углу оквира ћелије. Када се показивач постави на њега, он добија облик знака „+“. Потребно је држ



Слика 4 *Аутоматско попуњавање*

Поступак аутоматског попуњавања је следећи:

- Изабери једну или више ћелија које ће послужити као основа за попуњавање осталих ћелија. За низ 1, 2, 3, 4, 5, ... унеси бројеве 1 и 2 у прве две ћелије. За низ 2, 4, 6, 8, ... унеси бројеве 2 и 4. За низ 2, 2, 2, 2, ... унеси само број 2 у прву ћелију. Текстуални податак такође унеси само у прву ћелију;
- Означи унете бројеве или текст, а затим превуци ручицу за аутоматско попуњавање у жељеном смеру.



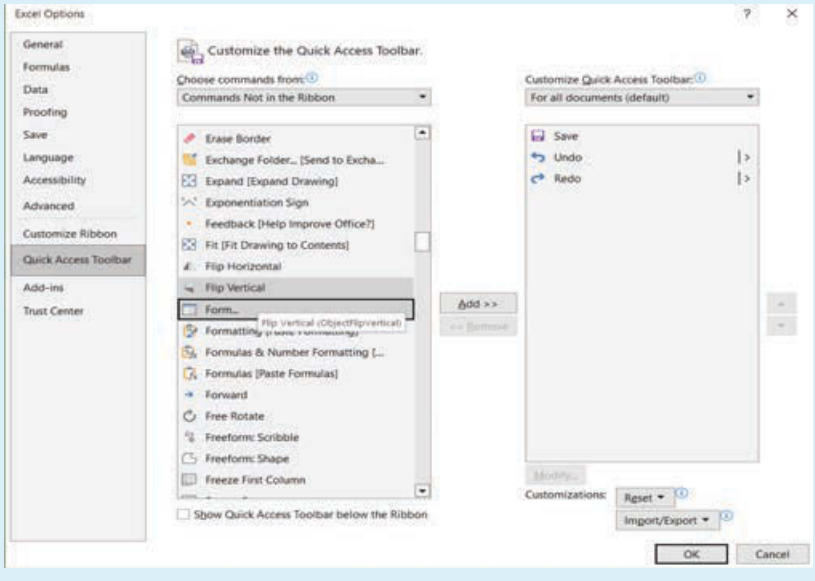
Слика 5 *Поступак аутоматског попуњавања*

- По потреби, може да се кликне на опцију за аутоматско попуњавање  и да се изабере жељена опција.

Креирање обрасца за уношење података у базу

Базе података обично садрже табеле са великом количином података. Ручно уношење тих података је напорно и дуготрајно, подложно људским грешкама и носи ризик од случајне измене претходних уноса. Поступак уношења података у табелу може се побољшати креирањем обрасца. Међутим, најпре треба да додаш одговарајућу алатку на траку са алаткама за брзи приступ.

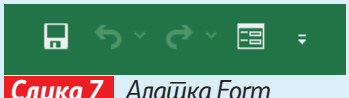
Додавање Form у траку са алаткама



Слика 6 *Прозор Excel Options*

Кораци:

1. отвори File -> Options - > Quick Access Toolbar,
2. у прозору Chose commands from одбери Commands Not in the Ribbon,
3. у доњи пано кликни на Form,
4. кликни на Add,
5. Form ће се пребацити у десни пано,
6. потврди са OK.



Слика 7 *Алатка Form*

Сада ће се у алаткама за брзи приступ јавити нова алатка Form

Израда обрасца

Пре употребе обрасца за унос података, табелу треба да претвориш у „паметну“ Excel табелу. То можеш да урадиш тако што ћеш означити табелу и изабрати Insert → Table. У прозору Create Table провери да ли је означена опција „My table has headers“.

	A	B	C	D	E
1	Назив производа	Категорија	Количина на залихама	Цена	Датум набавке
2	телевизор	електроника	32	25,000.00 дин	24-Jan-2025
3	лаптоп	електроника	45	42,000.00 дин	24-Jan-2025
4	мобилни телефон	електроника	16	28,000.00 дин	20-Mar-2025
5	таблет	електроника	56	18,000.00 дин	11-Mar-2025
6	мајице	одећа	220	480.00 дин	5-Mar-2025

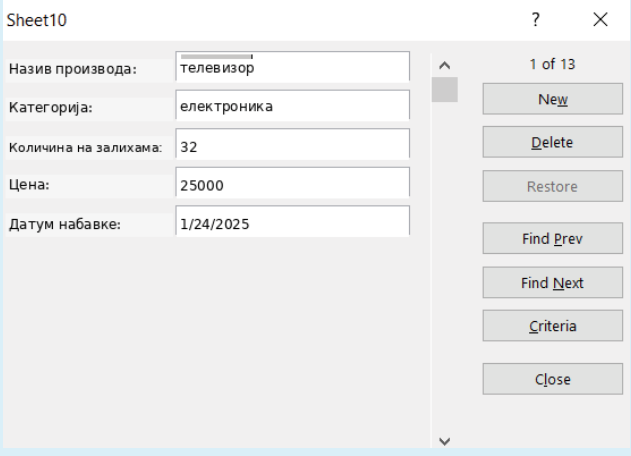
Слика 8 *Пример за Excel табела*

Кораци:

1. Кликни на било коју ћелију у табели.

2. На горњој траци кликни на новододато дугме „Form“. Отвориће се прозор у којем можеш да уносиш податке у одговарајућа поља.

3. Унеси податке и кликни на „New“ да би додао нови запис. Користи „Find Prev“ и „Find Next“ за кретање кроз записе. Ако желиш да избришеш запис, кликни на „Delete



Слика 9 *Образац за унос података у табели из примера*

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Шта представља радни лист у Excel-у и где можеш да га пронађеш?

2. У програму за табеларне прорачуне Сања уноси податак у једну од ћелија. Шта од наведеног може да буде адреса ћелије у коју Сања уноси податак?

а. AB1 б. 1ZA в. 1A1B г. A1B1

3. Које операције са радним листовима познајеш и објасни поступак њиховог извођења?

4. Зашто је корисно користити више радних листова у једном Excel документу?

5. Шта се подразумева под „редом“ у Excel-у? А шта је „колона“?

6. Објасни поступак истовремене промене ширине више колона!

7. Који типови података могу да се унесу у ћелију?

8. Зашто је важно правилно уредити ћелије када радиш са табеларним подацима?

9. Који су кораци за спајање више ћелија у једну?

10. Шта представља аутоматско попуњавање и како може да ти олакша рад при уносу података?

ПРАКТИЧНЕ ВЕЖБЕ:

1. У програму за табеларна израчунавања Excel направи табелу са насловом „Топ 5 омиљених филмова/књига/песама“ (изабери једну категорију која ти је блиска). Креирај табелу са пет редова који представљају твојих пет омиљених избора. Табела треба да има пет колона: Ранг Наслов ЖанрОцена (1–10) Коментар

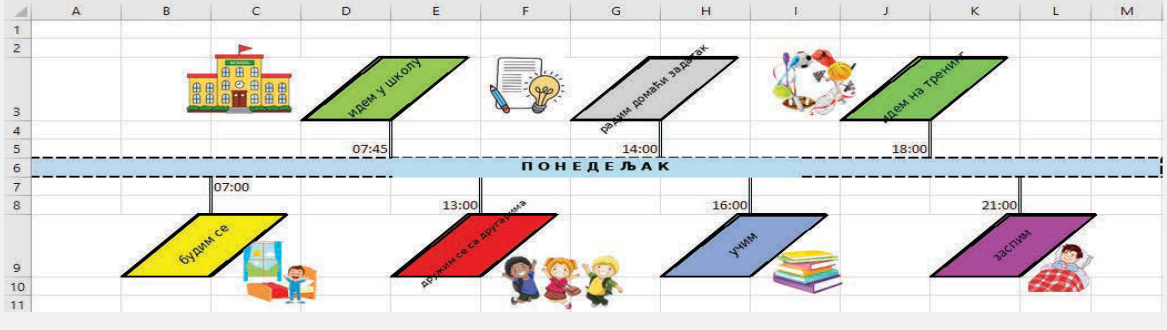
а) Наслов табеле (на пример, „Мојих топ 5 филмова“) форматирај на следећи начин: фонт Arial Black, величина 16, плава боја и центрирано изнад табеле у спојеним ћелијама.

б) Називе колона (Ранг, Наслов, Жанр итд.) подебљај и центрирај. Користи фонт Calibri или Verdana, величину 12 и светлосиву позадину.

в) Текст у колони Коментар испиши курзивом, величине 11.

г) Постави оквире табеле.

2. Креирај хронолошку траку за тему по сопственом избору.



Слика 10 *Пример хронолошке линије „Један дан из моје свакодневице“*

5.2 Напредно коришћење формула и функција

Када би имао табелу са оценама, како би најлакше сазнао који ученик има највишу просечну оцену? Ако имаш табелу са месечним трошковима, како би израчунао укупан износ потрошен само у јануару? Када имаш много података, који програм ти омогућава да их лако организујеш у табеле, брзо обавиш израчунавања и прикажеш резултате на разумљив начин? Шта користиш да би се израчунавања у табели обављала аутоматски, без „ручног“ рачунања?

5.2.1 Појмови у програму за табеларне прорачуне

ДА СЕ ПОДСЕТИМО:

- **Формула** је израз који се уноси у ћелију и који омогућава програму да изврши прорачун над подацима у табели. Формулом програму задајемо које вредности да узме, како да их повеже (на пример, да их сабере или одузме) и где да прикаже резултат. Формуле могу да садрже бројеве, адресе ћелија, математичке знакове и функције за добијање резултата.
- Примери формула:
  - =(5+6)*9
  - =(B1+B2)-B12
  - =9+10+B6
  - =((B10)*10)/(B2+4)
- **Функције** описују се као унапред дефинисане формуле које су уграђене у програм.

Елементи функције су **назив функције** и **аргументи.**

= ImeNaFunkcija (argumenti)

ImeNaFunkcija је реч која описује задатак који треба да се изврши. Примери: SUM – за сабирање, AVERAGE – за израчунавање просека, MIN – за израчунавање минималне, најмање вредности, MAX – за израчунавање максималне, највеће вредности.

Argumenti могу бити различите вредности у зависности од тога коју функцију користимо и шта желимо да постигнемо. Могу бити текст, адресе ћелија, опсези ћелија, логичке вредности или друге функције. Примери:

- SUM(A1:A10) аргумент је опсег ћелија A1:A10, који садржи бројеве за сабирање.
- AVERAGE(10, 20, 30) аргументи су константе за које ће се израчунати просек.
- MAX (B2, B3, B4) аргументи су три засебне адресе ћелија B2, B3, B4 из којих ће бити приказана највећа унета бројчана вредност.
- MAX (SUM(A1:A3), 100) аргументи су функција SUM(A1:A3) и константа 100.

. . .

Оператори су саставни део формула и функција који омогућавају конкретне математичке, логичке или текстуалне операције у оквиру формуле, односно функције. Постоје различити типови оператора, као што су:

- Аритметички оператори

+ (сабирање), - (одузимање), * (множење), / (дељење) ^ (степеновање).

Пример: Ако у ћелији A1 имаш број 8, а у B1 број 4, формула =A1 + B1 вратиће 12 — јер оператор + сабира та два броја.

- Оператори поређења > (веће), < (мање), = (једнако), >= (веће или једнако), <= (мање или једнако), <> (није једнако).

Пример: Ако је у ћелији A1 број 10, а у B1 број 15, формула =A1 < B1 вратиће TRUE — јер је 10 мање од 15.

- Текстуални оператор & (спајање текста).

Пример: ако се у ћелији A1 налази реч „Дигитални“, а у ћелији B1 реч „свет“, онда ће формула =A1 & « « & B1 приказати: „Дигитални свет“ (омогућава додавање размака између речи).

- Референтни оператори: (за опсег, користи се када желимо да изаберемо групу ћелија које су једна до друге).

На пример: Ако имаш бројеве у ћелијама од A1 до A6 и желиш да их сабереш, написаћеш =SUM(A1:A6), (за раздвајање, када желимо да изаберемо неколико ћелија или опсега, али они нису једни до других.).

Пример: функцијом =SUM(A1, C1:C3) сабира се вредност из A1 и све вредности из ћелија од C1 до C3. (размак, за пресек)

Пример: функцијом =SUM(A1:A5 A3:C4) као резултат ће се добити збир вредности у A3 и A4, јер су то ћелије које су заједничке за оба опсега.

. . .

Већ неколико пута спомињемо појам опсег ћелија. Шта значи тај појам?

Опсег ћелија представља групу ћелија изабраних за рад са подацима. Притом ћелије могу:

- да се налазе у истом реду, на пример B2:D2 (овај опсег обухвата ћелије B2, C2, D2),
- да се налазе у истој колони, на пример A1:A3 (овај опсег обухвата ћелије A1, A2, A3),
- да образују правоугаону област, на пример A1:C3 (овај опсег обухвата ћелије A1, A2, A3, B1, B2, B3, C1, C2, C3).

. . .

Адреса ћелије је јединствени идентификатор који означава положај ћелије у радном листу. Састоји се од слова колоне (означене словима: A, B, C...) и броја реда (означеног бројевима: 1, 2, 3...).

Пример:

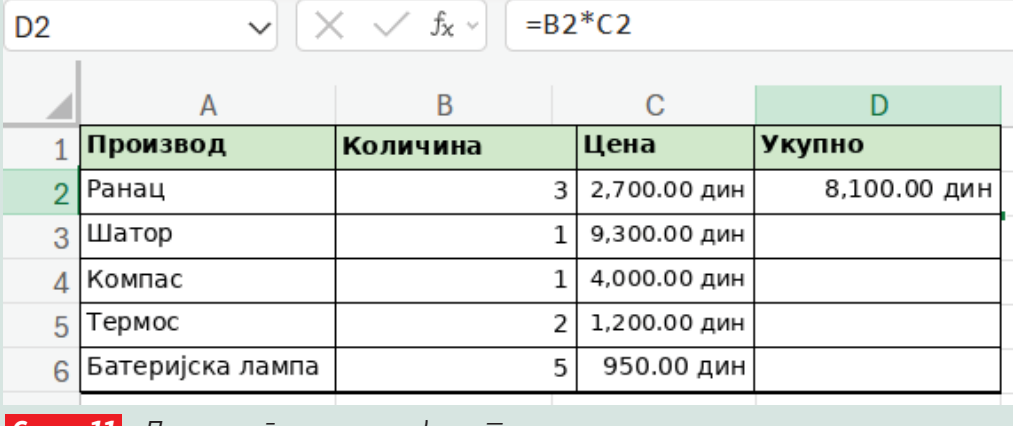
- A1 → Колона A, ред 1
- D5 → Колона D, ред 5

Постоји неколико врста адреса у зависности од тога како се адреса мења када се формула копира или премешта:

- Релативна адреса означава положај једне ћелије или опсега ћелија у табели, али није фиксна. Када се формула са релативном адресом копира или премешта у другу ћелију, адреса се аутоматски мења у складу са новом локацијом. Пише се без икаквих знакова испред слова колоне или броја реда.

На пример:

У ћелији D2 је формула =B2*C2. Ако формулу из ћелије D2 копираш у D3, она ће се аутоматски променити у =B3*C3
То значи да се релативне адресе прилагођавају како би се израчунавање обавило помоћу садржаја из „суседних" ћелија, у зависности од тога где поставиш формулу.



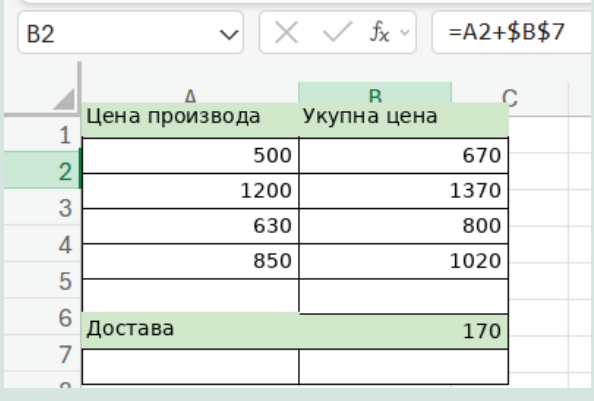
	A	B	C	D
1	**Производ**	**Количина**	**Цена**	**Укупно**
2	Ранац	3	2,700.00 дин	8,100.00 дин
3	Шатор	1	9,300.00 дин	
4	Компас	1	4,000.00 дин	
5	Термос	2	1,200.00 дин	
6	Батеријска лампа	5	950.00 дин	

Слика 11 *Пример за примена референтна адреса*

- Апсолутна адреса је фиксна адреса ћелије која остаје иста када формулу копирамо или премештамо у другу ћелију. Она увек показује на тачно одређену ћелију. Означава се додавањем знака $ испред слова колоне и броја реда. Тако је $A$1 апсолутна адреса која увек показује на ћелију у колони A, ред 1.

Пример: У ћелији B2 налази се формула =A2+$B$7. Ако формулу из B2 копираш у B3, она ће се аутоматски променити у =A3+$B$7. Дакле, адреса $B$7 остаје фиксна (увек узима цену из B7), док се адреса A2 мења у A3, A4 итд.

Постоји неколико врста адреса у зависности од тога како се адреса ћелије мења када се формула (функција) копира или премешта:



Слика 12 *Пример за примена на апсолутна адреса*

- Мешана адреса је комбинација апсолутне и релативне адресе што значи да је један део адресе фиксан, а други се мења кад копирамо формулу.

Пример: $A1 - Колона „A“ је фиксна, а ред „1“ се мења, A$1 → Ред „1“ је фиксан, а колона „A“ се мења.

5.2.2 Напредне функције

До сада си научио да користиш основне функције, као што су SUM, AVERAGE, MIN и MAX, за једноставна израчунавања у табелама. Али шта ако је потребно да из табеле добијеш прецизније информације, на пример:

- Колико ученика има оцену 5?
- Колико је укупно новца потрошено само на храну?
- Како да проверимо да ли је неки ученик положио или није, на основу оцене?

Одговоре на оваква питања дају напредне функције као што су COUNT, COUNTIF, SUMIF и IF.

Помоћу њих можемо:

- Бројимо колико пута се нешто појављује (COUNTIF),
- сумирамо вредности преема услову (SUMIF),
- направимо избор или одлуку у формули (IF),
- пребројавамо колико вредности има уопште (COUNT).

ДА СЕ ПОДСЕТИМО:

Функцију можеш да унесеш у ћелију директним уписивањем.

Кораци:

1. Кликни на ћелију у коју желиш да унесеш функцију (селектуј је).

2. Упиши = да означиш да уносиш функцију.

3. Унеси функцију, на пример: =SUM(A1:A10).

4. Притисни Enter да је примениш.

5. Функцију можеш да унесеш и преко менија Formulas.

Кораци:

1. Кликни на ћелију у коју желиш да унесеш функцију.

2. На траци менија кликни на картицу Formulas.

3. Изабери категорију функција (од категорија: финансијске, логичке, текстуалне, функције за датум и време, функције за референце, математичке и тригонометријске функције).

4. Изабери конкретну функцију.

5. Унеси аргументе које функција користи.

6. Притисни Enter да је примениш.

. . .

Функција COUNT

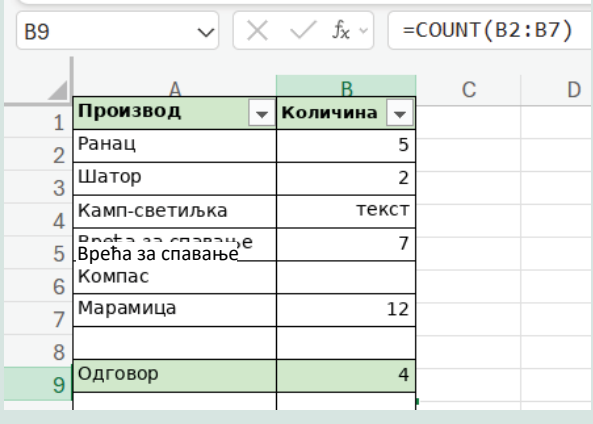
Функција COUNT пребројава колико ћелија у изабраном опсегу садржи бројчане вредности. То значи да броји само ћелије које садрже бројеве, док занемарује празне ћелије, текст и друге ненумеричке вредности.

Формат функције је:

=COUNT (опсег1, опсег2, ...)

при чему аргумент може бити један или више опсега у којима ће се бројати само бројчане вредности.

Пример: У продавници се води табела о дневној продаји различитих производа. У колону А унети су називи производа, а у колону В продате количине. Неке ћелије су празне или садрже текст ако производ тог дана није продат. Колико је различитих производа продато тог дана?



Слика 13 *Пример табеле дневне продаје различитих производа*

Одговор: =COUNT(B2:B7). Продата су 4 различита производа.

НАПОМЕНА: За разлику од функције COUNT, која броји само ћелије са бројчаним вредностима, COUNTA узима у обзир и ћелије са текстом, датумима, логичким вредностима и другим садржајима.

Функција COUNTIF

Функција COUNTIF броји колико пута је испуњен одређени услов у задатом опсегу ћелија. То значи да броји колико ћелија има конкретан број, колико ћелија има одређени текст или колико је ћелија са вредношћу већом или мањом од одређене вредности.

Формат функције је:

=COUNTIF(опсег, услов)

где је опсег део табеле у којем тражиш податке, а услов је критеријум који треба да испуњавају ћелије да би биле пребројане.

Пример:

- =COUNTIF (A1:A10, ">50") – броји само ћелије које имају вредност већу од 50 у опсегу A1:A10.

- =COUNTIF(B1:B20,"Завршено") – броји колико пута се реч „Завршено“ појављује у B1:B20.
- =COUNTIF(C1:C15, ">=10") – броји ћелије са вредношћу једнаком или већом од 10 у опсегу C1:C15.
- =COUNTBLANK(D1:D30) – броји празне ћелије у опсегу D1:D30.

Пример: Колико пута се јавља реч „Јабука “ у колони Воће?

	A
1	Воће
2	Јабука
3	Банана
4	Јабука
5	Лимун

Слика 14 *Пример за колону „воће“*

Одговор: = COUNTIF(A2:A5, "Јабука"). За пример дат на слици 14, одговор је 2 пута.

. . .

Функција SUMIF

Функција SUMIF сабира вредности из ћелија које се налазе у одређеном опсегу и испуњавају задати услов. То значи да функција не сабира вредности из свих ћелија, већ само из оних које одговарају критеријуму који задајеш.

Формат функције је:

=SUMIF(опсег, услов, опсег_за_збир]))

где је опсег опсег ћелија у којем се проверава услов, услов је критеријум који треба да испуни свака ћелија из опсега треба да испуни, док је опсег_за_збир (опционо) опсег из којег ће се сабирати вредности. Ако се не наведе, сабирају се ћелије из првог опсега.

Пример:

- =SUMIF(A1:A10, ">50")→ сакупља само бројеве веће од 50 у A1:A10.
- = SUMIF(A1:A10, 50) → сумира бројеве који су једнаки 50.
- = SUMIF(B1:B10, "Продато") → сумира ако је вредност „Продато“.

Пример: Из дате табеле (слика 15) сумира број продатих ранаца.

	A	B
1	**Производ**	**Продата количина**
2	Ранац	5
3	Шатор	2
4	Ранац	3
5	Компас	4
6		
7	Продати ранчеви	

Слика 15 *Пример табеле за сумирање продаје одређеног производа*

Функција IF

Функција IF омогућава логичку проверу услова и враћа различите вредности у зависности од тога да ли је услов испуњен (TRUE) или није (FALSE). То значи да функција проверава да ли је нешто тачно или не, након чега враћа једну вредност ако је услов тачан и другу ако није.

Формат функције је:

=IF(логички_услов, вредност_ако_је_истинито, вредност_ако_није_истинито)

где је логички_услов израз који проверавамо, вредност_ако_је_истинито — резултат ако је услов испуњен, вредност_ако_није_истинито — резултат ако услов није испуњен.

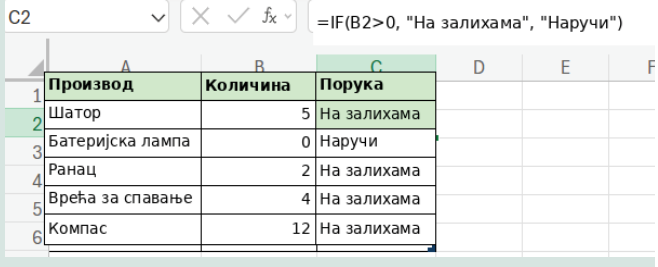
Пример:

- =IF(A1>50, "велики број", "мали број") – ако је вредност у ћелији A1 већа од 50, као резултат ће бити приказано „велики број"; у супротном, приказаће „мали број".
- =IF(A1>0, "Позитиван", "Негативан") – ако је вредност у ћелији већа од нуле, као резултат ће бити приказан текст „Позитиван"; у супротном, биће приказано „Негативан".

Пример: На слици 16 приказана је табела са производима и њиховом тренутном количином на залихама. У колони „Порука" треба да пише „На залихама" ако има 1 или више комада или „Наручи" ако има 0 комада. Коју функцију ћеш користити?

	A	B	C
1	**Производ**	**Количина**	**Порука**
2	Шатор	5	
3	Батеријска лампа	0	
4	Ранац	2	
5	Врећа за спавање	4	
6	Компас	12	

Слика 16 *Табела са производима и њиховом тренутном количином на залихама.*



Слика 17 *Решење задатка*

Одговор: =IF(B2>0, «На залихи», «Наручи»)

ЗАКЉУЧАК

Када користиш функцију у програму за табеларне прорачуне, прво наводиш назив прорачуна који желиш да извршиш (на пример: SUM, IF), а затим, у заградама, уносиш аргументе — односно шта тачно треба израчунати. Аргументи функцији дају флексибилност да ради са различитим врстама података. У формулама и функцијама оператори имају кључну улогу — Они се користе за израчинавања (на

▸▸▸

▸▸▸

пример: +, -, *, /), поређења (на пример: =, >, <) и рад са текстом (на пример: & за спајање текста).

Када користиш функцију, прво пишеш који прорачун желиш да извршиш (преко назива), а затим шта тачно треба да израчуна (преко аргумената у заградама). Аргументи функцијама дају флексибилност за рад са различитим подацима.

Оператори имају кључну улогу у прорачунима, поређењима и управљању подацима.

Опсег ћелија представља групу повезаних ћелија и користи се у формулама и функцијама за анализу или израчунавање података из више ћелија одједном.

Када радиш са формулама, важно је да разликујеш релативну адресу (на пример: B2), која се мења при копирању, и апсолутну адресу (на пример: $B$2), која остаје иста чак и ако се формула копира на друго место. То ти омогућава већу контролу над израчунавањима.

Напредне функције као што су COUNT, COUNTIF, SUMIF и IF, омогућавају бројање, сабирање и логичку проверу према задатим условима, посебно су корисне за анализу података.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Која је разлика између формуле и функције у програму за табеларна израчунавања?

2. Како изгледа синтакса (формат) функција у програму за табеларна прорачунавања?

3. Шта је аргумент функције и које врсте аргумената једна функција може да има?

4. Дата је формула =AVERAGE(A1:A10). Шта представља A1:A10 у овој формули?

5. Шта је оператор и које врсте постоје?

6. Који симболи се користе за дефинисање континуираног и раздвојеног опсега?

7. Како би се означио опсег ћелија од A1 до D10?

8. Објасни појмове апсолутна и релативна адреса! Када треба да користиш апсолутне адресе, а када релативне?

9. Шта ће се догодити ако формулу =A1*2 (из B1) копираш у B2?

10. Како се мешовита адреса ($A1 или A$1) разликује од апсолутне и релативне?

11. Наведи примере употребе функција COUNT, COUNTIF, SUMIF и IF!

ПРАКТИЧНЕ ВЕЖБЕ

1. Направи табелу „Производи“ са колоном производи и колоном основна цена! Сваки производ добија фиксну доплату од 20 динара за испоруку, која се налази у посебној ћелији.

	A	B	C	D
1	**Производ**	**Основна цена**	**Доплата за испоруку**	**Крајња цена**
2	Јабука	50	20	
3	Банана	40		
4	Поморанџа	60		
5				

Слика 18 *Табела „Производи“*

2. У колони D израчунај коначну цену додавањем доплате! Употреби релативну адресу за основну цену (како би се мењала приликом копирања) и апсолутну адресу за доплату из ћелије C1 (како се не би мењала приликом копирања)!

3. У табелу унеси различите вредности (бројеве, текст и празне ћелије)! Помоћу одговарајуће функције одреди колико има бројчаних вредности.

4. У табелу су унета имена ученика и њихов статус на данашњем часу информатике („Присутан“ или „Одсутан“). Изброј колико је ученика присутно на часу!

5. Креирај дату табелу и израчунај колико је поруџбина Марко направио и колико је новца потрошио на њих!

	A	B
1	**Клијент**	**Износ наруџбине**
2	Марко	1000
3	Ана	850
4	Марко	750
5	Иван	600

Слика 19 *Табела „Наруџбине“*

6. Направи табелу са колонама „Име ученика“ (најмање 10 ученика) и „оцена из Информатике“! Помоћу функције изброји колико ученика има оцену 5 из предмета Информатика!

7. Направи табелу у којој ћеш за недељу дана евидентирати колико сати дневно учиш за сваки предмет (колона А – предмети, колона Б – колико сати учиш)! Израчунај укупну суму радних сати које си посветио Математици!

8. У табели је уписан број извршених задатака од стране запослених. Потребно је да се прикаже „Да“ ако је запослени завршио 10 или више задатака, и „Не“ ако је супротно.

9. У табелу унеси вредности за температуру. Ако је испод 0 °C, поред колоне „температура“ треба да пише «Хладно», ако је изнад 30 °C, «Топло», иначе «Умерено».

5.3 Напредни рад са графиконима

Радни листови у програмима за табеларна израчунавања могу да садрже велики број података који су често тешки за разумевање или објашњавање. Подаци могу постати много јаснији када их визуелно представиш помоћу графикона. Све што треба да урадиш јесте да изабереш податке и одговарајућу врсту графикона. Постоје различите врсте графикона (стубичасти, линијски, кружни...), а треба да изабереш ону која ће најефикасније представити податке. Ако графикон није јасан, неће бити много користан.

ЗАДАЦИ ЗА ОБНАВЉАЊЕ:

Број посетилаца у парку током недеље

Унеси податке о посећености:

Ден	Број на посетители
Понеделник	120
Вторник	150
Среда	200
Четврток	180
Петок	220

Креирај стубасти дијаграм и анализирај када има највише посетиоца.

Могуће решење задатка за понављање је графикон приказан на слици 20.

Слика 20 *Графикон за задатак „Број посетилаца у парку током недеље“*

5.3.1 Елементи графикона

Да би разумео/ла графикон, важно је да препознаш његове главне елементе – као што су наслов, осе, легенда и сами подаци који су визуелно приказани.

1. Графикон обично има две осе:

- **X-оса** (хоризонтална) на којој су приказане категорије, датуми, дани итд.
- **Y-оса** (вертикална) на којој су приказане вредности (бројеви, проценти, мере итд.).

1. **Наслов графикона** (Chart Title) који треба да прикаже јасан опис података у графикону.

2. **Наслов осе** (Axis Labels) који је текст који означава категорије или вредности оса.

3. **Легенда (Legend)** чији је задатак да објасни које боје или знаци у графикону представљају одређене податке.

4. **Серије података (Data Series)** које су група повезаних података приказаних као колоне, линије, тачке и слично.

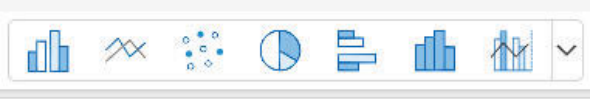
5. **Мрежне линије (Gridlines)** или помоћне линије које могу да буду хоризонталне или вертикалне, што олакшава читање вредности.

6. **Ознаке вредности (Data Labels)**, бројеви који стоје изнад или у самим елементима (стубови, линије) и показују тачну вредност.

Слика 21 *Елементи графикона*

5.3.2 Додатно уређивање графикона

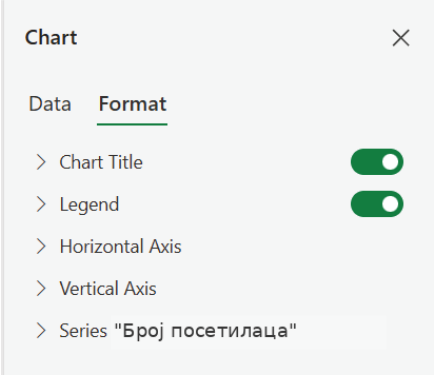
Након што направиш графикон, можеш да се предомислиш и закључиш да нека друга врста графикона боље приказује податке. У том случају можеш лако да промениш стил или тип графикона без поновног уношења података. Све што треба да урадиш јесте да кликнеш на графикон, из менија изабереш Chart, а затим са листе изабереш један од понуђених типова.



Слика 22 *Алатки за избор за тип на графикон*

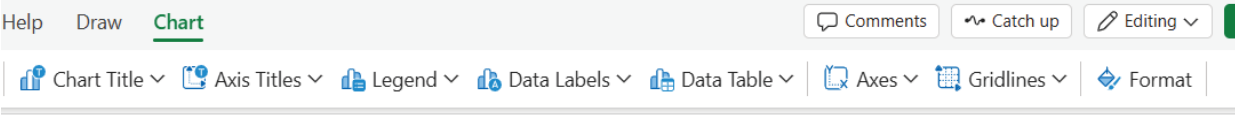
Уместо да користиш готов стил, сваки део графикона (стуб, линија, оса, мрежна линија, легенда итд.) можеш појединачно да уредиш. У Excel 365 то можеш да урадиш на два начина:

- Двојни клик на елемент – отвара се панел за форматирање, где можеш да мењаш стил, боју, величину и друге параметре.



Слика 23 *Алати за уређивање елемената графикона*

2. Коришћење менија „Chart“.

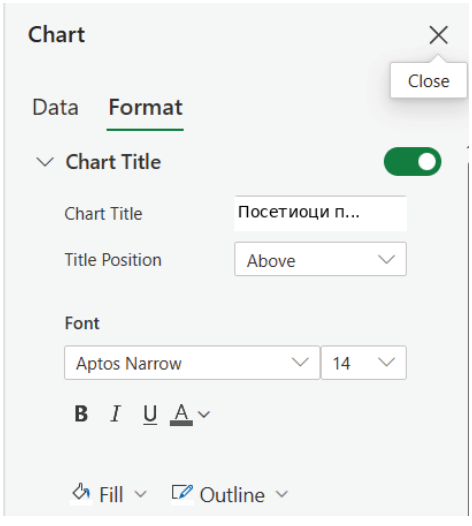


Слика 24 *Алати за уређивање – мени Chart*

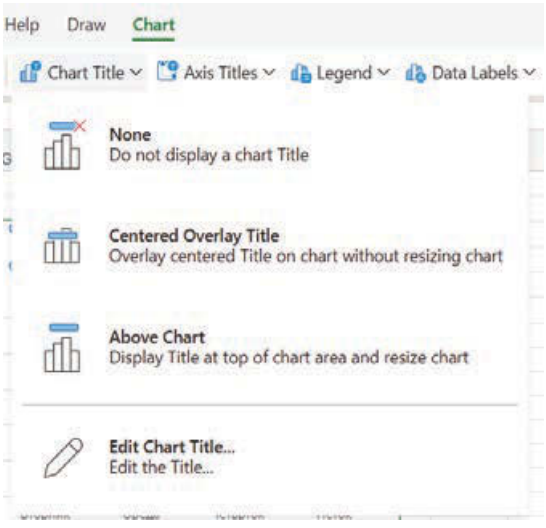
Уређивање наслова графикона

3. Два пута кликни на наслов и унеси нови текст. У Format > Chart Title, можеш да промениш тип фонта, величину, боју, поравнање (слика 25).

4. Ако користиш Chart > Chart Title, можеш да га поставиш на различите позиције (слика 26).



Слика 25 *Уређивање наслова*



Слика 26 *Позиционирање наслова*

Уређивање оса

- Два пута кликни на X-осу или Y-осу да би се отворила опција за форматирање оса.
- Можеш да мењаш опсег вредности, назив оса, скалу...
- У Chart > Axes можеш да укључиш или искључиш осе.

Уређивање легенде

- Преко Chart > Format > Legend легенду можеш да укључиш/искључиш, премештиш, форматираш или промениш фонт.
- Преко Chart > Legend можеш да је поставиш лево, десно, горе или доле.

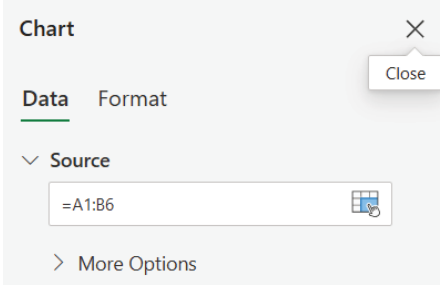
Прилагођавање серија података:

- Двапут кликни на стубове/линије > Chart > Format > Data Series и можеш да мењаш боје, стил и дебљину.

Поновни избор извора података

Ако желиш да промениш опсег података графикона:

- Кликни на графикон, иди на Chart > Select Data > Source



Слика 27 *Позиционирање наслова*

Уређивање вредносних ознака

- Два пута кликни на вредносне ознаке > Chart > Format > Data Labels и можеш да мењаш формат броја, фонт, садржај ознака.
- У Chart > Data Labels можеш да одабереш да ли да се прикажу вредносне ознаке података.

ЗАКЉУЧАК

Графикони су моћна алатка за визуелно приказивање података, чиме се омогућава лакше разумевање и анализа информација. Програми за табеларна израчунавања омогућавају и напредно уређивање графикона –мењање њиховог типа, уређивање наслова, оса, легенде и серије података, као и прилагођавање вредносних ознака. Осим тога, поновљеним избором извора података, графикони могу да се ажурирају без потребе за новим креирањем.

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

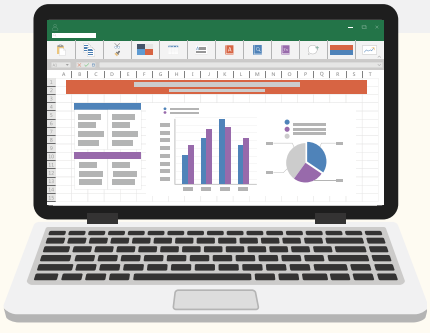
1. Шта представља графикон?
2. Шта је важно при избору типа графикона?
3. Који су основни елементи једног графикона?
4. У чему је улога X-осе, а у чему Y-осе?
5. Шта представља легенда графикона?
6. Шта су вредносне ознаке (data labels)?
7. Како можеш да промениш тип већ креираног графикона?
8. На који начин може да се уреди наслов графикона?
9. Како можеш да преместиш или да форматираш легенду?
10. Објасни како можеш поново да изабереш извор података за графикон!
11. Зашто је важно да графикон буде визуелно јасан и читак?
12. Како би одлучио/ла који тип графикона најбоље приказује дате податке?
13. Уколико вредности нису приказане на стубовима, да ли ће графикон бити разумљив? Објасни!

ПРАКТИЧНА ВЕЖБА:

Следећи подаци приказују температуре измерене током једне недеље:

Дан	Температура (°C)
Понедељак	12
Уторак	15
Среда	17
Четвртак	14
Петак	13
Субота	16
Недеља	18

1. Унеси податке у табелу.
2. Креирај стубасти графикон.
3. Додај наслов графикону.
4. Додај наслове осама („Дани“, „Температура“).
5. Прикажи вредносне ознаке (data labels) на врху сваког стуба.
6. Промени боју стубова – стуб који приказује максималну температуру обој црвеном бојом, за минималну температуру зеленом бојом, а остале плавом бојом.
7. Додај још један ред података и прошири графикон.
8. Сними документ.
9. Промени тип графикона у кружни (pie chart).
10. Прикажи проценте у ознакама делова круга.
11. Проследи закључак: Који тип графикона стално приказује податке?



5.4 Сортирање

Сортирање података значи њихово рећање по утврђеном редоследу. Реч је о једној од најосновнијих и најкориснијих функција. Сортирање може да се изврши по азбучном реду, нумерички (од најмањег до највећег или од највећег до најмањег броја), хронолошки (од најстаријег до најновијег и од најновијег до најстаријег датума/времена), по редоследу који је дефинисао корисник или по формату, укључујући боју ћелије, боју фонта или иконе. Основни разлог за сортирање података је омогућавање бољег разумевања, организације и проналажења података – сортирани подаци се много брже и лакше претражују.

На шта треба да обратиш пажњу: Приликом сортирања треба да изабереш целу табелу. Ако сортираш само једну колону без остатка табеле, редови ће се измешати, што доводи до нетачних података (на пример, оцена једног детета појавиће се поред имена другог детета). Избегавај празне редове, колоне и спојене ћелије јер могу да изазову проблеме приликом сортирања. Препоручује се и да колоне имају наслове.

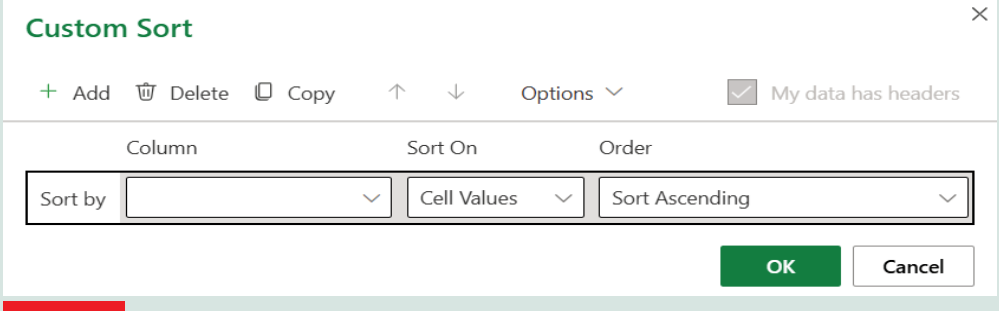
ДА СЕ ПОДСЕТИМО:

Сортирање се активира избором Home > Sort & Filter. Притом:

- Текст можеш да сортираш по азбучном редоследу (Sort A to Z) или по обрнутом азбучном редоследу (Sort Largest to Smallest).
- Бројеве можеш да сортираш од најмањег ка највећем (Sort Smallest to Largest) или од највећег ка најмањем броју (Sort Largest to Smallest).
- Датум и време можеш да сортираш хронолошки – од најстаријег ка најновијем (Sort Oldest to Newest) или од најновијег ка најстаријем (Sort Newest to Oldest).

. . .

Избор опције Custom Sort (прилагођено сортирање) омогућава ти да сортираш податке према више колона истовремено или према посебним критеријумима — као што су боја ћелије, боја фонта или икона, ако је примењено условно форматирање. Ово је посебно корисно када стандардно сортирање по азбучном или бројчаном редоследу није довољно.

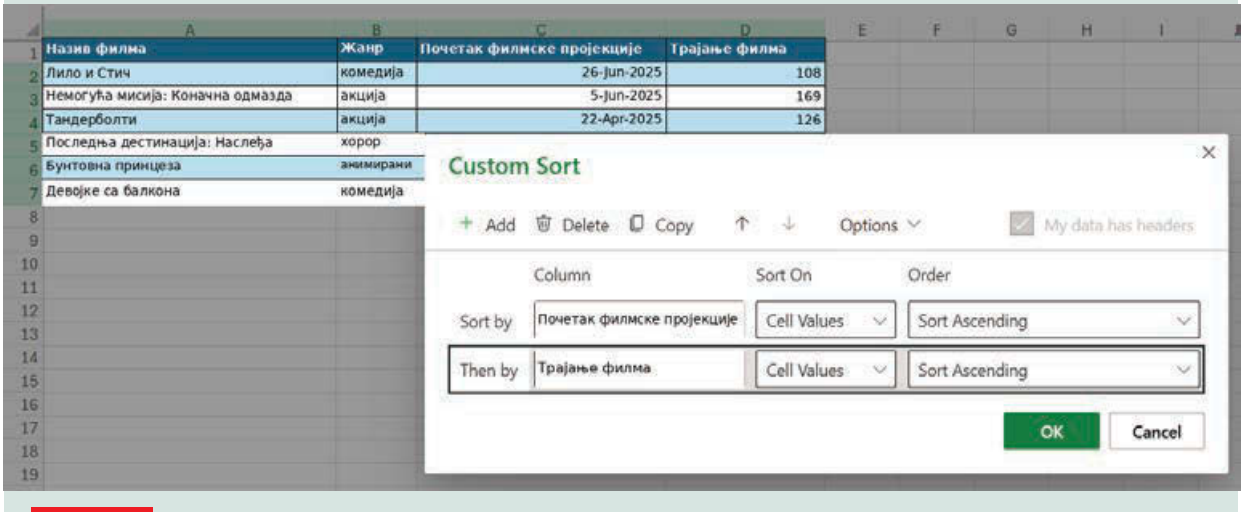


Слика 28 *Прозор Custom Sort*

У прозору Custom Sort треба да изабереш:

- колону по којој ћеш сортирати (у области Column),
- критеријум по којем ћеш сортирати (Cell Values – према садржају ћелије, Cell Color – према боји ћелије, Font Color – према боји фонта или Conditional Formatting Icon – према икони ако си применио/ла условно форматирање) и
- редослед по којем ћеш сортирати (растући/опадајући), након чега се бира OK.

Уколико желиш да додаш више критеријума, то можеш да урадиш кликом на дугме Add, које се налази у горњем левом углу дијалошког прозора Custom Sort. У новом критеријуму Then by на исти начин одређујеш како ће се сортирање извршити. Не заборави да први наведени критеријум има предност. Да би се сортирање извршило, на крају треба да потврдиш подешавања избором опције OK.



Слика 29 *Пример сортирања по два критеријума*

Да би обрисао/ла неки од критеријума, изабери Delete.

Не заборављај да изабереш (да селектираш) поље My data has headers уколико имаш наслове на колонама да спречиш мешање са другим редовима.

ЗАКЉУЧАК

Сортирање података представља једну од најважнијих и најкориснијих функција за ефикасно управљање информацијама. Без обзира на то да ли је реч о једноставном сортирању према једној колони или о напредном прилагођеном сортирању према више критеријума, ова алатка омогућава корисницима да брзо организују и анализирају податке на логичан и разумљив начин.

ПИТАЊА И ЗАДАЦИ ЗА ПРОВЕРУ

1. Шта представља сортирање и зашто је важно?

2. Шта се дешава ако се сортирају подаци без да буду изабране све повезане колоне?

3. Шта представља Custom Sort у Excel-у? Објасни својим речима!

4. Како можеш да сортираш податке на више колона истовремено? Објасни кораке!

5. Зашто је важно да имаш наслове (headers) у табели пре него што сортираш?

6. Који су неки од најчешћих проблема који могу да се појаве при сортирању и како да се реше?

7. Објасни својим примером када и зашто би користио/ла сортирање у свакодневној ситуацији (на пр., распоред часова, листа оцена, итд.)!

ПРАКТИЧНА ВЕЖБА:

1. Направи табелу са следећим колонама: Име ученика, Презиме ученика, Датум рођења, Време писања (време за које ће ученик написати реч „информатика“ у програму за текст). Сортирај табелу по Времену писања, а затим по Датуму рођења.

2. Замисли да твоја школа организује е-спорт турнир на којем се тимови ученика такмиче у популарним видео-играма. Сваки играч је освојио одређен број поена на основу свог учинка на тренинг утакмицама, а утакмице су распоређене у различите дане у недељи.

После сваке од наведених активности копирај сортирану табелу у нови радни лист:

- Сортирај играче према броју поена, од највише ка најмање.
- Сортирај најпре по Тиму, а затим по Поенима (сортирање по две колоне).
- Сортирај табелу по Дану за утакмицу, тако да дани буду поређани следећим редоследом: Понедељак, Уторак, Среда, Четвртак, Петак.
- Који играч има највише поена из Тима Б?

	A	B	C	D
1	**Име играча**	**Поени**	**Дан за утакмицу**	**Тим**
2	Ана	78	Петак	Тим Б
3	Јован	85	Понедељак	Тим А
4	Кристина	85	Среда	Тим А
5	Александар	65	Петак	Тим Б
6	Елена	90	Уторак	Тим А
7	Стефан	72	Четвртак	Тим Б

Слика 30 *Е-спорт турнир*

3. У данашње време YouTube представља једну од најпопуларнијих платформи за забаву, образовање и креативно изражавање, посебно међу младима. Следећа табела приказује неке од познатих YouTube канала. У табели су укључене информације о називу канала, његовој категорији, броју претплатника (у милионима) и датуму почетка рада на платформи.

	A	B	C	D
1	**Канал**	**Категорија**	**Претплатници (милиони)**	**Датум почетка**
2	T-Series	Музика	266	13.03.2006
3	Dude Perfect	Спорт/Забава	61	16.03.2009
4	TED-Ed	Образовање	22	22.03.2012
5	Lozano Music	Музика	0.29	08.02.2012
6	MKD Education	Образовање	0.04	25.10.2020
7	Slatkaristika Off.	Музика	0.31	20.09.2013
8	Fitnes so Aleks	Здравље/Фитнес	0.065	03.07.2019
9	Kids Learn MK	Образовање	0.08	18.03.2019

Слика 31 *Познати YouTube канали*

Након сваке од наведених активности сортирану табелу ископирај у нови радни лист:

1. Сортирај према колони „Претплатници (у милионима)“ – од највећег ка најмањем броју.

2. Сортирај према „Претплатници (милиони)“ – растућим редоследом.

3. Сортирај према „Датум почињања“ – од најстаријег до најновијег канала, а затим према „Категорији“.

5.5 Филтрирање

Филтрирање података у бази омогућава да се прикажу (да се „пропусте“) само они подаци који задовољавају одређене критеријуме, док се остали подаци скривају (не уклањају се). Коришћењем филтера може се уређивати, форматирати и штампати подтабела без промене распореда података. Главна сврха филтрирања јесте да олакша рад са подацима, омогући проналажење одређених информација и поједностави анализу података, без мењања или брисања оригиналних података. У програмима за табеларне прорачуне могу се филтрирати различити типови података: текст (на пример, имена, градови, категорије), бројеви (на пример, износи продаје, количине, проценти), датуми (на пример, датуми наруџбина, датуми рођења) и логичке вредности (TRUE/FALSE).

ДА СЕ ПОДСЕТИМО:

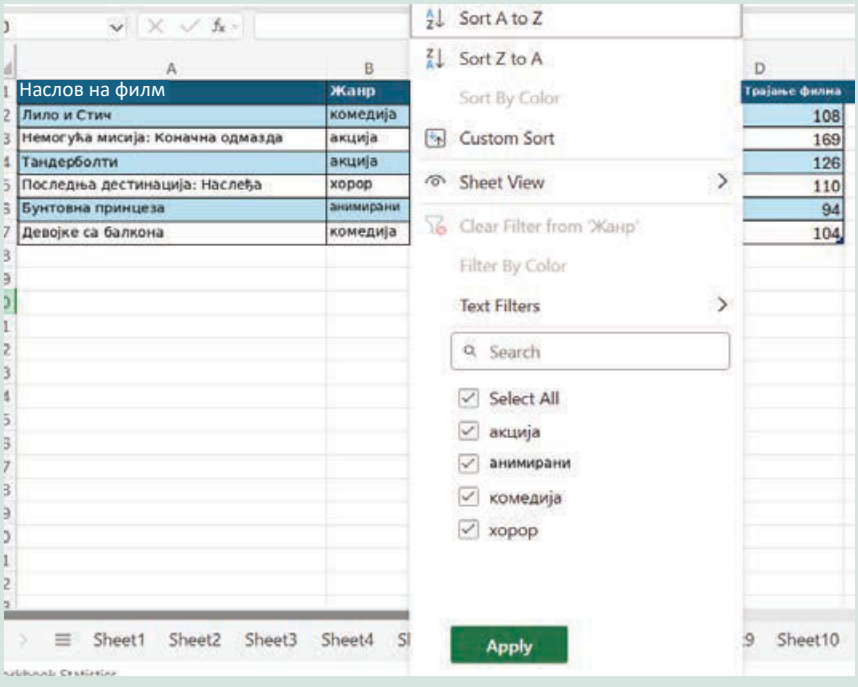
Филтрирање података по тачно одређеним вредностима

Кораци

1. Кликне се на податке које желимо да филтрирамо.

2. Бира се Home → Sort & Filter → Filter.

3. У првом реду табеле у свакој колони појављује се стрелица. Кликом на стрелицу отвара се падајућа листа у којој су приказани сви подаци из колоне, а испред сваког податка налази се поље за потврду.



Слика 32 *Филтрирање података по тачно одређеним вредностима*

4. У пољу за потврду треба да остану изабрани (означени) само потребни подаци (кликом на поље за потврду бирају се подаци).

5. Потврђује се избором Apply.

Филтрирање података постављањем додатног услова

Кораци:

1. Кликне се на податке које желимо да филтрирамо.

2. Бира се Home→ Sort & Filter → Filter.

3. У првом реду табеле у свакој колони појављује се стрелица. Кликне се на стрелицу, након чега се отвара листа из које се бира филтер за текст/број/датум (Text Filters/Number Filters/Date Filters).

4. Са додатне листе бира се опција према чему ће се филтрирати (на пример: за текст – започиње са...., за број – мање од....., за датум – пре...).

5. Отвара се дијалог прозор у који се довршава постављање услова.

6. Кликне се на ОК.

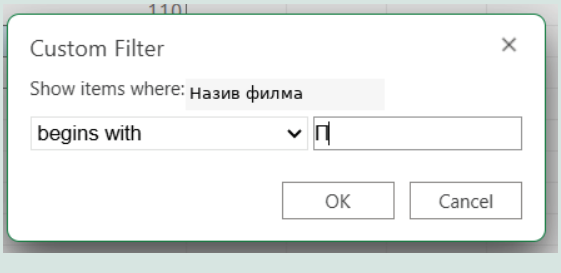
7. Помоћу And (И) и Or (ИЛИ) креирају се сложени услови.

Пример:

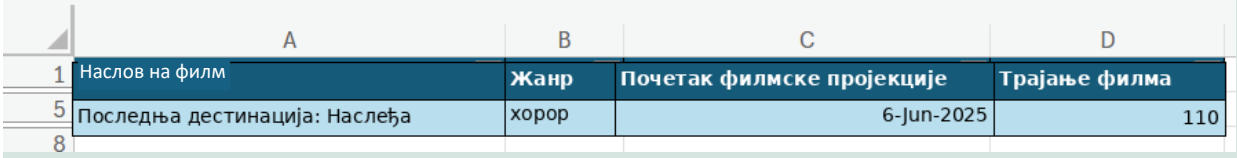
Филтрирање података у табели: Дата је табела са подацима: Наслов филма, жанр, датум пројекције, трајање филма. Прикажи називе филмова који почињу словом „П“.

	A	B	C	D
1	Наслов на филм	Жанр	Почетак филмске пројекције	Трајање филма
2	Тандерболти	акција	22-Apr-2025	126
3	Немогућа мисија: Коначна одмазда	акција	5-Jun-2025	169
4	Последња дестинација: Наслеђа	хорор	6-Jun-2025	110
5	Бунтовна принцеза	анимирани	16-Jun-2025	94
6	Лило и Стич	комедија	26-Jun-2025	108
7	Девојке са балкона	комедија	27-Jun-2025	104

Слика 33 *Табела за преглед на филмови*



Слика 34 *Прозор у којем су постављени захтеви за филтер*



	A	B	C	D
1	Наслов на филм	Жанр	Почетак филмске пројекције	Трајање филма
5	Последња дестинација: Наслеђа	хорор	6-Jun-2025	110
8				

Слика 35 *Филтрирана табела*

Уклањање филтера

1. Клика се на стрелицу у колони у којој је постављен филтер.

2. Са листе се бира Clear Filter From „име колоне".

3. Приказују се сви подаци.

Уклањање свих филтера у радном листу

4. Бира се Home > Sort & Filter > Clear.

ЗАКЉУЧАК

Помоћу филтера корисници могу да смање сложеност великих табела и да се усредсреде само на релевантне информације. То повећава ефикасност, олакшава рад и доприноси јаснијем представљању података и доношењу одлука. Ипак, приликом коришћења филтера треба водити рачуна да се не превиде важни подаци и да филтрирани подаци пружају целовиту слику.

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Шта представља филтрирање у Excel-у?

2. Како знамо да је филтер примењен на колону?

3. Који типови података могу да се филтрирају у Excel-у? (наведи бар 3)?

4. Шта се догађа са подацима који не испуњавају услове филтера – да ли се бришу?

5. Како можеш да филтрираш само редове у којима је вредност већа од 5 000?

6. Како можеш да уклониш све филтере који су примењени у табели?

7. Да ли можеш да филтрираш по више колона истовремено? Објасни како се то ради!

8. Када примениш филтер према тексту, које опције можеш да користиш (на пример: „почиње са“, „садржи“...)?

9. У табели је изнад 100 редова, имаш колону „Оцена“ (од 1 до 5). Како ћеш да филтрираш само редове са оценом 4 и 5?

10. Имаш колону „Цена“. Како ћеш приказати само производе са ценом између 1 000 и 3 000 динара?

11. Који проблеми могу да се појаве ако имаш празне редове или ћелије када примењујеш филтер?

12. Како можеш да комбинујеш филтер са сортирањем за бољу анализу података?

13. Зашто је важно да поставиш наслове колона пре коришћења филтера?

ПРАКТИЧНА ВЕЖБА:

1. Направи табелу „Потрошња тинејџера на омиљене производе“ са следећим називима колона: Име, Категорија, Производ, Цена (дин), Месец куповине и Оцена (1–5) и попуни је према слици.

	A	B	C	D	E	F
1	**Име**	**Категорија**	**Производ**	**Цена**	**Месец куповине**	**Оцена (1-5)**
2	Мила	Слушалице	JBL Tune 510BT	3,200.00 дин	Јануар	5
3	Алекс	Гејминг опрема	Razer Mouse Viper	4,500.00 дин	Фебруар	4
4	Сара	Паметни уређај	Xiaomi Smart Band 8	2,800.00 дин	Јануар	5
5	Мелек	Гејминг опрема	Xbox контролер	4,200.00 дин	Март	3
6	Хана	Слушалице	Sony WH-CH510	3,900.00 дин	Фебруар	4
7	Осман	Паметни уређај	Apple AirTag	1,800.00 дин	Март	5
8	Ана	Гејминг опрема	RGB Gaming Keyboard	5,500.00 дин	Јануар	5
9						

Слика 36 *Табела потрошње тинејџера на омиљене производе*

После сваке од наведених активности копирај филтрирану табелу у нови радни лист:

1) Филтрирај ученике који су купили слушалице.

2) Прикажи само производе са оценом 5.

3) Филтрирај производе купљене у фебруару.

4) Прикажи све производе из категорије „Гејминг опрема“.

5) Филтрирај производе који коштају више од 4 000 динара.

6) Допуни табелу сопственим подацима, примени различите филтере по имену, категорији, месецу или оцени и направи кратку анализу навика при куповини.

2. Креирај табелу „Активности тинејџера на друштвеним мрежама“ са колонама Име, Платформа, Тип активности, Време дневно (мин), Омиљени садржај и попуни је према слици.

	A	B	C	D	E
1	**Име**	**Платформа**	**Тип активности**	**Време дневно (мин)**	**Омиљени садржај**
2	Ива	Instagram	Објављивање приче	90	Мода
3	Марко	YouTube	Гледање видеа	120	Гејминг
4	Елена	TikTok	Креирање видеа	80	Плес
5	Марија	TikTok	Скроловање	60	Комедија
6	Стефан	Instagram	Лајковање и коментари	45	Фитнес
7	Ана	YouTube	Гледање видеа	100	Наука
8	Емин	facebook	Лајковање и коментари	30	Мода
9	Ајше	Instagram	Објављивање приче	80	Гејминг

Слика 37 *Табела активности тинејџера на друштвеним мрежама*

После сваке од наведених активности копирај филтрирану табелу у нови радни лист:

1) Допуни табелу сопственим подацима.

2) Филтрирај ученике који користе TikTok.

3) Прикажи само оне који на платформи проводе између 50 и 85 минута дневно.

4) Прикажи само оне који на платформи проводе више од 100 минута дневно.

5) Филтрирај по платформи „Instagram“ и активности „у причи“.

6) Искористи ову вежбу за дискусију о дигиталним навикама.

5.6 Креирање извештаја – пивот-табела

Пивот-табела је напредна алатка која омогућава брзо сумирање, груписање и анализу великих количина података. Уместо ручног бројања или израчунавања, пивот-табела омогућава да помоћу неколико кликова добијеш статистичке податке – колико је производа продато, у којем месецу је остварен највећи приход, колико је гласова добио сваки одговор у анкети итд.

Важно: Најпре треба да припремиш податке – у табелама не сме бити празних поља и свака колона треба да има назив.

	A	B	C	D	E	F	G	H	I
1	ID	Производ	Категорија	Количина	Јединична цена	Укупна продаја	Месец	Град	Запослени
2	1	Мајица	Одећа	15	800	12000	Јануар	Скопље	Марија Стојанова
3	2	Патике	Одећа	10	2500	25000	Февруар	Битољ	Александар Јанев
4	3	Телефон	Електроника	5	25000	125000	Јануар	Скопље	Иван Петров
5	4	Лаптоп	Електроника	2	45000	90000	Март	Охрид	Елена Наумовска
6	5	Торба	Додатак	7	3000	21000	Февруар	Битољ	Марија Стојанова
7	6	Слушалице	Електроника	12	1500	18000	Април	Тетово	Бојан Крстев
8	7	Јакна	Одећа	8	4000	32000	Март	Куманово	Александар Јанев
9	8	Рукавице	Додатак	3	900	2700	Март	Скопље	Велес
10	9	Ранац	Додатак	6	2200	13200	Април	Скопље	Елена Наумовска
11	10	Паметни сат	Електроника	3	7000	21000	Февруар	Тетово	Иван Петров
12	11	Наочаре	Додатак	4	2800	11200	Јануар	Битољ	Марија Стојанова
13	12	Мајица	Одећа	10	800	8000	Април	Куманово	Бојан Крстев
14	13	Патике	Одећа	9	2500	22500	Март	Скопље	Александар Јанев
15	14	Телефон	Електроника	6	24000	144000	Април	Битољ	Елена Наумовска
16	15	Лаптоп	Електроника	3	45000	135000	Јануар	Охрид	Иван Петров
17	16	Слушалице	Електроника	10	1500	15000	Февруар	Скопље	Бојан Крстев

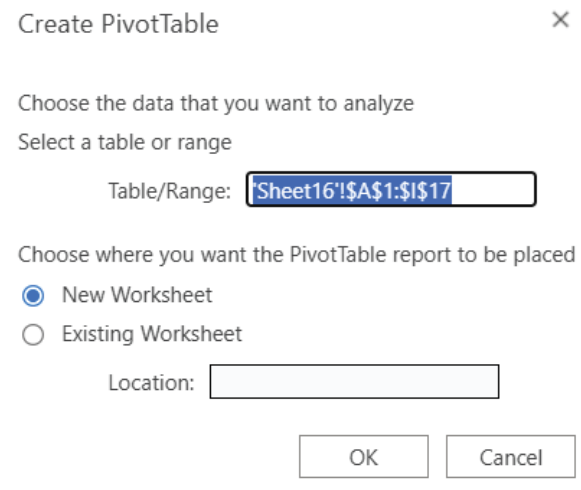
Слика 38 *Табела „Продажба“*

Кораци:

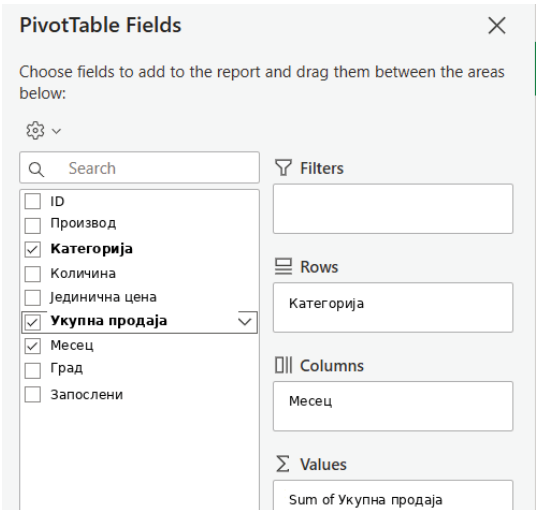
1. Кликне се било где у табели.

2. Се избира Insert > PivotTable.

3. Отвара се прозор PivotTable, где се бира опсег података (Table/Range), а затим и локација за нову табелу (New worksheet – нови радни лист (препорука), Existing Worksheet – постојећи радни лист). Избор се потврђује кликом на OK.



Слика 39 *Create PivotTable*



Слика 40 *Поља у пивот-табели из табеле „Продаја“*

4. Пивот-табела је још увек празна. Са десне стране приказани су називи колона и панел у који се поља могу превлачити у четири зоне:

5. Rows (Редови): Категорије по којима желиш да групишеш (на пример, категорија).

6. Columns (Колоне): За груписање по другој вредности (на пример, месец).

7. Values (Вредности): Шта желиш да израчунаш (збир, просек, број...).

8. Filters (Филтери): За филтрирање табеле према одређеним критеријумима.

За овај пример, уколико крајњи резултат треба да буде пивот-табела укупне продаје према категоријама и месецима, десни панел ће изгледати као на следећој слици:

Добијена пивот-табела има следећи облик:

3	Sum of Укупна продаја	Месец				
4	Категорија	Април	Јануар	Март	Фебруар	Grand Total
5	Додатак	13200	11200	2700	21000	48100
6	Електроника	162000	260000	90000	36000	548000
7	Одећа	8000	12000	54500	25000	99500
8	**Grand Total**	**183200**	**283200**	**147200**	**82000**	**695600**

Слика 41 *Пивот-табела од табела „Продажба“*

Да би ажурирао (освежио) пивот-табелу након измене података у изворној табели, један од начина је следећи: кликни десним тастером миша било где унутар пивот-табеле и у приказаном менију изабери Refresh (Освежи), након чега ће табела аутоматски преузети најновије податке из извора.

ЗАКЉУЧАК

Употреба пивот-табела је од суштинског значаја у процесу анализе података, посебно због њихове флексибилности, брзине и способности да обрађују велике количине информација без потребе за напредним формулама. Ипак, њихова ефикасност зависи од квалитета почетних података и од способности корисника да постави праве параметре. Препоручује се да свака анализа помоћу пивот-табеле почне добро припремљеним и чистим подацима, коришћењем смислених категорија за редове и колоне, као и јасним циљем анализе.

Пивот-табеле значајно побољшавају способност доношења одлука у пословном контексту представљањем информација на сажет и аналитички начин. Без обзира на то да ли се користе у финансијској анализи, праћењу трошкова, анализи задовољства клијената или оперативном планирању, оне доприносе информисаном одлучивању заснованом на подацима.

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. За шта се најчешће користе пивот-табеле?

2. Који типови података су погодни за анализу помоћу пивот-табела?

3. Зашто је важно да подаци имају наслове колона пре креирања пивот-табеле?

4. Које функције за прорачуне могу да се користе у пивот-табели (на пример, збир, просек)?

5. Шта ће се догодити са пивот-табелом ако се промене подаци у изворној табели?

6. Како се ажурира (refresh) пивот-табела након промене података?

7. Које су неке од најчешћих грешака при раду са пивот-табелама?

ПРАКТИЧНА ВЕЖБА:

- У твојој школи организују се бројне активности – сваког месеца одржавају се радионице, такмичења, спортски дани и креативни пројекти. Неки ученици се такмиче, други стичу нове вештине, док трећи помажу у организацији. Наставничко веће жели боље да разуме укљученост ученика и ефекте ових активности. Добијаш задатак да помоћу пивот-табеле у програму Excel анализираш одржане догађаје.

Изворна табела коју ћеш користити садржи следеће колоне:

- Име ученика
- Одељење
- Тип догађаја – Радионица, такмичење, спортски дан, презентација итд.
- Датум одржавања – Када се догађај одржао
- Трајање (у минутима) – Колико је трајала активност

Помоћу пивот-табеле треба одговорити на питања као што су:

а) Који тип догађаја је најчешћи?
б) Која одељења су најактивнија?
в) Колико у просеку трају догађаји?
г) Који ученици имају највише учешћа?

Предлог за изворну табелу:

	A	B	C	D	E
1	**Име ученика**	**Одељење**	**Тип догађаја**	**Датум одржавања**	**Трајање (у минутима)**
2	Ана Петрова	VIII-1	Такмичење	12.03.2024	90
3	Јован Стојков	IX-2	Радионица	15.03.2024	120
4	Ева Милошевска	VIII-2	Спортски дан	20.03.2024	60
5	Петар Јованов	IX-2	Радионица	25.03.2024	90
6	Ана Петрова	VIII-1	Такмичење	05.04.2024	95
7	Ева Милошевска	VIII-2	Презентација	10.04.2024	45
8	Јован Стојков	IX-2	Такмичење	12.04.2024	100

Слика 42 *Табела „Ученички активности“*

Решења за самопроверу

а) **Rows:** Тип догађаја
Values: Count of Тип догађаја
б) **Rows:** Одељење
Values: Count of Име ученика
в) **Rows:** Тип догађаја (или Одељење)
Values: Average of Трајање (у минутима)
г) **Rows:** Име ученика
Values: Count of Тип догађаја

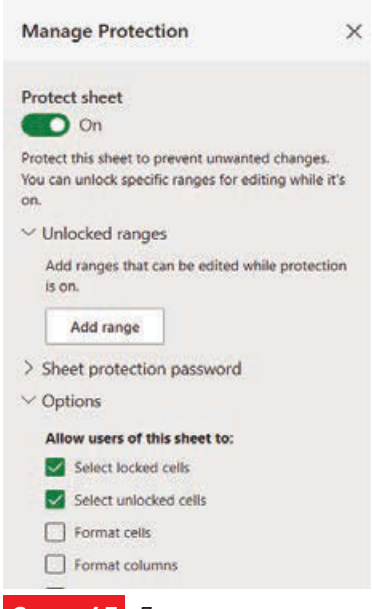
НАПОМЕНА:

Да би променио функцију у пољу Values (на пример: Sum, Count, Average), кликни на стрелицу десно од њеног назива, изабери Value Field Settings..., а затим жељену функцију.

5.7 Заштита података

У стварном раду са табелама често се уносе важни подаци – било да су то оцене, финансијски извештаји, распореди или планови. Али шта ако неко грешком избрише формулу? Или ако унесе неважећу вредност, као што је текст уместо броја? Да би се спречиле такве грешке, постоје алатке за заштиту и контролу података.

У програму Excel 365 функција Protect користи се за заштиту радних листова, ћелија или целог документа од неовлашћених измена.



Слика 43 *Део панела Manage Protection*

Заштита целог листа

Кораци:

1. Изабери Review.

2. Кликни на Protection -> Manage Protection.

3. Унеси лозинку (опционо).

4. Изабери шта корисници могу да мењају.
(на пример, форматирање, селектовање ћелија).

Важно: Када се Excel документ извезе у други формат, као што су PDF, CSV или HTML, заштита постављена у програму Excel не преноси се у извезени документ. Ако желиш да документ остане заштићен, треба да га сачуваш у Excel формату (.xlsx). Ако се документ налази на сервису OneDrive, изабери: File → Save a Copy.

НАПОМЕНА: Уколико радиш у онлајн верзији програма Excel 365, опције за заштиту се донекле разликују од оних у десктоп верзији. Овде можеш да заштитиш цео радни лист или да ограничиш уређивање помоћу опције Manage Protection. У оквиру ње налази се опција Add range, помоћу које можеш да одредиш који делови радног листа могу да се мењају док је лист закључан. У програму Excel све ћелије су подразумевано „закључане“, али то нема ефекта док се не укључи заштита радног листа. Зато, ако желиш да закључаш само одређене ћелије, најпре треба да „откључаш“ све ћелије, а затим да „закључаш“ само оне које су ти потребне.

Откључавање свих ћелија

Кораци:

1. Селектуј све ћелије (Ctrl+A).

2. Десен клик → Format Cells → Protection tab.

3. Искључи опцију Locked.

Закључавање ћелија

1. Селектуј ћелије за које желиш да не могу да се мењају.
2. Десни клик → Format Cells → Protection таб.
3. Одабери поље Locked. (обележи га).
4. Иди на Review → Protect Sheet и активирај заштиту.

Сада све ћелије које си означио/ла као „Locked" не могу да се мењају, док су остале слободне за уређивање!

НАПОМЕНА: Ако желиш да сакријеш формуле и функције, поступак је исти као код закључавања ћелија, али уместо Locked треба да изабереш Hide.

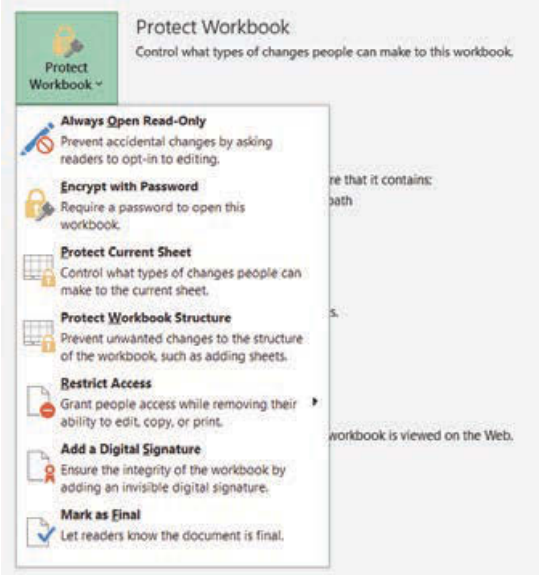
Заштита целе радне свеске

Кораци:

1. Иди на File → Info.

2. Кликни на Protect Workbook → Encrypt with Password.

3. Унеси лозинку и потврди.



Слика 44 *Прозор protect workbook*

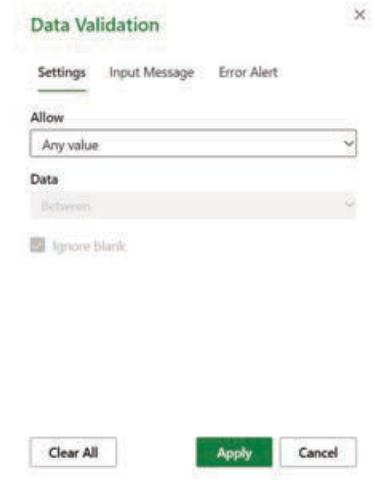
Валидација података

Замисли да креираш табелу у Excel-у у коју ученици треба да унесу оцене од 1 до 5. Али, један ученик је грешком унео 7, а други текст уместо броја. Такве грешке могу да доведу до погрешних резултата у прорачунима. Управо ту валидација података представља решење.

Валидација података омогућава постављање правила о томе које вредности смеју да се унесу у одређене ћелије. На пример, можеш да дозволиш само бројеве између 1 и 10, само датуме после данашњег датума или да направиш падајућу листу са унапред дефинисаним вредностима. На тај начин се избегавају грешке, контролише тачност података и олакшава обрада информација.

Валидација података активира се избором Data > Data validation.

У Allow одређујеш какав тип података је дозвољено унети у изабрану ћелију или опсег ћелија. Дозвољене вредности су:



Слика 45 *Прозор Data validation*

Опција	Објашњење
Any value	Дозвољава било шта (нема ограничења).
Whole number	Дозвољава само целе бројеве (без децимала).
Decimal	Дозвољава бројеве са децималама.
List	Дозвољава избор из унапред дефинисане листе.
Date	Дозвољава унос само датума.
Time	Дозвољава унос само времена.
Text length	Ограничава колико карактера може да се унесе.
Custom	Дозвољава постављање сопственог правила са формулом.

Постављање валидације (пример са целим бројем)

Кораци:

1. Означи ћелије где желиш да се унесе цео број између 1 и 5.

2. Иде у Data > Data Validation.

3. Испод „Allow“ изабери Whole number.

4. У прозору Data validation у Data одабери between,

5. у Minimum напиши 1, а у поље Maximum напиши 5.

6. Потврди избор кликом на Apply.

Сада, уколико покушаш да у ове ћелије унесеш број другачији од дефинисаног опсега (од 1 до 5), појавиће се порука за грешку.

Слика 46 *Прозор Data validation за цео број*

Постављање валидације (пример са листом, са директним уношењем вредности)

Замисли да креираш табелу у којој ученици треба да изаберу један од неколико унапред дефинисаних одговора, као што су „Да“, „Не“ или „Можда“. Уместо да свако уноси одговор на свој начин, што може довести до грешака, можеш унапред да им понудиш избор из падајућег менија.

Кораци:

1. Изабери ћелију или опсег где желиш да додаш падајућу листу.

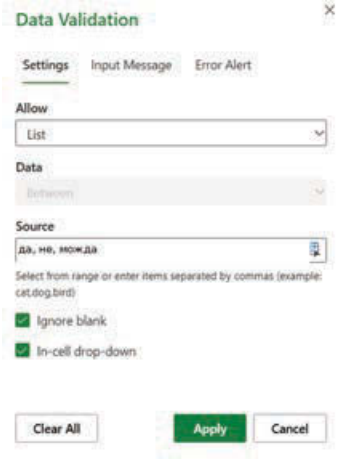
2. Изабери Data > Data Validation.

3. У прозорче које ће се отворити:

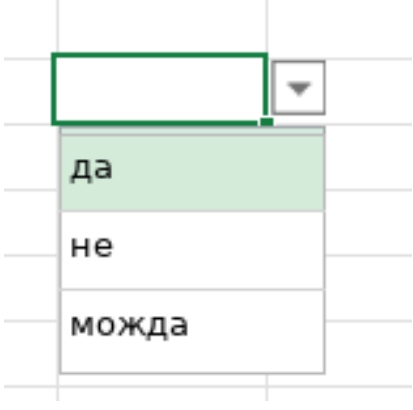
- У поље Allow (Дозволи) изабери: List (Листа).
- У поље Source (Извор), директно унеси опције, подељене запетама.

4. Потврди избор (Apply).

Сада ће се у изабраном пољу (или опсегу), на десној ивици ћелија, појавити стрелица. Кликом на њу отвара се падајућа листа са које треба да изабереш једну од понуђених опција.



Слика 47 *Прозор Data validation за падајућу листу*



Слика 48 *Ћелија са падајућим менијем*

НАПОМЕНА:

У прозору data Validation има још два језичка:

- Input Message служи за писање информацијске поруке која се појављује када корисник кликне на ћелију која има постављену валидацију података.
- Error Alert где можеш да напишеш какву поруку желиш да се појави када неко унесе вредност која није дозвољена према правилима валидације.

ЗАКЉУЧАК

Валидација података је практична и моћна алатка која омогућава контролу уноса података у табеле. Помоћу ње може се ограничити шта се сме унети у одређене ћелије – на пример, само бројеви, датуми, текст или вредности са падајуће листе. Ово је посебно важно када се ради са већим табелама или када се документи деле са другим особама, јер помаже у спречавању грешака и очувању тачности података.

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Шта значи закључивање ћелија у Excel и да ли оне одмах постају заштићене?

2. Који је први корак који треба да урадиш ако желиш да дозволиш уређивање само неких ћелија?

3. Који је други корак после закључивања ћелија да би се активирала заштита?

4. Како се заштићује један радни лист у Excel 365?

5. У чему је разлика између „заштите радног листа“ и „заштите радне свеске“?

6. Које опције можеш да омогућиш или онемогућиш при заштити листа?

7. Може ли да се дозволи другим корисницима да унесу податке само у одређене ћелије?

8. Да ли је обавезно да се постави лозинка при заштити радног листа или радне свеске?

9. Шта се догађа ако покушаш да уређујеш закључану ћелију у заштићеном листу?

10. Како се уклања (искључује) заштита радног листа или радне свеске?

11. Шта представља валидација података?

12. Који је циљ коришћења валидације података?

13. Које су могућности у падајућем менију „Allow“ у прозору за валидацију?

14. Шта треба да садржи поље „Source“ при уношењу листе вредности?

15. Шта се догађа ако неко унесе вредност која није дозвољена према валидацији?

16. Да ли можеш да примениш исту валидацију на више ћелија одједном? Како?

ПРАКТИЧАН ЗАДАТАК

1. Имаш Excel табелу са следећим подацима

Име	Презиме	Оцена 1	Оцена 2	Укупно
Марко	Јанев	5	4	9
Елвир	Баку	4	5	9
Ивица	Стојанов	3	4	7

- Закључи ћелије са изузетком колона „Оцена 1“ и „Оцена 2“!
- 2. У табели са производима попуни одговарајуће празна поља, користећи формуле које ћеш сакрити!

Производ	Цена по комаду	Количина	Укупно
Хлеб	40	2	
Млеко	60	3	
Сок	90	1	
Шећер	50	2	
УКУПНО			

3. У колони B (ћелије B2 до B10), направи падајућу листу са следећим опцијама: I година, II година, III година, IV година!

4. У колону C унеси узраст ученика! Дозволи унос само целим бројевима од 14 до 19.

5. У колони D унеси датум уписа. Дозволи унос само датума од 01.09.2024 до 30.09.2025.

ПОНАВЉАЊЕ ЗА ТЕМУ 5

1. Наведи три основне формуле или функције које се користе у Excel за израчунавање података (на пример: SUM, AVERAGE, IF)!

2. Објасни шта значи појам „релативно адресирање“ и како се разликује од „апсолутног адресирања“ у Excel!

4. Креирај табелу са именима ученика и њиховим оценама! Примени формулу за рачунање просека за сваког ученика!

5. У истој табели примени условно форматирање да црвеном бојом означиш све оцене ниже од 2,50!

6. Имаш базу података са колонама: Име, Предмет, Оцена, Датум. Филтрирај само ученике са оценом ≥ 4 и објасни како си добио резултате!

7. Анализирај који тип графикона (линијски, стубичасти или пита-дијаграм) је најбољи за приказ промене оцене током месеца. Образложи избор!

8. Дата је табела са трошковима и приходима. Процени да ли је функција IF најбољи избор за проверу да ли је буџет позитиван или негативан! Објасни своју одлуку!

9. Имаш табелу са важним финансијским подацима. Одлучи који тип заштите је најпотребнији: закључавање ћелија, заштита листа или заштита целе радне свеске! Образложи!

10. Креирај пивот-табелу из базе података са најмање 20 записа (на пример: продаја, оцене, трошкови)! Организуј је тако да приказује укупне суме по категоријама!

11. Направи свој извештај који ће садржати:

- табела као база података,
- Прорачуни са најмање 3 напредне функције (IF, VLOOKUP/XLOOKUP, COUNTIF),
- графикон по избору,
- филтри за брзо претраживање,
- пивот-табела,
- основна заштита најважнијих ћелија.