Рачунарство и информатика
оба типа гимназије

Циљ и задаци

Циљ наставног предмета рачунарство и информатика је стицање знања, овладавање вештинама и формирање вредносних ставова који доприносе развоју информатичке писмености неопходне за даље школовање, живот и рад у савременом друштву, као и оспособљавање ученика да ефикасно и рационално користе рачунаре на начин који не угрожава њихово физичко и ментално здравље.

Задаци наставе рачунарство и информатика су да ученици:

- развију свест о неопходности коришћења рачунара у свакодневном животу и раду и значају информатике за функционисање и развој друштва;
- овладају коришћењем програма за обраду текста и табеларних података и креирање докумената у коме су интегрисани текст, слика и табела;
- ефикасно користе програмски језик заснован на прозорима за решавање различитих проблема у даљем образовању, професионалном раду и свакодневном животу;
- стекну знања потребна за подешавање параметара оперативног система на нивоу корисничког интерфејса, коришћење могућности оперативних система и система датотека конкретног оперативног система;
- разумеју принципе функционисања интернета, локалних мрежа и оспособе се за коришћење мрежних ресурса, интернет сервиса и система за електронско учење;
- јачају способност за прецизно и концизно дефинисање проблема; упознају се са алгоритамским начином решавања проблема и основним алгоритмима;
- развију способности писања програма вођених догађајима и разумеју принципе креирања модуларних и добро структуираних програма;
- упознају основни концепт и принципе Веб дизајна и Веб програмирања, разумеју логику анимације и овладају њеном употребом у креирању сопствених Веб пројеката;
- упознају основне концепт и принципе различитих парадигми програмирања (процедурално, објектно, функционално, логичко)
- упознају принципе представљања и обраде цртежа и слика на рачунару и овладају техникама коришћења једног од графичких програма за обраду цртежа и слика;
- упознају начине израде презентација и оспособе се за израду једноставнијих презентација;
- упознају концепт базе података, њену организацију, коришћење упита за добијање тражених података из базе, прављење извештаја и дистрибуцију података;
- јачају способност решавања проблема развојем логичког и критичког мишљења;
- унапреде способности за брзо, ефикасно и рационално проналажење информација коришћењем рачунара, као и њихово критичко анализирање, складиштење и преношење;

- развију прецизност, рационалност и креативност у раду са рачунаром;
- унапреде стратегије и технике самосталног учења користећи могућности рачунара и развију спремност за учење током целог живота;
- на адекватан начин користе предности рачунара и друштвених мрежа у удруживању са другима и покретању акција чији је циљ ширење корисних информација или пружање помоћи и подршке онима којима је то потребно;
- примене стечена знања и вештине у савладавању програма других наставних предмета;
- изграде правилне ставове према коришћењу рачунара, без злоупотребе и претеривања које угрожава њихов физичко и ментално здравље;
- упознају савремена ергономска решења која олакшавају употребу рачунара и изграде спремност за праћење нових решења у области информатичке технологије.

Први разред,

(0+2 часа недељно, 74 часа годишње)

САДРЖАЈИ ПРОГРАМА

1. **Основи информатике (10)**
   - Основни појмови информатике и рачунарства (информације, подаци, информационо-комуникационе технологије, предмет и области информатике и рачунарства).
   - Представљање информација у дигиталним уређајима (дигитализација, представљање слика, звука, текста, бројевни системи, запис бројева, јединице за мерење количине информација).
   - Развој информационих технологија (прикупљања, складиштења, обраде, приказивања и преноса података).
   - Утицај савремене рачунарске технологије на живот људи (значај и примена рачунара, карактеристике информационог друштва, утицај коришћења рачунара на здравље, безбедност и приватност).

2. **Архитектура рачунарског система (6)**
   - Структура и принцип рада рачунара.
   - Процесор.
   - Меморијска хијерархија.
   - Улазно-излазни уређаји.
   - Магистрале.
   - Савремена хардверска технологија.
   - Утицај компоненти на перформансе рачунара.

3. **Програмска подршка рачунара (4)**
   - Апликативни софтвер.
   - Системски софтвер.

- Оперативни систем.
- Дистрибуција софтвера (верзије и модификације програма, власништво над софтвером, заштита права на интелектуалну својину).

4. Основе рада у оперативном систему са графичким интерфејсом (6)

- Основни елементи ГКИ и интеракција са њима (радна површина, прозори, менији, дугмад, акције мишем, пречице на тастатури, ...).
- Основна подешавања оперативног система (подешавање датума и времена, радне површине, регионална подешавања, подешавања језика и тастатуре, коришћење и подешавање корисничких налога).
- Инсталирање и уклањање програма (апликативних програма, драјвера).
- Рад са документима и системом датотека.
- Средства и методе заштите рачунара и информација.

5. Обрада текста уз помоћ рачунара (12)

- Приступи уносу и обради текста (текст-едитори, језици за обележавање, текст-процесори).
- Радно окружење текст-процесора и његово подешавање.
- Унос текста и његово једноставно уређивање (ефикасно кретање кроз текст, копирање, премештање, претрага, замена текста).
- Форматирање и обликовање текста (странице, пасуса, карактера).
- Посебни елементи у тексту (листе, табеле, слике, математичке формуле, ...).
- Коришћење и израда стилова, генерисање садржаја.
- Алатке интегрисане у текст-процесоре (провера граматике и правописа, редиговање текста, библиографске референце, индекс појмова, циркуларна писама, ...).
- Коришћење готових шаблона и израда сопствених шаблона.
- Штампа докумената.

6. Слајд-презентације (6)

- Презентације и њихова примена (правила добре презентације, етапе у изради презентација).
- Радно окружење програма за израду слајд-презентација и његово подешавање (погледи на презентацију).
- Креирање слајдова (уметање и форматирање текста, графикона, слика, звучних и видео-записа, ...).
- Складно форматирање слајдова (мастер слајд).
- Анимације (анимације објеката на слајдовима, анимације прелаза између слајдова, аутоматски прелазак између слајдова и снимање нарације).
- Интерактивне презентације (хипервезе, акциона дугмад).
- Штампање презентације.

7. **Увод у рачунарске мреже и интернет (8)**
   - Појам и врсте рачунарских мрежа.
   - Организација уређаја у мрежи (мрежна топологија, клијенти, сервери, p2p).
   - Мрежни слојеви, мрежни хардвер и мрежни софтвер.
   - Глобална мрежа (интернет).
   - Интернет-провајдери и технологије приступа интернету.
   - Сервиси интернета (електронска пошта, веб, друштвене мреже, блогови, форуми, учење, мапе, електронска трговина и банкарство, аудио и видео комуникација).
   - Лепо понашање, право и етика на интернету. Безбедност и приватност на интернету.

8. **Увод у алгоритме (20)**
   - Зашто учити алгоритме и програмирање?
   - Појам процедуралног алгоритма, структура алгоритма и начини описивања алгоритама.
   - Примери алгоритама.
   - Визуелно (блоковско) програмирање.

Наведени фонд часова обухвата обраду новог градива, утврђивање и обнављање и провере знања.

Упутство за реализацију

Настава се изводи у двочасу, са половином одељења у рачунарском кабинету.

На почетку наставе урадити проверу нивоа знања и вештина ученика, која треба да послужи као оријентир за организацију и евентуалну индивидуализацију наставе.

У уводном делу двочаса наставник истиче циљ и задатке одговарајуће наставне јединице, затим реализује теоријски део неопходан за рад ученика на рачунарима. Уводни део двочаса, у зависности од садржаја наставне јединице, не би требало да траје више од 30 минута. Након тога организовати активност која, у зависности од теме, подстиче изградњу знања, анализу, критичко мишљење, интердисциплинарно повезивање. Активност треба да укључује практичан рад, примену ИКТ, повезивање и примену садржаја различитих наставних предмета, тема и области са којима се сусрећу изван школе. Активности осмислити тако да повећавају мотивацију за учење и

подстичу формирање ставова, уверења и система вредности у вези са развојем језичке и информатичке писмености, здравим стиловима живота, развојем креативности, способности вредновања и самовредновања.

При реализацији програма дати предност пројектној, проблемској и активној настави, кооперативном учењу, изградњи знања и развоју критичког мишљења. Подстицати тимски рад и сарадњу нарочито у областима где наставник процени да су присутне велике разлике у предзнању код појединих ученика. Уколико услови дозвољавају дати ученицима подршку хибридним моделом наставе (комбинацијом традиционалне наставе и електронски подржаног учења), поготово у случајевима када је због разлика у предзнању потребна већа индивидуализација наставе.

При реализацији тематске целине **Основи информатике** ученици би требало да се упознају са предметом изучавања информатике и рачунарства, са основним појмовима којима се ове области баве (појам информација, података, знања) и са основним областима информатике и рачунарства.

Потребно је затим увести појам дигитализације (дискретизације), објаснити како се у дигиталним уређајима све информације представљају (кодирају) помоћу бројева и продискутовати предности дигиталног у односу на аналогни запис. Ученици би требало да стекну представу о томе како се кодирају текстуалне, графичке и звучне информације. Увести појам бројевних основа (пре свега бинарне, декадне, хексадекадне и окталне) и приказати како је могуће број записати у некој од наведених бројевних основа и прочитати број записан у некој основни (уз помоћ дигитрона, али и без њега). Ученици треба да усвоје појмове бит, бајт, и редове величине за мерење количине информација.

Развој информационих технологија сагледати у контексту значаја развоја ових технологија за развој и ширење писмености и развој људског друштва уопште. Подстаћи ученике да повезују развој ИКТ-а са темама из историје, математике, физике и осталим областима људске делатности. Из овог угла сагледати значај информатике, области примене рачунара (и њихов развој) и карактеристике информационог друштва. Не инсистирати на прецизном познавању чињеница (не анализирати детаљно перформансе одређених рачунара, не инсистирати на тачном познавању година увођења одређених технологија), већ ову причу учинити што занимљивијом и пријемчивијом ученицима.

Скренути пажњу ученицима на друштвене промене које је изазвала информациона револуција и на позитивне промене које су ИКТ донеле у све сегменте људског живота. Развити код ученика свест и о опасностима и неопходним мерама заштите здравља од претеране и неправилне употребе рачунара као и о питањима безбедности и приватности приликом употребе уређаја, нарочито у данас неизбежном мрежном окружењу.

При реализацији тематске целине **Архитектура рачунарског система** потребно је да ученици стекну знања о структури (хардверу) и принципу рада рачунара (али без упуштања у детаље техничке реализације, електронске схеме и слично), функцији хардверских компоненти рачунара и њиховом утицају на перформансе рачунара. Ученици би требало да знају да наведу и практично препознају из којих се компоненти састоји стони или преносни рачунар. При том, прва, површна класификација разликује кућиште, монитор, тастатуру, миш, штампач, док друга, из стручне перспективе много важнија, разликује процесор, меморије, улазно-излазне уређаје и магистрале које их повезују. Ученици би требало да умеју да објасне улогу процесора у функционисању рачунарског система (да познају појам радног такта, да разумеју улогу регистара, аритметичко-логичке и контролне јединице, да буду упознати са појмом процесорских инструкција и свођењу комплексних операција на низ елементарних инструкција), да објасне врсте и улогу различитих меморија у рачунарима и да разликују унутрашње меморије (кеш, RAM) од спољашњих, складишних меморија (хард-дискова, флеш-меморија, SSD уређаја, оптичких дискова). Инсистирати на хијерархијској организацији меморија и објаснити разлику у брзини, капацитету и цени различитих облика меморија. Објаснити основне врсте улазно-излазних уређаја и начине комуникације са њима. Описати и различите врсте магистрала и њихову улогу у остваривању комуникације између различитих компонената унутар рачунара.

Са ученицима заједно продискутовати карактеристике у том тренутку актуелне хардверске технологије (на пример, анализирати детаље хардверских конфигурација које се описују у огласима за продају рачунара). Ученици могу анализирати конфигурације школских рачунара (уз помоћ података доступних из оперативног система) и за домаћи им је могуће задати да анализирају конфигурације својих кућних рачунара. Ученицима је могуће приказати и поступак расклапања и склапања рачунара и указати им на једноставне кварове које могу сами препознати и отклонити.

Ученицима је могуће приказати и архитектуру и хардверске компоненте савремених мобилних уређаја (таблета, паметних телефона).

При реализацији тематске целине **Програмска подршка рачунара** потребно је да ученици стекну знања о значају програмске подршке (софтвера) за функционисање рачунара и утицају на могућности рачунара. Са ученицима продискутовати софтвер који они свакодневно користе и на основу тога формирати класификацију апликативног софтвера (на пример, софтвер за приступ услугама интернета, канцеларијски софтвер, софтвер за креирање и обраду мултимедијалних садржаја, рачунарске игре, ...). Увести затим појам системског софтвера и ученицима разјаснити појам и улоге оперативног система (ОС) из мало дубље перспективе него што је то само коришћење интерфејса ОС за основно управљање радом за рачунаром (не упуштати се у напредне детаље рада ОС, попут детаља распоређивања процесора, организације виртуелне меморије и слично). Описати и приказати и услужни софтвер (на пример, антивирусне програме, заштитне зидове и слично).

Ученицима скренути пажњу на појам власништва над софтвером, софтверских лиценци и заштите ауторских права. Описати разлику између власничког и слободног софтвера и софтвера отвореног кода. Описати и различите облике дистрибуције софтвера (пробне верзије, делимичне верзије). Ученицима (и на личном примеру) развијати правну и етичку свест о ауторским правима над софтвером, али и над подацима који се дистрибуирају путем мреже. Посебну пажњу посветити потреби коришћења лиценцираних програма, заштити програма и података, вирусима и заштити од њих.

При реализацији тематске целине **Основе рада у оперативном систему са графичким интерфејсом** ученик треба да стекне знања, вештине и навике битне за успешно коришћење основних могућности оперативног система. Може се претпоставити да ученици већ умеју покрећу и користе апликативне програме (на пример, програме који су у саставу оперативног система за приказ мултимедијалних садржаја, уређење текста, цртање и једноставна нумеричка израчунавања, али и друге програме инсталиране на рачунар). Кроз неколико примера приказати поступак инсталације и уклањања апликативних програма, али и управљачких програма (драјвера) за одређене уређаје. Ученик би требало да уме да подеси основне параметре оперативног система (изглед окружења, датум и време, регионална и језичка

подешавања, укључивање и искључивање приказа скривених датотека, честих екстензија датотека, подешавање подразумеваног програма за рад са одређеним типом датотека и слично).

Детаљно приказати рад са системом датотека (фајлова, докумената). Ученик треба да разликује намену датотека и директоријума (фасцикли, фолдера, каталога) и да познаје намену типова датотека који се најчешће користе (.txt, .jpeg, .avi, .docx, ...), да уме да из програма, коришћењем стандардних дијалога, учита и сними своје документе у одабраном формату на жељене локације, да пребаци документе са једног на други уређај или партицију диска, да хијерархијски организе своје документе коришћењем директоријума, да разликује логички и физички поглед на систем датотека (на пример, да познају положај фасцикле Desktop или Documents у вишекорисничком окружењу, да умеју да користе пречице и библиотеке), да примењује технике архивирања и компресије података, да уме да изврши основне операције са системом датотека из командне линије оперативног система (промени текући директоријум, прегледа његов садржај, копира, обрише или премести одређене докуемнте и слично).

Скренути пажњу на честе концепте који се јављају током рада у окружењима са ГКИ и на нивоу оперативног система и на нивоу различитих апликативних програма (на пример, приказати концепт селекције и начине селектовања мишем и тастатуром, концепт клиборда и примену на копирање и пребацивање података). Ученик треба да зна да прати и да одреагује на најчешће поруке оперативног система (при брисању датотека и каталога; при затварању програма, а да није претходно сачуван документ, итд.). Пажњу посветити и питањима заштите (подешавања антивирусног програма и заштитног зида).

Неки елементи ове тематске целине се могу прожимати са другим тематским целинама. На пример, програм калкулатор (који се налази у оквиру оперативног система) се може користити када се уче бројевни системи, структура и перформансе конкретног рачунара се могу сагледати коришћењем података о уређајима добијених од оперативног система итд.

При реализацији тематске целине **Обрада текста на рачунару** потребно је да ученик стекне знања, вештине и навике неопходне за успешно коришћење програма за обраду текста. Ученицима приказати неколико различитих приступа обради текста (WYSIWYG приступ и језике за обележавање какви су LaTeX или HTML). Објаснити разлику између чистих текстуелних докумената креираних у текст-едиторима (.txt

документи, обележени текстови, изворни кодови програма) и форматираних текстуалних докумената креираних у текст-процесорима. Нагласити да су основни кораци у раду са текстом (уношење текста, кретање кроз текст, копирање, исецање и премештање делова текста, претрага и замена) заједнички за широку класу програма који раде са текстом (текст-едиторе, текст-процесоре, разне апликативне програме, уобичајене контроле за унос текста). Инсистирати да ученици умеју вешто и ефикасно врше основне операције са текстом, коришћењем само тастатуре (да се крећу кроз текст карактер по карактер, реч по реч, пасус по пасус, да користе тастере Home и End, да селектују текст помоћу тастера Shift и кретања кроз текст, користе пречице за копирање, исецање и лепљење и слично). Скренути пажњу ученицима на вештину слепог куцања и мотивисати их да у самосталном раду савладају ту вештину.

Обучити ученике за рад у једном конкретном процесору текста. Ученик треба да уме да подеси радно окружење текст процесора, унесе текст (у ћириличком и латиничком писму), сачува унети текст, отвори постојећи текстуални документ, затвори активни документ, премешта садржај између више отворених докумената. Ученик треба да уме да изврши основно форматирање текста (да подеси димензије странице, маргине, да подеси својства пасуса, фонтове, својства карактера и слично). Ученик треба да уме да текст организује коришћењем листа (нумерисаних и ненумерисаних), да у текст уметне и форматира табеле, да организује текст у секције и сложи га у више колона, да уметне у текст и исправно позиционира специјалне симболе, датум и време, слике, дијаграме, формуле, итд.

Пре преласка на рад са дужим и комплекснијим документима, потребно је објаснити разлику између логичке структуре докумената и њиховог визуелног и стилског обликовања и форматирања и увести стилове као основну технику логичког структурирања докумената. Инсистирати на томе да у свим дужим документима морају бити коришћени стилови (постојећи и кориснички дефинисани). У сложеније документе ученик треба да уме да уметне аутоматску нумерацију страница, да подеси подножја и заглавља страница, да аутоматски генерише садржај, индекс појмова, списак библиографских референци и слично. Ученике треба упознати са логичком структуром типичних докумената (молби, обавештења, итд.), школских реферата, семинарских и матурских радова и у свим вежбањима потребно је користити документе какви се срећу у реалном животу и инсистирати на њиховој униформности и прегледности, а не на усиљеним естетским подешавањима (избегавати документе који немају смислен садржај и који служе само да прикажу што више различитих могућности текст-процесора). За вежбу се може од ученика тражити да неформатирани

дужи текст форматирају на основу датог узора (на пример, на основу датог документа у PDF формату).

Ученицима приказати алат за исправљање граматичких и правописних грешака, за коментарисање и обележавања измена у тексту и слично.

Ученик треба да уме да прегледа текстуални документ пре штампе, подешава параметре

за штампу и штампа.

На крају, ученицима је могуће приказати и рад у неколико различитих програма за обраду текста (нарочито, ако је неки од њих "у облаку") и нагласити сличност са текст-процесором коришћеним у претходном току наставе.

При реализацији тематске целине **Слајд-презентације** потребно је да ученици схвате предности коришћења слајд-презентација у различитим ситуацијама, препознају ситуације у којима се може користити слајд-презентација, планирају и израђују адекватне презентације. При томе је потребно да знају основне етапе при развоју слајд-презентације, основне принципе доброг дизајна презентације (број информација по слајду, естетика, анимација у служби садржаја).

Ученике треба обучити коришћењу бар једног програма за креирање слајд презентација. Ученик треба да уме да подеси радно окружење, бира одговарајући поглед на презентацију, креира слајдове, поставља на њих текст и нетекстуалне објекте (слике, табеле, графиконе) доследно их форматира (користи мастер слајд). Ученик треба да уме да креира и интерактивне презентације које садрже линкове и акциону дугмад, да подешава анимације објеката на слајдовима и анимације преласка између слајдова, али те анимације треба да буду једино у функцији садржаја (избегавати анимације "по сваку цену" које оптерећују презентацију).

Примери презентација треба да буду смислени, из реалног живота (најбоље је да се користе слајд-презентације у којима се обрађују теме из наставе, како информатике и рачунарства, тако и других предмета). Ученици неке презентације могу да креирају и у склопу домаћих задатака, а на часу је могуће анализирати презентације направљене код куће.

На крају, ученицима је могуће приказати још неколико програма за креирање слајд-презентација (нарочито, оне "у облаку") и подвући сличности са програмом који је коришћен током наставе.

При реализацији тематске целине **Увод у рачунарске мреже и интернет** потребно је да ученици разумеју основне принципе по којима функционишу рачунарске мреже и интернет и систематизују своје познавање и коришћење различитих сервиса које савремени интернет и веб нуде.

Ученик треба да буде упознат са различитим врстама рачунарских мрежа (локалне мреже, мреже широког распона), њиховом топологијом и организацијом (клијент-сервер, p2p2). Ученицима приказати мрежни хардвер (рутере, свичеве, хабове, мостове, модеме, ...) и комуникациону опрему (каблове и технике бежичне комуникације). Неформално објаснити значај слојевитости мрежа и описати основне мрежне слојеве и протоколе. Навести и основне врсте мрежног софтвера. Полазна тачка при упознавању локалних мрежа треба да буде конкретна школска мрежа, као и мале кућне мреже. Детаљни преглед рачунарских мрежа предвиђен је током наставе у четвртом разреду, тако да у овом тренутку преглед рачунарских мрежа може да буде само информативног карактера.

Ученици од раније умеју да користе прегледаче веба (читаче, браузере) да приступе различитим веб страницама и претраживаче веба да потраже странице које их занимају. На одређеном скупу примера приказати добре стратегије за претрагу интернета и одабир релевантних страница из приказаних резултата претраге. Ако већ нису, ученике научити да отворе и подесе налог електронске поште и користе електронску пошту за размену порука и докумената. Ученицима приказати различите облике веб-сајтова (друштвене мреже, блогове, форуме, едукативне сајтове, географске мапе, сајтове за електронску трговину и банкарство и слично). Илустровати употребу програма за аудио и видео комуникацију преко интернета и користити је у групном раду ученика (на пример, током израде домаћих задатака ученици могу да сарађују коришћењем неког сервиса за аудио и видео комуникацију, на пример, Skype или Google Hangouts). Упознати ученике и са појмом веб-апликација (тј. апликација "у облаку").

Скренути ученицима пажњу на лепо понашање на интернету (netiquette), као и на етичка и правна питања приступа садржајима (ауторска права, лиценце).

Такође, продискутовати важна питања безбедности и приватности на интернету (нарочито у контексту употребе друштвених мрежа). При свему овоме неопходно је перманентно радити на развијању свести о важности поштовања правних и етичких норми при коришћењу Интернета, критичком прихватању информација са веба,

поштовању ауторских права при коришћењу информација са веба, поштовању права приватности.

Тематска целина **Увод у алгоритме** у првом и затим **Увод у програмирање** у другом разреду су основа за даљу надоградњу знања из програмирања у трећем разреду. Наставу треба организовати тако да се код ученика развија логичко и критичко мишљење приликом решавања постављених проблема.

Пре почетка наставе програмирања, кроз пар часова, ученицима треба објаснити важност алгоритамског приступа решавању проблема и вештине програмирања у данашњем свету за све, па и за ученике који се неће професионално бавити програмирањем. Ученике треба упознати са позитивним утицајем наставе програмирања на развој прецизног, формалног изражавања и на развој логичког мишљења. Често се у животу дешава да треба да организујемо неку групу људи и да им објаснимо како даураде неки посао, а познавање техника описивања алгоритама ће нам помоћи да ту увек урадим на јасан и недвосмислен начичн. ИКТ уређаји нас често збуњују чудним понашањем, које се испоставља као сасвим природно и нормално, ако разумемо строге принципе по којима функционишу. Даље, потребно је ученицима кроз серију пажљиво одабраних, једноставних примера приказати могућност употребе техника алгоритамског размишљања и програмирања на аутоматизовање практичних задатака са којима се ученици и иначе срећу током образовања и личног и касније професионалног живота. Данас су окружења за програмирање таква да је уз веома мало труда могуће направити апликације за мобилни телефон која омогућавају и пуштање музике и приказ слика и видео-материјала, лако се повезују са интернетом, омогућавају дељење информација и слично. Поред имплементирања заокружених апликација, технике програмирања се могу применити за писање различитих скриптова који олакшавају решавање различитих облика проблема (командни скриптови оперативног система, скриптови у веб-страницама, скриптови у математичком, техничком и статистичком софтверу, скриптови у канцеларијском софтверу и слично). Тако је ученицима, на пример, могуће показати како уз веома мало програмирања (практично кроз програме линијске структуре) у погодним окружењима (на пример, GNU Octave или SciPython) могу много једноставније да реше задатке из физике и графички прикажу резултате, него када се то ради на класичан начин, уз помоћ обичног калкулатора. Навести ученицима колико познавање принципа алгоритама и програмирања може помоћи скоро у свакој професији (на пример, будући психолози и социолози користе напредан статистички софтвер за обраду својих података и да им

елементи програмирања могу помоћи да прилагоде тај софтвер потребама своји специфичних анализа, а будући стручњаци за језик могу помоћу релативно једноставних програмерских техника извући потребне делове језичких корпуса и статистички их обрађивати).

У наредних пар часова, ученике је кроз неколико једноставних примера потребно неформално упознати са појмом (процедуралног) алгоритма и различитих начина да се неки алгоритам опише. Ученицима је потребно приказати алгоритме описане на природном језику, алгоритме описане на псеудокоду и алгоритме описане различитим дијаграмским језицима (на пример, блок дијаграми тока програма). Пожељно је један исти алгоритам описати на различите начине. Ученицима разјаснити шта се сматра елементарним кораком у алгоритму, како се кораци секвенцијално могу ређати један иза другог, како се врши условно извршавање одређених делова алгоритма (гранање) и како се одређени делови алгоритма могу понављати (петље тј. циклуси). Увести и могућност скока (тзв. GOTO) на обележени корак (на пример, на број корака у алгоритму представљеном у облику нумерисаног низа корака) и укратко на примеру продискутовати мане таквих решења.

У наставку, кроз неких шест часова, је пожељно са ученицима продискутовати велики број примера алгоритама из реалног живота (на пример, кухињски рецепти, организација летовања коришћењем интернета, одређивање победника у игри папир-камен- маказе, слагање Рубикове коцке, слагање Ханојских кула, погађање непознатог броја половљењем интервала), али и из претходног математичког образовања (на пример, алгоритам за сабирање бројева потписивањем, алгоритам за одређивање записа броја у датој основи, множење два полинома, Еуклидов алгоритам за одређивање НЗД, алгоритам за одређивање највећег од три дата броја). Алгоритме описивати на разне начине (говорним језиком, псеудокодом, дијаграмима). Инсистирати на томе да сви ученици могу да спроведу задати алгоритам корак по корак, експлицитно записујући (на пример у облику табеле) стање, тј. текуће вредности променљивих током извршавања алгоритма, пре него што пређу на следећи ниво, а то је самостално осмишљавање и описивање алгоритама. Анализирати са ученицима више различитих решења истог проблема. Радити на развијању свести код ученика о томе да решење проблема треба да буде образложено јасно и прецизно (најбољи тест је то да је алгоритам описан тако да га други ученик може спровести без икаквих додатних упутстава), као и да је потребно да се решења до којих се дошло провере и процене у односу на друга решења.

У склопу десеточасовне обраде визуелног програмирања ученике упознати са неким системом за учење програмирања који подржава задавање алгоритма слагањем блокова (на пример, Code.org, MIT Scratch, Alice). Кроз велики број примера ученике провести кроз алгоритме различите сложености (од најједноставнијих секвенци корака, па до алгоритама који садрже и сложеније облике гранања и понављања). У сложенијим примерима инсистирати на декомпозицији и рашчлањавању проблема на делове који се засебно лакше решавају (приступ програмирању одозго-наниже). Инсистирати на детаљном тестирању решења, пажљивом праћењу свих порука о грешкама, извршавању алгоритама корак по корак, техникама дебаговања и проналажењу и исправљању грешака.

II разред
(60 часова годишње)

САДРЖАЈИ ПРОГРАМА

1. Рад са табелама (18)

Објаснити основне појмове о програмима за рад са табелама (табела, врста, колона, ћелија,…) и указати на њихову општост у програмима овог типа.

При уношењу података у табелу, објаснити разлику између различитих типова података (нумерички формати, датум и време), као и грешке које могу из тога да настану. Приликом манипулација са подацима (означавања ћелија, кретање кроз табелу, премештање, копирање,…), указати на општост ових команди и упоредити их са сличним командама у програмима за обраду текста.

Код трансформација табеле указати на различите могућности додавања или одузимања редова, или колона у табели. Објаснити појам опсега.

Код форматирања приказа податка у ћелији, приказати на примерима могућност различитог тумачења истог нумеричког податка (број, датум, време). Такође, нагласити важност доброг приказа података (висине и ширине ћелија, фонта, поравнања) и истицања појединих података или група података раздвајањем различитим типовима линија и бојењем или сенчењем.

Указати на повезаност података у табели и могућност добијања изведених података применом формула. Објаснити појам адресе и различите могућности референцирања ћелија. Указати на различите могућности додељивања имена подацима или групама података и предности коришћења имена.

Приказати функције уграђене у програм и обратити пажњу на најосновније функције, посебно за сумирање и сортирање.

Указати на различите могућности аутоматског уношења података у серији.

Посебну пажњу посветити различитим могућностима графичког представљања података. Указати на промене података дефинисаних у табели формулама, и графикону у случају измене појединих података у табели. Указати на могућност накнадних промена у графикону, како у тексту, тако и у размери и бојама (позадине слова, скале, боја, промена величине,…).

Указати на важност претходног прегледа података и графикона пре штампања, као и на основне опције при штампању.

Све појмове уводити кроз демонстрацију на примерима. Од самог почетка давати ученицима најпре једноставне, а затим све сложеније примере кроз које ће сами практично испробати оно што је демонстрирао наставник.

Препоручљиво је да се сви нови појмови уведу у првих 10 часова тако што ће ученици радити задатке које је припремио наставник (текстуалним описом задатка или задатом коначном табелом, одштампаном, без увида у формуле) а затим ученицима дати конкретне мале пројекте различите природе: да направе електронски образац (на пример предрачун или нешто слично), прикупљање и обраду података који се односе на успех ученика из појединих предмета, неку појаву или процес из других наставних и ваннаставних области рада и интересовања ученика.

2. Рачунарска графика (11)

Увод у рачунарску графику (препорука: 1 час)

Објаснити разлику између векторског и растерског представљања слике, предности и недостатке једног и другог начина. Објаснити основне типове формата слика и указати на разлике међу њима. Указати на постојање библиотека готових цртежа и слика.

При увођењу појмова растерске и векторске графике, нека ученици на својим рачунарима паралелно отворе прозоре програма за цртање који је у саставу оперативног система и нпр. текст-процесор, рећи им да у оба нацртају елипсу и максимално зумирају, нацртају затим обојени квадрат преко дела елипсе и покушају да га „преместе“, при свему томе захтевати од њих да изводе закључке у вези са карактеристикама једне и друге врсте графике. Направити паралелу између ове две врсте графике у односу на цртеже воденим бојама и колаже од папира. Код наставне јединице која се односи на формате датотека илустровати конкретним примерима, урађеним од једне фотографије, зумирати слике, поредити величине датотека.

Пример програма за креирање и обраду растерске графике (препорука: 6 часова)

Припремити за часове дигитални фото-апарат или мобилни телефон са камером и на часу правити фотографије. На претходном часу дати ученицима задатак да донесу фотографије које ће на часу скенирати. Ученици могу на својим фотографијама да

увежбавају технике основних корекција и обраде фотографије: уклањање „црвених очију“, ретуширање, фото-монтажу, промену резолуције и формата слике, а затим направе фото-албум свих радова.

Пример програма за креирање векторске графике (препорука: 4 часова)

Посебну пажњу посветити пројектовању цртежа (подели на нивое, уочавању симетрије, објеката који се добијају померањем, ротацијом, трансформацијом или модификацијом других објеката итд.), као и припреми за цртање (избор величине и оријентације папира, постављање јединица мере, размере, помоћних линија и мреже, привлачења, углова, итд.).

Код цртања основних графичких елемената (дуж, изломљена линија, правоугаоник, квадрат, круг, елипса) објаснити принцип коришћења алатки и указати на сличности са командама у различитим програмима. Слично је и са радом са графичким елементима и њиховим означавањем, брисањем, копирањем, груписањем и разлагањем, премештањем, ротирањем, симетричним пресликавањем и осталим манипулацијама. Указати на важност поделе по нивоима и основне особине нивоа (видљивост, могућност штампања, закључавање).

Код трансформација објеката обратити пажњу на тачно одређивање величине, промену величине (по једној или обе димензије), промену атрибута линија и њихово евентуално везивање за ниво. Посебно указати на разлику отворене и затворене линије и могућност попуњавања (бојом, узорком, итд.).

Указати на важност промене величине приказа слике на екрану (увећавање и умањивање цртежа), и на разлоге и начине освежавања цртежа. Код коришћења текста указати на различите врсте текста у овим програмима, објаснити њихову намену и приказати ефекте који се тиме постижу.

Код штампања указати на различите могућности штампања цртежа и детаљно објаснити само најосновније.

За увежбавање дати ученицима конкретан задатак да нацртају грб школе, свог града или спортског друштва, насловну страну школског часописа, рекламни пано и сл.

3. Мултимедија (5)

Обраду ове теме засновати на искуствима ученика, резимирати њихова знања, запажања и искуства у раду са звуком и видеом.

Направити упоредни преглед неколико програма за репродукцију звука.

При упознавању са основним форматима записа звука, направити паралелу између растерске и векторске графике са једне стране и снимљеног и синтетичког звука са друге стране. Дати ученицима прилику да сниме сопствени глас и репродукују га. Повезати са темом израде презентација у првом разреду и могућношћу снимања нарације уз слајдове.

Направити упоредни преглед неколико програма за репродукцију видео-записа.

Рад са видео-записима засновати на видео радовима ученика направљених на часу или припремљених унапред (у виду домаћих задатака). Потребно је да ученици савладају основне технике монтаже видео материјала, звука, ефеката и натписа.

Посебну пажњу обратити на проблематику ауторских права и етичких норми при коришћењу туђих звучних и видео записа, као и на поштовање права на приватност особа које су биле актери снимљених материјала и тражње њихових дозвола за објављивање.

4. Напредно коришћење интернета (6)

Веб апликације и дељење докумената (препорука: 3 часова)

Упознати ученике са принципима, предностима и недостацима употребе веб-апликација и радом „у облаку“ (енгл. Cloud computing). Подстицати ученике на размишљање и наводити да они изводе закључке о овој теми.

Представити ученицима различите системе за рад са веб апликацијама и дељење докумената, а ученицима пружити прилику да раде у једном од њих.

Блог, вики, електронски портфолио (препорука: 3 часова)

Приказати ученицима конкретне примере блога, викија, и електронског портфолија, размотрити могућности примене, ученицима пружити прилику да креирају садржаје и коментаре на вебсајтовима и порталима са слободним приступом или у саставу школског веб-сајта или платформе за електронски подржано учење. Активности осмислити тако да подстичу тимски рад, сарадњу, критичко мишљење, процену и самопроцену кроз рад на часу, примену у другим наставним областима и домаће задатке.

Посебну пажњу обратити на проблематику ауторских права, етичких норми, поштовање права на приватност, правилно писање и изражавање и правила лепог понашања у комуникацији.

5. Увод у програмирање (20)

“Увод у програмирање” у другом разреду су основа за даљу надоградњу знања из програмирања у трећем разреду. Наставу треба организовати тако да се код ученика развија логичко и критичко мишљење приликом решавања постављених проблема.

Програмирање као начин решавања проблема (2)

Пре почетка наставе програмирања ученицима треба објаснити важност алгоритамског приступа решавању проблема и вештине програмирања у данашњем свету за све, па и за ученике који се неће професионално бавити програмирањем. Ученике треба упознати са позитивним утицајем наставе програмирања на развој прецизног, формалног изражавања и на развој логичког мишљења. Често се у животу дешава да треба да организујемо неку групу људи и да им објаснимо како даураде неки посао, а познавање техника описивања алгоритама ће нам помоћи да ту увек урадим на

јасан и недвосмислен начин. ИКТ уређаји нас често збуњују "чудним" понашањем, које се испоставља као сасвим природно и нормално, ако разумемо строге принципе по којима функционишу.

6. Увод у програмирање у текстуалним програмским језицима (18)

Настава у другом разреду треба да помогне ученицима да направе прелаз са неформалних и дијаграмских описа алгоритама на потпуно прецизне описе алгоритама задате у виду кода у неком програмском језику. Ученике је потребно упознати са појмом програмског језика, а затим и са неким програмским језиком и одговарајућим системом или библиотеком специјализованом за наставу програмирања. Такви су рани наставни програмски језици *Logo* и *Karel*, али и савремени системи, попут система *Greenfoot* имплементираног у програмском језику *Java* или сличних система имплементираних у другим програмским језицима (на пример, *Python, C#, Lazarus/Delphi* или *JavaScript*). Пожељно је да текстуални језик у коме се врши ова почетна обука програмирања буде неки програмски језик који се користи у савременом програмирању. Једна могућност је да се тај језик користи у настави и током трећег разреда (да ученици не би морали да у трећем разреду уче нову синтаксу).

Након излагања одређеног броја примера, скренути ученицима пажњу на основне појмове програмирања који се јављају у написаним програмима (наредбе, изрази, променљиве, константе, додељивање вредности, оператори, декларације, дефиниције, потпрограми, објекти, атрибути, методи и слично) и инсистирати на томе да ученици у готовим програмима могу да разликују основне елементе синтаксе и да наведу њој одговарајућу семантику.

Инсистирати на примени конвенција које програме чине што разумљивијим осталим програмерима (назубљивање кода, пажљиво именовање променљивих, избегавања понављања кода, писање адекватних коментара).

Пошто је у настави програмирања у трећем разреду предвиђено покривање неких основних концепата објектно оријентисане парадигме (пре свега креирање и коришћење објеката који су део библиотеке окружења које се користи), пожељно је да се у уводном прегледу алгоритама и програмирања покрију ти концепти. За то су веома погодни *Greenfoot* (погодан за наставу у другом разреду). Овај систем је заснован на програмском језику *Java*, који је широко распрострањен и може се користити и у настави у трећем разреду.

III разред
(оба типа гимназије)

- **Програмски језици и развој програма (2)**
- **Објектно оријентисано програмирање (2)**
- **Креирање аплкација у интегрисаном развојном окружењу (4)**
- **Преглед основних елемената програмског језика (6)**
- **Алгоритми и програми линијске структуре (4)**
- **Алгоритамска декомпозиција и примена потпрограма (4)**

- **Алгоритми и програми разгранате структуре (10)**
- **Алгоритми и програми цикличне структуре (16)**
- **Сложени типови података и њихова употреба у програмима (8)**
- **Приступ датотекама из програма (4)**

Начин реализације програма

Реализација програма рачунарства и информатике постиже се добром организацијом наставног процеса, што практично значи:

- рационално коришћење расположивог фонда часова,

- добра организација практичних вежби на рачунару,

- добар избор задатака који се решавају.

Примере, који се користе у реализацији наставе као и фонд часова посвећен обради појединих тема може се прилагодити појединачним смеровима гимназије. Уколико ученици имају потешкоћа у савладавању неких предвиђених тема, предлаже се да се одвоји већи број часова за њихово увежбавање, па и по цену да се неке напредније теме обраде у мањем обиму или чак потпуно изоставе (на пример, ако наставник процени да је ученицима тешко да савладају једнодимензионалне низове, боље је посветити већи број часова њиховој обради, него по сваку цену инсистирати да се обраде и вишедимензионални низови).

У погледу организације рада, значајно је обратити пажњу на следеће елементе:

- Теоријска настава се изводи са целим одељењем и, по потреби, наставник практично демонстрира поступак решавања проблема уз употребу рачунара. Уколико услови то дозвољавају, препоручује се извођење и теоријске наставе у рачунарском кабинету.

- За извођење вежби одељење се дели на две групе. Увежбавање и практичан рад изводи се у рачунарском кабинету, под контролом наставника. Ученици изводе вежбе самостално или у групама, уз потребна упутства наставника. Током вежбе ученицима треба понудити задатке различите тежине. Одређен број задатака за самостално решавање треба да захтева примену алгоритама и техника које су приказане током теоријске наставе (на пример, ако се на теорији прикаже одређивање максимума, на вежбама се може захтевати одређивање минимума неког скупа вредности).

- Оцењивање ученика треба обављати систематски у току школске године. Елементи за оцењивање треба да буду резултати рада на практичним вежбама, усмене провере знања, као и укупан ученików однос према раду. Наставник може проверу знања да

врши и кроз највише два једночасовна писмена задатка, по један у сваком полугодишту.

Током извођења наставе и оцењивања инсистирати на дисциплинованом и уредном писању програмског кода (назубљивању, пажљивом именовању променљивих, писању коментара, избегавању делова који се понављају), јер је један од главних циљева овог предмета да развије јасноћу и прецизност у изражавању.

Упутства за реализацију појединачних тема

1) **Програмски језици и развој програма (2)**
- Кратак преглед развоја програмских језика
- Фазе у развоју програма
- Окружења за развој програма, превођење и изградња програма

Ученике подсетити на појам алгоритма и начине описа алгоритама, као и на појам програмског језика као изражајног средства за потпуно прецизну спецификацију алгоритама. Приказати неколико најчувенијих програмских језика (на пример, Fortran, Cobol, Lisp, Algol, C, Pascal, Prolog, Basic, C++, Java, C#, JavaScript, PHP) и продискутовати њихов значај и употребу током историје, али и у данашњем свету. С обзиром на то да ученици немају пуно искуства у програмирању, не инсистирати на карактеристикама тих програмских језика, као ни на разјашњавању програмских парадигми. Након описа језика вишег нивоа, продискутовати и појам асемблерског и машинског језика, кроз неки елементарни пример, али без улажења у детаље. Описати и појам превођења програма.

Укратко и само информативно описати најзначајније фазе у развоју сложенијег софтвера.

Укратко описати и развој окружења за развој програма и концепт савремених интегрисаних развојних окружења. Навести основне компоненте интегрисаног окружења (едитор, компилатор, линкер, дебагер) и укратко објаснити поступак изградње извршног програма, тј. објаснити шта се све дешава када се притисне дугме за покретање програма (наравно, само информативно, без улажења у техничке детаље).

2) **Објектно оријентисано програмирање (2)**
- Објекти, атрибути, методи
- Класе и односи међу класама и објектима

Током обраде ове теме ученике упознати са основним концептима објектно оријентисане парадигме, али само у мери која је потребна да би ученици могли да креирају и користе објекте библиотеке програмског језика (на пример, објекте који одговарају елементима корисничког интерфејса). Ученицима је потребно детаљно објаснити концепт објекта, његових атрибута и његових метода. Навести примере објеката из стварног света и њихових могућих атрибута и метода (без приказа њихове имплементације). Такође, навести одређени број објеката који представљају рачунарске апстракције (на пример, прозор, дугме, ток). Након увођења објеката, дефинисати појам класа, и разјаснити однос између класа и објеката. Дефинисање класа и њихову имплементацију могуће је приказати само информативно и то у каснијим фазама наставе, када ученици детаљно овладају свим елементарнијим концептима (типовима података, потпрограмима). Детаљно објашњавање напреднијих концепата

(наслеђивања, полиморфизма, агрегације, асоцијације) није потребено (ученици немају довољно предзнање да их у овом тренутку разумеју, а ни потребу да их користе у својим програмима).

3) **Креирање аплкација у интегрисаном развојном окружењу (4)**
- елементи развојног окружења
- апликације са командно-линијским и графичким корисничким интефејсом
- структура програма и пројекта
- дизајн корисничког интерфејса; појам контрола и врсте контрола
- појам догађаја и обраде догађаја; програмирање обраде догађаја
- графика (цртање по прозору)
- дебаговање

Показати ученицима развојно окружење изабраног програмског језика. Објаснити развојно окружење и елементе радне површине.

Објаснити разлику између програма са командно линијским интерфејсом (КЛИ) и са графичким корисничким интерфејсом (ГКИ) и приказати како се у одабраном окружењу започиње развој и једне и друге врсте програма. Објаснити појам пројекта и како се пројекат креира, чува и отвара. Током наставе инсистирати на креирању апликација са ГКИ, али приказати и креирање апликација са КЛИ у мањој мери и где је то адекватно (на пример, у програмима који само учитавају неке бројеве, обрађују их и приказују резултат у облику једог или више бројева).

Приликом креирања апликација са ГКИ могу се уочити две етапе које се преплићу: етапа дизајна корисничког интерфејса и етапа кодирања. Кроз веома једноставне примере апликација објаснити ученицима овај поступак.

Упознати ученике са основним контролама које ће бити коришћене у програмима (прозори, поља за унос текста, натписи, дугмад). Напредније котроле могуће је уводити и у каснијим фазама наставе (али увек је пожељно експлицитно их увести и описати, пре коришћења у примерима).

Ток апликације са ГКИ вођен је догађајима, који се догађају асинхроно (обично на основу акција корисника апликације, каква је клик на дугме, притисак тастера или превлачење мишем). Објаснити концепт догађаја и механизам обраде догађаја. Навести и разјаснити основне врсте догађаја.

На почетку описа фазе кодирања апликације ученицима дати неки груби преглед структуре програма. Описати датотеке које чине пројекат и њихову улогу као и општу структуру аутоматски генерисаног кода у датотеци коју ће ученици модификовати (пре свега допуњавати имплементацијама метода обраде догађаја). Након превођења и изградње извршног програма, приказати ученицима извршну датотеку и демонстрирати да ју је могуће покретати потпуно независно од интегрисаног развојног окружења (инсистирати на томе да се тај код може покретати и на рачунарима на којем развојно окружење уопште није инсталирано).

Употребити ранији преглед објектно оријентисаног програмирања да би се објаснио приступ интерфејсу из програма. Приказати дефиницију класе која представља прозор (форму). Приказати објекте којима су представљене контроле. Објаснити поступак генерисања методе за обраду догађаја и приказати генерисане костуре тих метода у програмском коду.

Приказати како је из метода за обраду догађаја могуће приступати и мењати одређене атрибуте контрола које су постављене на прозор (форму), као и одређене атрибуте самог прозора. Провежбати одређени број једноставних примера апликација које током обраде догађаја само мењају својства контрола (на пример, притском на

дугме, мења се његова позиција или боја позадине прозора, кликом на прозор он се помера, кликом на неко од три радио-дугмета мења се поравнање текста у неком натпису, унети текст се преписује из унетог поља у наслов прозора и слично). Обратити пажњу да ученици у овом тренутку још нису прешли преглед основних елемената програмског језика (изрази, наредбе), тако да у методима за обраду догађаја још није пожељно писати сложенији код (на пример, програм који сабира два унета броја захтева познавање ниски и бројева и конверзије међу њима, тако да се може приказати касније).

Приказати ученицима како се ради са графиком (цртање и бојење различитих облика), јер задаци са графиком могу бити веома интересантни ученицима и наводе их да развијају геометријске способности и вештине пројектовања, поред програмерских вештина решавања проблема. Подсетити ученике на појам координатног система, координата и геометријских облика. У овој фази могуће је приказати једноставне програме који исцртавају неке цртеже по прозору (аутомобилчић, кућицу, Чича Глишу, ...).

Још једна веома важна компонента савремених интегрисаних окружења је интегрисани дебагер. Приказати ученицима како се користи дебагер да би се код извршавао корак по корак, да би се извршавао до тачке прекида, да би се прегледале вредности променљивих (у овом примеру објеката контрола и њихових својстава). На ову тему се враћати стално током наставе и инсистирати на томе да коришћење дебагера мора ученицима да постане уобичајена пракса.

4) **Преглед основних елемената програмског језика (6)**
- изрази (променљиве, константе, оператори), типови
- наредбе (додела, гранање, петље)
- потпрограми (пренос параметара и повратна вредност)

Увести укратко ученицима основне концепте програмског језика. Иако овај део наставе треба да буде одређена систематизација лексичких и синтаксичких концепата језика, требало би настојати да прича не буде чисто енциклопедијског карактера и да излагање не буде у стилу референтног приручника, већ је пожељно важне концепте увести приказујући конкретне, једноставније примере програма и идентификујући у њима све концепте о којима се прича. При том, треба узети у обзир да су ученици током првог и другог разреда већ стекли одређено искуство у програмирању (па чак и програмирању у текстуалном језику), тако да су се са многим од концепата који се уводе већ срели (додуше на неформалан начин). Рани преглед мало ширег скупа концепата програмског језика решава проблем циркуларних зависности и омогућава ученицима да доста рано крену да пишу програме који имају нетривијалну функционалност. На овом месту је добро повући паралелу са учењем страног језика где се увек истовремено уводе сви синтаксички елементи (на пример, незамислив је приступ у коме би се прво до ситних детаља проучавале именице, без икаквог помињања глагола и придева).

Увести основне лексичке (идентификаторе, бројевне константе, дословне ниске, коментаре) и синтаксичке елементе језика (изразе, наредбе). Скренути пажњу ученицима на лексичке конвенције које су изразито важне у програмирању (исправно назубљивање кода, пажљиво именовање променљивих, писање коментара) и инсистирати на њиховој примени у свим задацима који следе (на пример, није допустиво елементе интерфејса називати Edit7 или Button2).

Увести појам израза и прецизно описати поступак изградње сложених израза (применом оператора и функција на једноставније изразе, при чему су елементарни

изрази константе и променљиве). Увести појам типова података тј. типа израза (типа променљивих, декларације променљивих, дефиниције и типа константи, типа оператора и функција). Дефинисати бројевне типове, логички тип, карактерски тип, тип ниске карактера (стринг), набројиве типове, интервалне типове, скуповни тип. Сложене типове података (низове, ниске карактера, структуре) поменути само укратко јер је њихова детаљна обрада предвиђена касније. За сваки тип описати и основне операторе и најчешће коришћене функције које оперишу над вредностима тог типа. Увести појмове н-арности, приоритета и асоцијативности оператора. У језицима са богатим скупом оператора не инсистирати на самом почетку на веома специфичним операторима (на пример, разлику између префиксног и постфиксног оператора ++ није потребно описивати у првој фази приче о изразима, већ је могуће продискутовати је касније, када ученици стекну одређено искуство и сигурност). Приликом писања израза предност дати разумљивости почетницима, а не језгровитост и што краћи запис. Увести појам имплицитне и експлицитне конверзије типова.

Увести појам наредби и увести синтаксу и семантику основних типова наредби. Дискутовати посебно наредбу доделе, наредбе гранања и наредбе понављања (петље). Детаљнија разрада алгоритама који користе понављање и гранање предвиђена је за касније и тада је могуће се опет вратити на детаљнији преглед одговарајућих наредби, тако да их је у првом пролазу могуће увести потпуно информативно. Слично је могуће урадити и са потпрограмима.

5) **Алгоритми и програми линијске структуре (4)**
- алгоритми засновани на примени математичких формула

У оквиру ове теме са ученицима урадити већи број једноставнијих примера програма линијске структуре у којма се учитавају подаци (обично бројеви), обрађују применом једне или више математичких формула и приказује резултат. Кроз ову тему ученици вежбају запис израза у програмском језику, рад са бројевним типовима података и коришћење једноставних контрола (поља за унос текста, натписа, дугмади). Препоручује се да се са ученицима прикаже и неколико примера креирања конзолних апликација, а да се у апликацијама са графичким интерфејсом примени визуализација тј. цртање, када год је то примерено. Погодни примери који се у овој теми могу обрађивати су, на пример, израчунавање обима, површина и запремина геометријских фигура и тела, различита прерачунавања јединица, израчунавања заснована на примени тригонометрије, примена физичких формула (кретање, хоризонтални, вертикални и коси хитац) и слично. Нарочито битни су изрази засновани на модуларној аритметици (одређивање цифара у запису броја, рад са записима у основи 60 која се користи у запису времена и при мерењу углова и слично) и ученицима обавезно скренути пажњу на ове технике.

Саветује се да се кроз ове примере увежбава и процес моделовања математичких и физичких система и њихових решења, тако што се ученицима дају текстуални задаци у којима ученици прво треба да осмисле одговарајући модел, изведу потребне формуле и тек онда их представе у облику израза програмског језика и имплементирају. На задацима таквог типа пожељно је инсистирати и у каснијим областима (гранању, петљама), али увек је потребно прво осигурати да ученици довољно владају техником потребном да математички модел проблема и алгоритам који осмисле умеју да претворе у исправан програмски код. Способност моделовања је пожељна, али не би требало да пресудно утиче на оцену ученика (ученике који не владају довољно математиком или физиком не би требало кажњавати лошијом оценом из програмирања).

6) **Алгоритамска декомпозиција и примена потпрограма (4)**

- издвајање потпрограма
- пренос параметара и повратне вредности

Изузетно важна техника у решавању сложенијих проблема је његова декомпозиција на једноставније потпроблеме који се онда једноставније решавају. Основна техника процедуралне алгоритамске декомпозиције заснива се на издвајању потпрограма који врше одређена израчунавања (на пример, израчунава растојање између две тачке задате својим координатама) или одређен испис или исцртавања (на пример, потпрограма који црта аутомобил или кућицу дате величине и боје на датој позицији на екрану).

Ученицима детаљно увести појам потпрограма, врсте потпрограма, њихову синтаксу и семантику. Описати параметре потпрограма (објаснити разлику између формалних параметара који се користе у дефиницији потпрограма, и стварних параметара тј. аргумената који се наводе у позиву) и повратну вредност потпрограма. Објаснити начине преноса параметара и могућност да се преко параметара врати више вредности из потпрограма (на пример, написати потпрограм који на основу угла у радијанима, одређује најближи угао задат степенима, минутама и секундама). Објаснити разлику између обичних (глобалних или статичких) потпрограма и метода (који приступају атрибутима класе). Објаснити појам опсега (области дефинисаности променљивих) и разликовати локалне и глобалне променљиве (при чему коришћење глобалних променљивих треба избегавати када год је то могуће).

У сарадњи између ученика и наставника одређени број сложенијих програма рашчланити на једноставније потпроблеме и сваки потпроблем решити помоћу засебног метода (ученици могу да имплементирају једноставније методе, а најбољи ученици или наставници могу да имплементирају оне компликованије). Инсистирати код ученика да, пре саме реализације потпрограма, јасно дефинишу шта су улазни подаци (листа параметара), а шта је излазни податак или излазни подаци. Направити разлику између потпрограма који враћају неку вредност и оних које то не раде (void функције тј. методе или процедуре). Приказати примере и једних и других.

Приказати како се увођењем потпрограма спречава понављање сличног кода на више места у програму. Такође приказати како се исти потпрограми могу користити у решавању различитих задатака (ако наставници процени да је то примерено, он може ученицима приказати како се креирају библиотеке које садрже потпрограме и како се оне укључује и користи у различитим програмима).

Објаснити како је увођење потпрограма механизам који помаже да се интерфејс програма веома јасно раздвоји од његове централне функционалности (на пример, исту централну функцију употребити и у графичкој и у конзолној апликацији).

Након увођења алгоритамске декомпозиције и потпрограма, активно их користити у свим програма који следе (на пример, када за то дође време, увести потпрограм за израчунавање факторијела или проверу да ли је број прост).

7) **Алгоритми и програми разгранате структуре (10)**

- вишеструко гранање и угнежђене наредбе гранања
- гранање на основу дискретног скупа вредности, гранање на основу припадности интервалу, хијерархијско гранање
- контроле избора (радио-дугме, поље за потврду, ...)

У оквиру ове теме са ученицима је потребно обрадити имплементацију програма сложеније разгранате структуре.

Приказати одређен број примера једноставног гранања (коришћењем непотпуног и потпуног облика наредбе if-then-else). На пример, приказати израчунавање апсолутне вредности броја, одређивање максимума два, а затим и три броја и слично. Ако језик то подржава, приказати и тернарни оператор гранања (обично је то оператор ?:) и нагласити разлику између гранања унутар израза и наредби гранања.

Приказати да је вишеструко гранање могуће постићи и помоћу специјалних наредби (switch, case) али и угнежђавањем простих наредби гранања (конструкцијом else-if). Приказати одређени број примера гранања и класификације на основу дискретног скупа вредности (на пример, одређивање имена дана на основу редног броја у недељи), затим класификације на основу припадности интервалу (на пример, одређивање агрегатног стања воде или одређивање успеха ученика на основу вредности просечне оцене). У одређеном броју примера илустровати реализацију потпрограма који врше овакву класификацију и коришћење набројивих типова за репрезентацију коначног скупа вредности (на пример, успех ученика би требало да буде представљен набројивим типом). Приказати одређен број примера хијерархијског гранања (на пример, одређивање квадранта у којем се налази тачка или класификација на основу неког стабла одлучивања).

Могући сложенији примери укључују рад са датумима (одређивање преступне године, одређивање сутрашњег или јучерашњег датума и слично). Када год је то погодно уводити потпрограме којима се постиже разумљивост и једноставност кода (на пример, приликом рада са датума веома је корисно имати потпрограм којим се израчунава број дана у датом месецу дате године). Математички примери у којима се користе сложенији облици гранања су дискусија броја и врсте решења и решавање линеарних једначина, система једначина, квадратних једначина и слично.

Одређене контроле ГКИ се често користе тако што програм гранањем на основу њиховог статуса врши избор онога што се рачуна (таква су радио-дугмад, поља за потврду и слично). Ученицима скренути пажњу на рад са овим контролама и користити их током обраде наредби гранања када год је то потребно.

8) **Алгоритми и програми цикличне структуре (16)**

- читање и приказ серије елемената; контроле за унос и приказ серија елемената (листа, мемо-поље, ...)
- алгоритми линеарне обраде серије елемената
- угнежђене петље
- једноставнији алгоритми теорије бројева

Са ученицима се детаљно подсетити различитих наредби којима се реализују петље у програмима (бројачка петља, петља са провером услова на почетку, са провером услова на крају, наредбе за прекид петље и текуће итерације петље) и продискутовати везе између њих (могућност трансформације једног облика петље у други) и адекватност њихове употребе. Пре преласка на напредније алгоритме потребно је осигурати да ученици разумеју семантику сваке врсте петље, што се најбоље може проверити инсистирањем на постпуном приказу вредности променљивих током сваког корака извршавања петље (на пример, у облику табеле).

С обзиром на то да програми са понављањем често имају потребу за уносом или приказом серија елемената (серија речи, серија бројева и слично), приказати како се у програмима са ГКИ може вршити унос и приказ таквих серија (приказати за то погодне контроле, попут листе, мемо-поља и слично). Још једна могућност је итеративно

приказивање системског дијалога за унос и системског дијалога за приказ порука (MessageBox), као и могућност поступног уношења серије коришћењем обичног поља за унос текста и дугмета чијим се притиском унети текст учитава и обрађује као наредни елемент серије (и на пример, преписује у листу која садржи претходно унете елементе).

Увести алгоритме за линеарну обраду серија података. Са ученицима обрадити алгоритме генерисања серија елемената (на пример, генерисање правилно распоређених серија целих бројева попут серије непарних бројева из датог интервала, генерисање еквидистантно распоређених тачака неког реалног интервала, генерисања серије насумичних елемената, генерисања серије цифара записа датог броја и слично). Серије могу бити приказане у одговарајуће контроле, графички приказане на одређени начин, уписане у низове или додатно обрађиване.

Увести алгоритам за филтрирање елемената серије (на пример, издвојити све парне бројеве међу бројевима унетим са улаза). Увести алгоритам за пресликавање серије (на пример, приказати таблицу корена свих бројева од 1 до 100 или таблицу конверзије миља у километре). Увести различите алгоритме за агрегацију серије, тј. израчунавање збира, производа, минимума, максимума и сличних статистика серије (на пример, израчунати факторијел датог броја, степен за дату основу и изложилац, највећи међу унетим бројевима и слично). Увести алгоритме за претрагу серије (на пример, проверити да ли међу унетим или насумично генерисаним елементима постоји неки негативан број). На крају, приказати примере у којима се комбинује више алгоритама линеарне обраде серије елемената (на пример, одредити збир квадрата непарних цифара броја, у којем се комбинује алгоритам генерисања серије цифара у запису броја, филтрирања серије на основу испитивања парности, пресликавања добијене серије у серију квадрата и на крају агрегирања добијене серије у њен збир). Ученицима објаснити да описани алгоритми не зависе од тога одакле серија потиче нити од врсте петље којом се серија обрађује (алгоритам сабирања се увек може описати са “постави збир на нулу, а затим итерирај кроз елементе серије један по један и током итерације увећавај збир за текући елемент”, а једино што се разликује је како се та итерација кроз појединачне елементе остварује, што није зависно од алгоритма сабирања, већ од природе самог задатка). Серије елемената које се обрађују могу бити смештене и у низ (методички, сабирање елемената низа једноставније је од сабирања цифара броја, јер друго подразумева и алгоритам одређивања цифара из записа). Са друге стране, низови се могу одложити и за касније и обрадити у склопу обраде сложених типова података. Пожељно је кроз неколико примера увести и серије у којима се сваки следећи елемент одређује на основу претходног или неколико претходних (на пример, процена броја е сабирањем реда се може урадити ефикасније ако се примети да се сваки нови члан реда може једноставно добити од претходног, Фибоначијеви бројеви се рекурентно задају и слично).

Са ученицима урадити и одређени број класичних примера из теорије бројева (пре свега, одређивање делилаца броја, проверу да ли је број прост, растављање броја на просте чиниоце и слично). Повезати ове алгоритме са општим алгоритмима обраде серија (на пример, провера да ли је број прост се заправо заснива на претрази серије потенцијалних делилаца броја и провери да ли она садржи стварни делилац, тако да алгоритам претраге серије мора предтходити алгоритму испитивања да ли је број прост). Ако наставник процени да ови алгоритми математички представљају проблем ученицима (а то може да се деси на друштвеном смеру), могуће их је прескочити.

Увести појам угнежђене петље и веома детаљно и поступно га обрадити са ученицима (пожељно је извршавање програма корак по корак уз коришћење дебагера). Могући примери су испис таблице множења, испис свих минута и секунди у једном

сату и слично. Показати ученицима како се вишеструке петље понекад могу избећи издвајањем унутрашњих петљи у посебне потпрограме.

Још једна изузетно погодна група примера су примери у којима се користи графика и цртање разних правилних облика (на пример, пахуљица разних облика и величина, шаховског поља, зида испуњеног циглицама разне боје и слично).

9) **Сложени типови података и њихова употреба у програмима (8)**
- низови
- ниске карактера (стрингови)
- вишедимензионални низови (матрице)
- стурктуре (слогови)
- класе

Увести појам низа, алокације меморије потребне за смештање елемената низа, али детаљно обрадити само статички алоциране низове (у језицима у којим није подржана статичка алокација разматрати динамички алоциране низове чија је димензија унапред позната). Продискутовати пренос низова у потпрограм и враћање низова из потпрограма.

Ученицима објаснити основне алгоритме обраде низова. Пошто низови представљају серије елемената, сви раније уведени алгоритми обраде су директно примењиви и на низове. Додатно, потребно је урадити и специфичне алгоритме трансформације низова. На пример, избацивање елемената на датој позицији, избацивање елемената са датим својствима и слично.

Изразито важни алгоритми обраде низова су алгоритми њиховог сортирања. Ученицима је неопходно приказати бар један елементаран алгоритам сортирања низова. Сортиране низове је могуће ефикасно претраживати коришћењем бинарне претраге и тај алгоритам је такође потребно приказати ученицима.

Ниске карактера се могу обрађивати на веома сличан начин као о други низови. Важне операције над нискама су претрага подниске и разбијање ниске на основу датог сепаратора (на пример, поделити реченицу на низ речи које је сачињавају).

Вишедимензионалне низове увести, а комплексност задатака у којима се они користе проценити на основу раније приказаних резултата ученика.

Увести појам структуре и демонстрирати га на одређеном броју једноставних примера (структуре за репрезентовање тачака, разломака, ученика и слично). Дефинисање једноставних класа (на пример, класа ученика или класе разломка или комплексног броја) приказати само ако то време и до сада показани резултати у раду ученика то допуштају.

10) **Приступ датотекама из програма (4)**
- читање из датотеке
- писање у датотеку

Ученицима објаснити појам датотеке и приступ датотекама из програма. Обрадити текстуалне датотеке (бинарне датотеке евентуално обрадити само са нарочито заинтересованим ученицима). Објаснити поступак отварања и затварања приступа датотеци. Приказати читање података који су у датотеци јасно структурирани и правилно распоређени (на пример, читање линију по линију, читање линија таквих да се у свакој налази исти број података растављених зарезима, читање низа бројева и

слично). Приказати упис у датотеку и елементарне технике форматирања. Ако раније нису коришћени, у графичким апликацијама је у овом тренутку могуће демонстрирати употребу менија (File→Open, File→Save).

На крају приказати и један већи пример апликације који интегрише све обрађене елементе (на пример, пример телефонског именика, школског дневника и слично).