

ИНФОРМАТИКА И РАЧУНАРСТВО 6

Уџбеник за шести разред основне школе

Аутори

Доц. др Милош Папић, Далибор Чукљевић

Илустрације

Shutterstock

Рецензенти

Јелена Биџа, ОШ „Ђорђе Крстић“, Београд

Саша Петровић, ОШ „Ђура Јакшић“, Ковин

Момчило Ранђеловић, ЕТШ „Никола Тесла“, Ниш

Никола Рајевац, ОШ „Андрa Савчић“, ОШ „Нада Пурић“, Ваљево



ВУЛКАН
ЗНАЊЕ

Издавач

Вулкан издаваштво

Вулкан знање

Господара Вучића 245

Београд

www.vulkanznanje.rs

За издавача

Мирослав Јосиповић, Ненад Атанасковић, Саша Петковић

Директор

Нада Осмајић

Извршни уредник

Оливера Ђикановић

Уредник

Смиљка Наумовић

Лектура

Весна Ђукић

Коректура

Редакција Вулкан знања

Ликовни уредник

Предраг Ивановић

Графичко обликовање

Ана Јовановић

Дизајн корица

Никола Јованетић

Штампа

Графостил, Крагујевац

Прво издање, 2019

Тираж

12.000

ISBN 978-86-10-02412-8

CIP - Каталогизација у публикацији - Народна библиотека Србије, Београд

37.016:004(075.2)

ПАПИЋ, Милош, 1986-

Информатика и рачунарство 6 : уџбеник за шести разред основне школе / Милош Папић, Далибор Чукљевић. - 1. изд. - Београд : Вулкан издаваштво, Вулкан знање, 2019 (Крагујевац : Графостил). - 184 стр. : илустр. ; 28 cm

Тираж 12.000. - Речник појмова: стр. 178-181. - Библиографија: стр. 182.

ISBN 978-86-10-02412-8

1. Чукљевић, Далибор, 1985- [аутор]

COBISS.SR-ID 277249548

Министарство просвете, науке и технолошког развоја Републике Србије одобрило је овај уџбеник за употребу у школама решењем број:

650-02-00057/2019-07 од 9. 5. 2019.

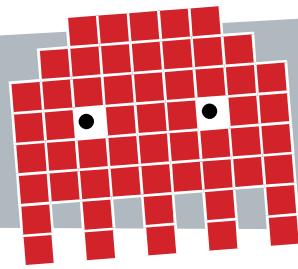
Copyright © Вулкан издаваштво, Вулкан знање, Београд

Ниједан део ове књиге, ни у целини ни у деловима, не сме се умножавати, прештамповати или на било који начин дистрибуирати, укључујући и фотокопирање, снимање и сл., без дозволе аутора и издавача. Сва права задржана.

САДРЖАЈ

1. ИНФОРМАЦИОНО-КОМУНИКАЦИОНЕ ТЕХНОЛОГИЈЕ	6
1.1. ДИГИТАЛНИ УРЕЂАЈИ И КОРИСНИЧКИ ПРОГРАМИ	7
1.2. УПРАВЉАЊЕ ДИГИТАЛНИМ ДОКУМЕНТИМА	13
1.3. РАД СА ДИГИТАЛНИМ СЛИКАМА	21
1.4. РАД СА ТЕКСТОМ.....	42
1.5. РАД СА ОСТАЛИМ ЕЛЕМЕНТИМА МУЛТИМЕДИЈЕ	61
1.6. МУЛТИМЕДИЈАЛНЕ ПРЕЗЕНТАЦИЈЕ	68
ТЕСТИРАЈ ЗНАЊЕ	80
2. ДИГИТАЛНА ПИСМЕНОСТ	82
2.1. ДИГИТАЛНИ УРЕЂАЈИ У МРЕЖНОМ ОКРУЖЕЊУ	83
2.2. ИНТЕРНЕТ СЕРВИСИ	89
2.3. ПРЕТРАЖИВАЊЕ ИНТЕРНЕТА	101
2.4. ПРАВИЛА БЕЗБЕДНОГ РАДА НА ИНТЕРНЕТУ	109
ПРОЈЕКТНИ ЗАДАТАК 1	114
ТЕСТИРАЈ ЗНАЊЕ	118
3. РАЧУНАРСТВО	120
3.1. ТЕКСТУАЛНО ПРОГРАМИРАЊЕ	121
3.2. ИНСТАЛАЦИЈА И РАЗВОЈНО ОКРУЖЕЊЕ ПРОГРАМСКОГ ЈЕЗИКА <i>PYTHON</i>	125
3.3. ОСНОВНЕ АРИТМЕТИЧКЕ ОПЕРАЦИЈЕ	131
3.4. ПРОМЕНЉИВЕ.....	135
3.5. ФУНКЦИЈЕ.....	143
3.6. РАД СА ТЕКСТУАЛНИМ ПОДАЦИМА.....	149
3.7. ЦРТАЊЕ У ПРОГРАМУ <i>PYTHON</i>	154
3.8. ПОНАВЉАЊЕ.....	160
3.9. ГРАНАЊЕ.....	166
ПРОЈЕКТНИ ЗАДАТАК 2	174
ТЕСТИРАЈ ЗНАЊЕ	176
РЕЧНИК ПОЈМОВА	178
ЛИТЕРАТУРА	182
ТЕСТИРАЈ ЗНАЊЕ – РЕШЕЊА.....	183

ВОДИЧ КРОЗ УЏБЕНИК



2.1. ДИГИТАЛНИ УРЕЂАЈИ У МРЕЖНОМ ОКРУЖЕЊУ

КЉУЧНЕ РЕЧИ

мрежа, интернет, глобална мрежа, локалне мреже, мреже широког распона, мрежни уређаји, хакер, антивирус, заштитни зид

Цивилизација не би достигла данашњи степен развоја да људи с временом нису научили да размењују информације и сарађују. Тако је и с рачунарима. Све погодности које пружа примена информационих технологија не би вределе када рачунари не би могли да буду **повезани** и да **комуницирају**. Група међусобно повезаних рачунара који деле ресурсе и информације назива се **рачунарска мрежа**.



Слика 2.1.1. Интернет је највећа рачунарска мрежа

Највећа рачунарска мрежа на свету је интернет, јер повезује различите територијално удаљене мреже на планети. Зато се каже да је интернет светска, односно **глобална рачунарска мрежа**. Често га називају и „мрежом свих мрежа“. Интернет омогућава комуникацију и размену информација између рачунара, pamетних телефона и других дигиталних уређаја широм света. Прошле године си научио/ла да ти је за везу с интернетом, поред дигиталног уређаја, потребан интернет провајдер (фирма која обезбеђује везу с интернетом). Провајдери могу пружити ову услугу на различите начине (преко ADSL-а, оптичког интернета, кабловског интернета и бежичног интернета – *Wireless*).

РЕЧНИК

Људи користе природне ресурсе (земљиште, воду, фосилна горива). И рачунари имају своје ресурсе. За рачунаре су ресурси све оно што им је потребно за рад (меморија, програми, штампач, скенер итд.).

ЗАНИМЉИВОСТ

Прва рачунарска мрежа на свету и претеча интернета је био ARPANET. Развиле га је америчко министарство одбране крајем 60-их година. ARPANET је повезивао све важне научне и војне институције.

83

Кључне речи су истакнуте на почетку лекције.

Речник – објашњења непознатих речи налазе се на страни на којој се помињу.

Занимљивост којом ћеш допунити своје знање.

2.1. ДИГИТАЛНИ УРЕЂАЈИ У МРЕЖНОМ ОКРУЖЕЊУ

Да би повезао рачунар у мрежу потребно је да имаш делове хардвера које до сада нисмо помињали. Они се убрљају у мрежну опрему. На претходној слици си видео/ла два таква уређаја (модем и рутер).

Модем (енгл. *Modem*) јесте уређај који повезује локалну мрежу с интернетом. Он омогућава рачунару да шаље и прима податке преко телефонске линије, каблова кабловске телевизије итд. Као што видиш на претходној слици, за LAN мреже је довољан један модем преко кога је цела мрежа повезана на интернет.

Рутер (енгл. *Router*) служи да повеже уређаје у мрежи или да повеже више локалних мрежа. Везу с интернетом коју омогућава модем, рутер дели свим уређајима у мрежи. Постоје типови модема који имају уграђен бежични рутер. Тада ти посебан рутер није потребан.

Мрежна картица је основни и најважнији елемент мрежне опреме. Она омогућава повезивање рачунара у мрежу. Често се назива и **мрежни адаптер** (енгл. *Network adapter*). Мрежна картица се повезује на одговарајући слот на матичној плочи. Матичне плоче данашњих pamетних телефона, таблета и лаптоп рачунара углавном имају већ уграђене мрежне картице за бежично повезивање на интернет. То само показује колико се у данашње време подразумева коришћење интернета.



Pаметни телефони имају опцију **Hotspot & tethering** (хотспот и привезивање) помоћу које се могу претворити у такозване **бежичне приступне тачке**. Тада се понашају као виртуелни рутери па други уређаји могу да користе њихову везу с интернетом. Преко pamетног телефона на којем је активирана ова опција, може се повезати на интернет до 10 других уређаја.



Сазнај више – подаци којима ћеш проширити своје знање о основној теми лекције.

Инфо плус – додатне корисне информације.

Вежба која ти помаже да боље савладаш градиво.

ИНФО ПЛУС

Mejllovi поред саме поруке могу садржати прилоге (енгл. Attachment) у виду слика, текстуалних и других фајлова. Једна порука може бити послата на више адреса одједном и може садржати више прилога.



За слање и пријем мејлова, можеш користити програме какав је Microsoft Outlook. Међутим, корисници се у данашње време најчешће опредељују за неке од бесплатних сервиса електронске поште. Најпознатији овакав сервис је Gmail. За дописивање преко неког од бесплатних сервиса, потребно је да имаш налог. Пре него што одлучиш да креираш налог, добро је да знаш да ће бити потребно да оставиш број телефона са којим ће налог бити потврђен. Саветујемо ти да налоге за било који сервис интернета отвараш увек у присуству родитеља.

ВЕЖБАЈ

Направи налог електронске поште. Можеш користити Gmail или неки други сервис.

За помоћ око креирања Gmail налога, погледај упутства на званичном сајту Google-а. Наћи ћеш их на страници <https://support.google.com/mail/answer/562567h=st>.

Ако ти то делује компликовано, погледај упутство које је један наставник информатике окачио на сајт за своје ђаци: <http://nastavnikinformatike.com/index.php/otvaranje-gmail-naloga>. На дотичној страници пронаћи ћеш и задатке за вежбање слика и примања електронске поште. Погледај и остале лекције и задатке који су окачени на сајт. Могу ти бити веома корисни!



Веб-адреса на којој су доступни садржаји за проширивање знања.

QR бар-код помоћу којег можеш да отвориш веб-страницу на pametnom telefonu или таблети.

Литература – списак веб-адреса где се налазе садржаји коришћени при писању текста.

Питања помоћу којих можеш да провериш усвојено градиво.

ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Јеси ли некада, не знајући да се то сматра пиратеријом, преузео/-ла нешто са интернета?
2. Шта је слично код ЦЕ-Це лиценци и појединих права коришћења у напредној претрази преко Google-а?
3. Да ли ЦЕ-Це лиценце подразумевају иста права коришћења као ставке у претрази слика приказане на слици 2.3.10?

ЛИТЕРАТУРА

Creative Commons Србије, Лиценце, доступно на сајту: http://creativecommons.org.rs/?page_id=74

Creative Commons, Share your work, доступно на сајту: <https://creativecommons.org/share-your-work/>

УКРАТКО

Трагање по вебу за информацијама помоћу претраживача представља посебан сервис интернета. Обавља се задавањем кључних речи (појмова претраге) које се уносе у поље за претрагу. Често се појам за претрагу куца директно у адресној линији браузера (прегледача). За унети појам можеш добити приказ само слика, вести, видеа, мапа итд. Поред тога, постоје и напредне технике претраге. ЦЕ-Це лиценце одређују услове под којима ће се ауторско дело користити. Неке лиценце су мање а неке више строге. Свака подразумева да мораш правилно навести име аутора ако ћеш користити његово дело. Ако си и сам/-а аутор неког дела, веома једноставно ћеш га заштитити одговарајућом ЦЕ-Це лиценцом на сајту Creative Commons-а (<https://creativecommons.org/share-your-work/>).

108

Укратко – кратак преглед лекције помоћи ће ти да поновиш градиво.

УВОДНА РЕЧ

Здраво! Пред тобом је уџбеник за шести разред основне школе из предмета Информатика и рачунарство. У петом разреду си поставио/-ла темеље а сада се пењеш степеницу више. Поново ћемо те провести кроз три тематске целине: ИКТ, Дигитална писменост и Рачунарство.

У шестом разреду ћеш унапредити своја знања из рада са текстом, звуком, видеом и мултимедијалним презентацијама. Савладаћеш елементарне технике обраде дигиталних слика и видећеш како изгледа рад са слојевима. ИКТ уређаје већ употребљаваш на одговоран и безбедан начин, али ћеш сада видети како то изгледа у мрежном окружењу. Поред правила безбедног рада на интернету, велику пажњу смо посветили и теми дигиталног насиља.

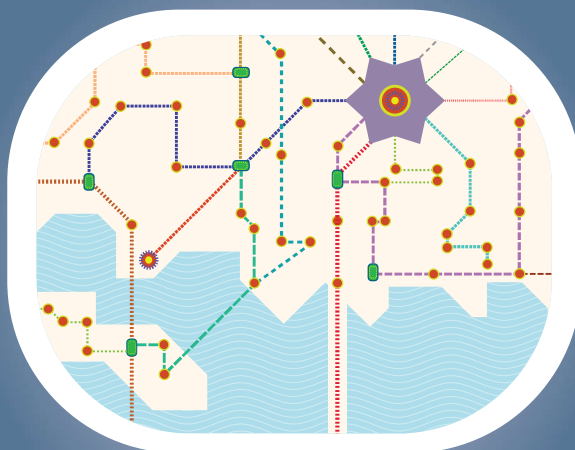
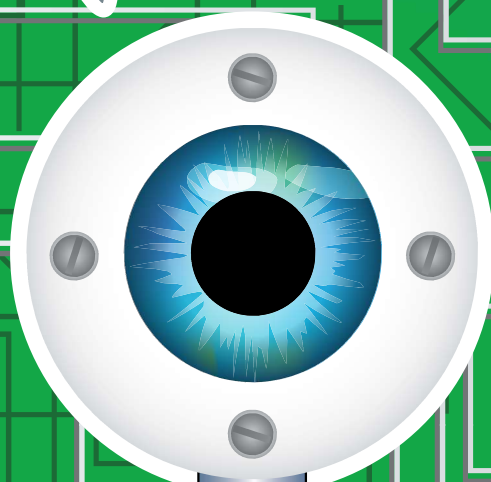
Захваљујући новом наставном програму, већ у шестом разреду ћемо те увести у свет текстуалног програмирања. Наиме, у оквиру области рачунарство упознаћеш се са једним од најпопуларнијих програмских језика данашњице – Пајтоном (енгл. Python). Кроз примере из програмирања у Пајтону, решаваћеш проблеме и из других предмета.

Трудили смо се да ти кроз уџбеник представимо најновије верзије софтвера, као и најсавременије технологије. Наставник ће и даље имати више менторску улогу. Усмераваће те да самостално кроз практичан рад прођеш кроз садржај уџбеника. Надамо се да си садржаје из петог разреда, поред рада у школи, савладавао/-ла и самостално код куће. Тај начин учења је веома значајан за усвајање знања и вештина из Информатике и рачунарства. Топло ти препоручујемо да и ове године наставиш да радиш исто.

Желимо ти много успеха у томе!

Аутори

1. ИНФОРМАЦИОНО- КОМУНИКАЦИОНЕ ТЕХНОЛОГИЈЕ



1.1. ДИГИТАЛНИ УРЕЂАЈИ И КОРИСНИЧКИ ПРОГРАМИ

КЉУЧНЕ РЕЧИ

дигитални уређаји, рачунарски систем, хардвер, софтвер, човек, Интернет ствар, правила, рачунарство у облаку



У петом разреду си се упознао/-ла са сложеним електронским машинама. Оне су намењене аутоматској обради података. Наравно, ради се о рачунарима. Видео/-ла си од чега се састоје и шта можеш помоћу њих да урадиш. Тада смо приказали најчешћа четири типа персоналних рачунара. То су стони (десктоп) рачунар, преносиви (лаптоп) рачунар, таблет и паметни телефон. Сви се састоје од видљивих и опипљивих, односно чврстих делова (хардвера). Само, за рад им је потребан и софтвер (и системски и апликативни). Ова два елемента чине нераскидиву целину звану **рачунарски систем**. Трећи елемент рачунарског система је човек (корисник рачунара).



о Слика 1.1.1. Човек као трећи елемент рачунарског система

Данас је све више такозваних паметних дигиталних уређаја. Поред паметног телефона и паметног телевизора, постоје и паметни сатови, камере, конзоле за видео-игре, фрижидери, рерне, веш-машине итд. Говори се о паметним аутомобилима, кућама, па чак и о паметним градовима. Ове уређаје чине паметним могућност умрежавања и повезивања на интернет, као и бројни сензори. Људи могу преко паметног телефона укључивати,

искључивати и подешавати уређаје у целој кући, чак и ако се у датом моменту налазе на другом крају планете. Ови уређаји се могу и сами укључивати и искључивати.



о **Слика 1.1.2.** Уређајима у паметним кућама је могуће управљати преко паметног телефона

ВЕЖБАЈ

Погледај на *YouTube*-у видео-снимке о Интернету ствари и паметним кућама.

ИНФО ПЛУС

Сви умрежени паметни уређаји, заједно са системом за прикупљање и обраду података, чине **Интернет ствари** (енгл. *Internet of Things*, скраћено *IoT*). Предвиђа се да ће до 2020. године на Интернет ствари бити бежично повезано више од 30 милијарди уређаја.

Правила руковања дигиталним уређајима односе се на све врсте уређаја – не само на рачунаре. Најважнија општа правила која важе за све врсте дигиталних уређаја јесу:

1. правилно укључивање и искључивање;
2. правилно руковање хардвером;
3. правилна употреба софтвера;
4. правилно одлагање е-отпада.

ИНФО ПЛУС

Можда си приметио/-ла да сви мобилни оператери нуде најновије паметне телефоне за само 1 динар уз уговоре од две године. Да ли си размишљао/-ла зашто је то тако? Разлог је једноставан. Након две године уређаји почињу да се кваре и излазе из моде. Корисници због тога уређај мењају новијим моделом и тако сваке две године. Ово је једна од одлика „потрошачког друштва“. То није случај само са паметним телефонима, већ са свим дигиталним уређајима. Тако се гомилају огромне количине е-отпада. Зато је важно да се придржаваш правила и ограничиш време употребе дигиталних уређаја. О одликама потрошачког друштва и последицама по животну средину можеш детаљније разговарати са наставницом или наставником грађанског васпитања.

Хардвер

Два основна елемента хардвера десктоп рачунара јесу кућиште (са својим **компонентама**) и периферни односно спољни уређаји (**периферије**). Надамо се да си успешно савладао/-ла употребу тастатуре. У твоје вештине коришћења миша или тачпеда (енгл. *Touchpad*) не сумњамо. Сигурно се сећаш занимљивих илустрација најзначајнијих рачунарских компонента у уџбенику за пети разред. На следећој слици је компонента за коју се каже да је „мозак“ рачунара. Процесор код лаптопа и паметног телефона је мањи од процесора код десктоп рачунара.

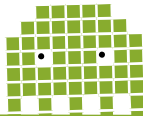


о Слика 1.1.3. Процесори за лаптоп и паметни телефон

САЗНАЈ
ВИШЕ

Данас се за рачунаре најчешће користе процесори са 2 или 4 језгра. Постоје процесори и са чак 12 језгара. Процесор са више језгара брже обрађује податке.

ВЕЖБАЈ



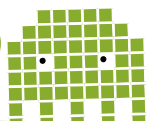
Пронађи на интернету информације о томе који су најновији, а који најзаступљенији процесори код паметних телефона.

Да ли се сећаш која рачунарска компонента служи за дуготрајно а која за краткотрајно памћење односно меморисање података и информација? Капацитет и једне и друге се изражава у гигабајтима (GB). Погледај наредну слику. Шта би одговорио/-ла? Ако не можеш да се сетиш, погледај лекцију 1.2. у уџбенику за пети разред.



о Слика 1.1.4. Хард-диск, SSD уређај и RAM меморија

ВЕЖБАЈ



Уколико постоје услови, било би корисно да вам наставник/-ца извади матичну плочу и покаже како се скидају, а потом и повезују модули меморије, процесор, графичка и друге картице.

САЗНАЈ ВИШЕ

Постоје апликације које приказују ознаке и марке свих компонената у кућишту. Једна од таквих апликација је *Spessy*. Бесплатна је и може се презети са веб-адресе www.ccleaner.com/specsy.



Радна меморија (RAM) веома је значајна за бржи рад рачунара. Програми и апликације покретаће се брже и радити боље када рачунар има више меморије. За *Windows 10* је пожељно да рачунар има макар 4 GB радне меморије.

На матичну плочу (енгл. *Motherboard*) за коју се каже да је „срце“ рачунара спајају се све рачунарске компоненте и периферије. За то се употребљавају подножја (сокети), слотови и прикључци (портови). Подножје односно **сокет** (енгл. *Socket*) служи да се на њега постави процесор. Преко процесора се ставља хладњак с вентилатором (енгл. *Cooler*).

За радну меморију постоје обично два или четири **слота** на матичној плочи. Капацитет меморије можеш проширити додавањем комада радне меморије (слика 1.1.4. десно), тзв. **модула**. Поред слотова за меморију, постоје слотови за звучну, мрежну, ТВ и друге картице. Неке од компонената могу бити уграђене (интегрисане) у саму плочу. То је често случај са графичким картицама. Без обзира на то, на плочи увек постоји један слот за графичку картицу у случају да она није интегрисана.

Прикључци односно **портови** користе се за повезивање периферних уређаја. Када се матична плоча монтира у кућиште, портови се налазе са његове задње стране.

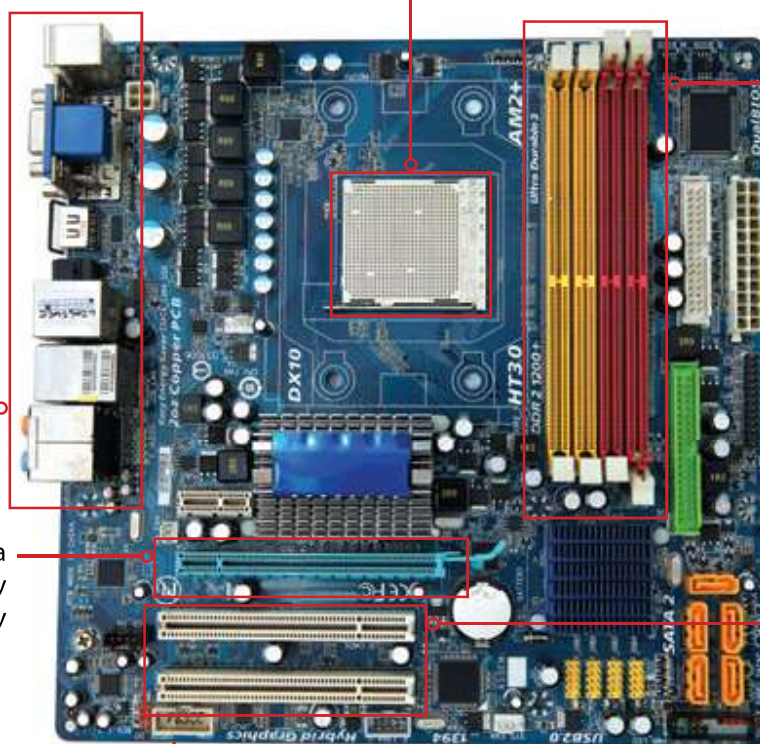
Прикључци (портови)
за периферне уређаје

Слот за
графичку
картицу

Подножје (сокет)
за процесор

Слотови за
меморију

Слотови
за друге
картице



о Слика 1.1.5. Делови матичне плоче

Софтвер

Поред познавања свих рачунарских компонената и периферија, до сада си потпуно оспособљен/-а за правилан и безбедан рад у **оперативном систему** *Windows 10*. Кроз *Windows Settings* умеш да прилагодиш сва подешавања својим жељама и потребама. Преко *Windows* безбедносног центра (*Windows Defender Security Center*) можеш и требало би да знаш да заштитиш свој налог од вируса и осталих претњи по софтвер твог уређаја.

Што се тиче **апликативног софтвера**, до сада си савладао/-ла основе рада у *Paint*-у, *Word*-у, *PowerPoint*-у, као и у неколико апликација које постоје у *Windows 10* оперативном систему (*Photos*, *Groove Music*, *Voice Recorder*, *Movies & TV* и *Camera*). Иконице наведених апликација су у истом редоследу приказане на слици 1.1.6. На примерима наведених апликација (корисничких програма) научио/-ла си да креираш дигиталну слику, текстуални документ и мултимедијалну презентацију. Научио/-ла си да снимаш и репродукујеш аудио и видео фајлове. У поглављима која следе проширићеш постојећа и усвојити нова знања и вештине.



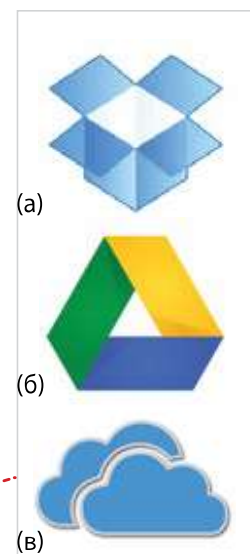
Слика 1.1.6. Апликације с којима си се упознао/-ла у петом разреду

Област информационо-комуникационих технологија стално напредује. Једна од грана која се убрзано развија је **рачунарство у облаку** (енгл. *Cloud computing*). У петом разреду си сазнао/-ла за апликације у облаку (софтвер као сервис). Пример софтвера као сервиса (енгл. *Software as a Service*, скр. *SaaS*) јесте онлајн верзија програмског пакета *Microsoft Office – Office 365*.

Поред апликација у облаку, постоје и **складишта у облаку** (енгл. *Cloud storage*). То су одређене локације или места на интернету где можеш да чуваш своје податке. Најпознатија складишта у облаку су *DropBox*, *Google* диск (енгл. *Google Drive*) и *OneDrive* који је инсталиран уз *Windows 10*. Она имају огромне меморијске капацитете за чување података. *Google Drive* има 15 GB бесплатног меморијског капацитета (ово укључује и сву електронску пошту на *Gmail*-у). *OneDrive* има 5 GB бесплатног меморијског капацитета. Онлајн складиштима можеш приступити с било које локације и с било ког уређаја. Наравно, мораш бити повезан/-а на интернет. Она су погодна за чување **резервних копија** података (енгл. *Backup*). Ипак, препоручујемо ти да поред тога резервне копије и даље чуваш на екстерним меморијама или на CD/DVD-у.

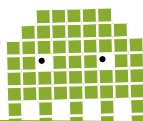


Најновија верзија програмског пакета *Microsoft Office (Office 2019)* можда ће бити последња десктоп верзија *Office*-а. Ако се то догоди, корисници ће у будућности моћи да употребљавају само верзију у облаку – *Office 365*. *Microsoft* би овим потезом означио коначни прелаз на рачунарство у облаку.



Слика 1.1.7. Логотипи најпознатијих апликација за рад са складиштима у облаку: а) *DropBox*, б) *Google* диск и в) *OneDrive*

ВЕЖБАЈ



1. На десктопу креирај два нова фолдера. Назови их **Levi** и **Desni**.
2. Селектуј све фајлове и фолдере који се налазе у фолдеру **Documents** и копирајих у фолдер **Levi**.
3. Тастерском пречицом **Win + E**, отвори два прозора *File Explorer*-а. Један закачи на леву, а други на десну половину екрана.
4. Испробај сваки од наведених начина за копирање тако што ћеш све фајлове који се налазе у фолдеру **Levi** појединачно копирати у фолдер **Desni**.
5. Испробај све начине за брисање фајлова и фолдера тако што ћеш појединачно обрисати све фајлове из фолдера **Levi**.
6. Испробај све начине за премештање фајлова и фолдера тако што ћеш све што се налази у фолдеру **Desni** појединачно преместити у фолдер **Levi**.
7. На крају обриши фолдер **Desni** и испразни корпу за отпатке.



УКРАТКО

Фајл менаџер који се користи у оперативном систему *Windows 10* је *File Explorer*. У њему можеш креирати, премештати, копирати и брисати фајлове и фолдере. Можеш им мењати називе, а за њихово проналажење можеш користити опцију претраге. Да би било шта могао/-ла да урадиш са фајлом или фолдером, потребно је да буде селектован. Можеш користити леви клик у комбинацији са тастерима **Ctrl** и/или **Shift**, као и опције са **Home** картице (**Select All**, **Select None** и **Invert Selection**). Премештање и копирање можеш вршити на различите начине. Пожељно је да усвојиш тастерске пречице **Ctrl+X** за исецање (**Cut**), **Ctrl+C** за копирање (**Copy**) и **Ctrl+V** за лепљење (**Paste**). Ако отвориш два прозора *File Explorer*-а и поставиш их један до другог, наведене акције ће ти бити олакшане. Складишта у облаку су одређене локације или места на интернету где можеш да чуваш своје податке. Апликације за рад са њима ти омогућавају да својим фајловима приступаш са било ког уређаја. Када направиш измене на фајлу, оне ће бити синхронизоване и на осталим уређајима.

1.3. РАД СА ДИГИТАЛНИМ СЛИКАМА

КЉУЧНЕ РЕЧИ

Paint, *GIMP*, алатке, селекције, обрада, филтери, слојеви

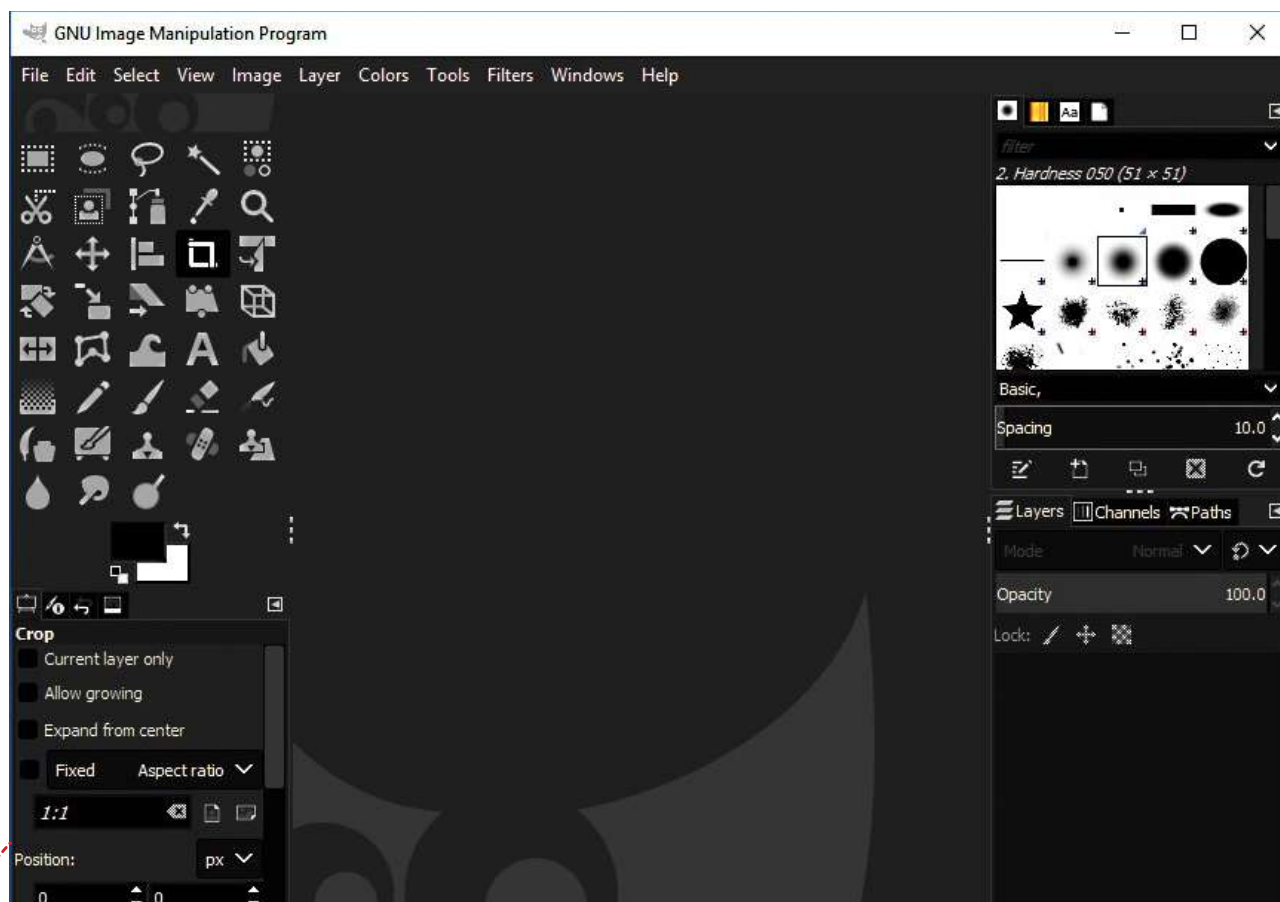


У петом разреду си упознао/-ла *Paint*. То је један од најпознатијих програма за цртање и обраду дигиталних слика. Он је лако доступан јер се инсталира уз *Windows*. Ипак, некад ти може затребати напреднија обрада слика. За то ћеш користити алатке којих у *Paint*-у нема. Зато ћемо те у овом поглављу упознати с програмом **GIMP**.

GIMP је бесплатан. Има много корисних опција које ћемо приказати на верзији 2.10.8 овог програма. *GIMP* преузимаш с веб-странице <https://www.gimp.org/>. Након инсталације, можеш га покренути преко путање **Start => All apps => G => GIMP 2.10.8**. или куцајући **GIMP** у претрази. По покретању, *GIMP* изгледа овако:



Професионалци који обрађују слике, користе *Photoshop*.



о Слика 1.3.1. Прозор програма *GIMP* након покретања

Креирање нове слике

Нову слику креираш тако што изабереш опцију **New** из **File** менија или преко тастерске пречице **Ctrl + N**. Након тога се приказује прозор **Create a New Image**.

Опције за избор оријентације слике (портрет, пејзаж)

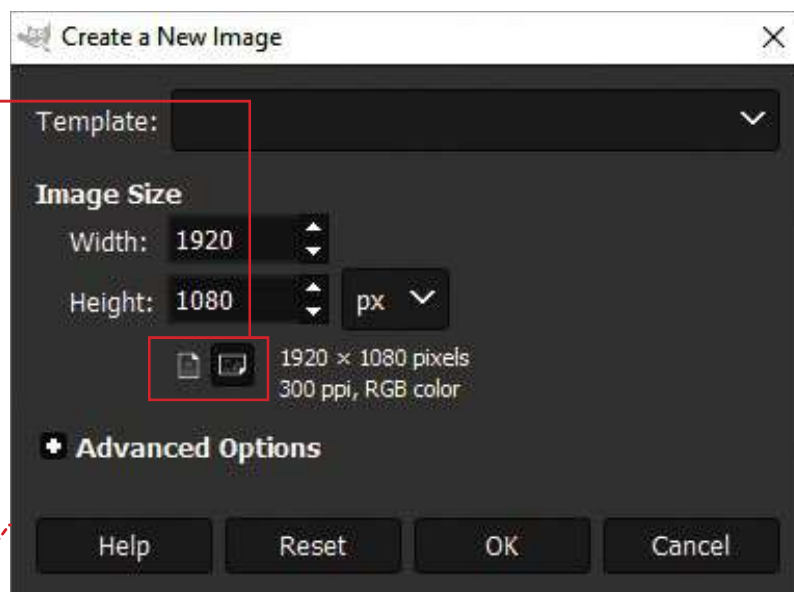


(a)



(б)

о **Слика 1.3.3.** (a) Усправан положај слике (портрет)
(б) Водораван положај слике (пејзаж)



о **Слика 1.3.2.** Прозор за креирање нове слике

У овом прозору дефинишеш димензије нове слике. Уносиш ширину слике (**Width**) и висину (**Height**). Подразумеване димензије су 1920 x 1080 *px* (пиксела). Јединицу мере мењаш у падајућем менију који позиваш кликом на стрелицу надолу поред ознаке пиксела. Поред пиксела, можеш одабрати инче, милиметре, центиметре, метре или друге понуђене јединица. Када одредиш димензије, можеш изабрати оријентацију слике – портретну (енгл. *Portrait*) или пејзажну (енгл. *Landscape*). Кликом на **OK** све је спремно за цртање.

ЗАНИМЉИВОСТ

Појмови портрет и пејзаж којима се означава положај слике (или оријентација стране при раду с текстом) преузети су из сликарства. Портрет означава приказ лика и фигуре одређеног човека. Најпознатији портрет је Мона Лиза, чувено ремек-дело ренесансног сликара Леонарда да Винчија. Пејзаж је приказан предео. За разлику од портрета, пејзажи се најчешће раде на платну које је шире него дуже.

ИНФО ПЛУС

Често се каже да је пиксел (енгл. *Pixel* скраћено од *Picture Element*) најмања тачка на слици. Ово је само делимично тачно јер је пиксел квадратног облика. Када креираш нову слику у *GIMP*-у, она подразумевано има 1920 пиксела по хоризонтали и 1080 пиксела по вертикали. Укупан број пиксела на слици одређује њену јасноћу, односно оштрину. Мера за оштрину слике је резолуција. Резолуција од 1920x1080 пиксела представља висок стандард у квалитету слике. Често се о њој говори као о **Full HD** односно пуној високој резолуцији (енгл. *Full High Definition Resolution*, скр. *FHD*). Можда си негде видео/-ла ознаку **1080p**. То је такође ознака *Full HD* резолуције. Веће резолуције од ње су *Quad HD* (2560x1440 px), *4K Ultra HD* (3840x2160 px) и *8K Ultra HD* (7680x4320 px).



(a)

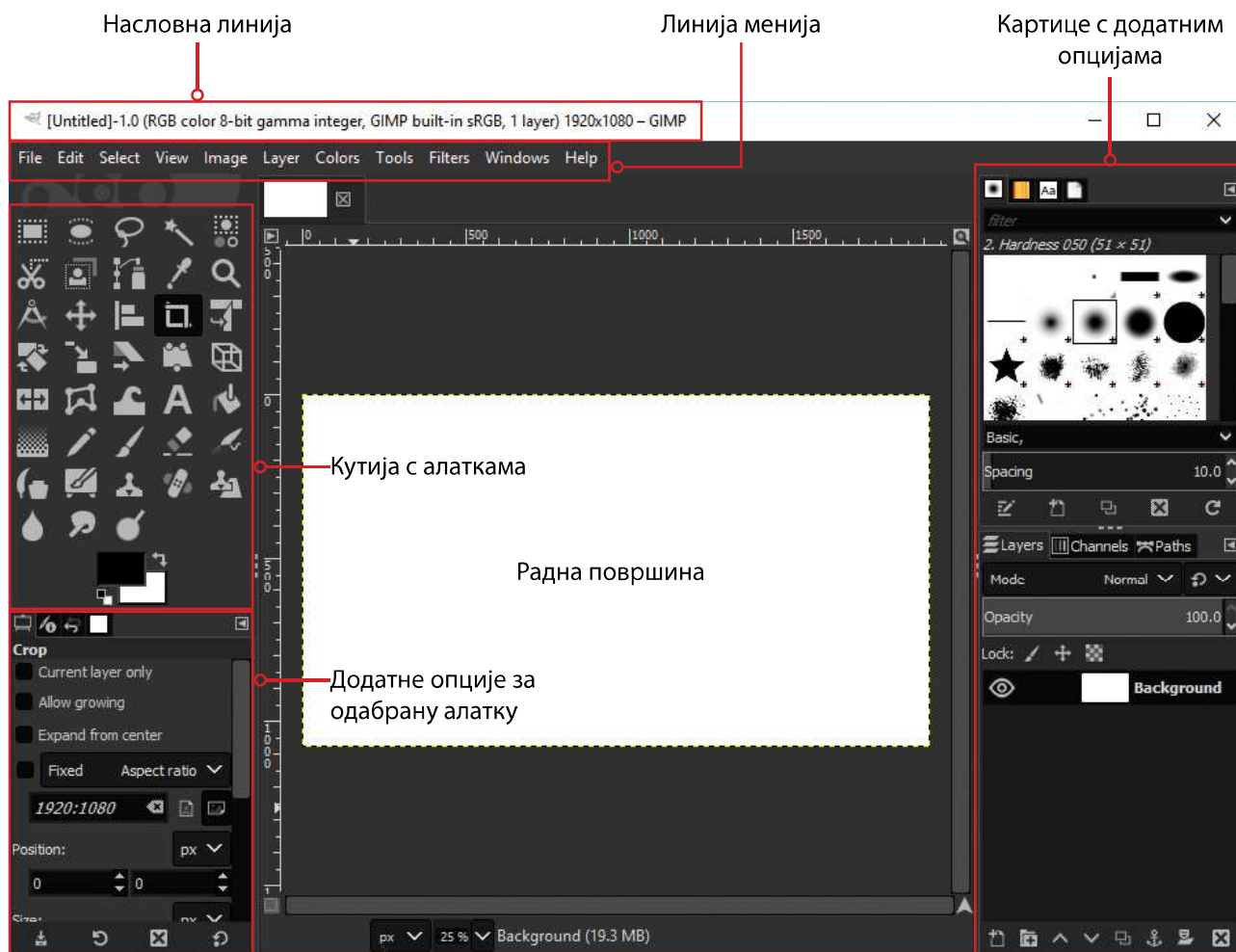


(b)

о **Слика 1.3.4.** (a) Поређење квалитета слике у *Full HD* и *4K Ultra HD* резолуцији и (b) Поређење величине слика различите резолуције

Радно окружење GIMP-а (слика 1.3.5) чине следећи елементи:

1. насловна линија (енгл. *Title bar*) на којој се налазе дугмад за минимизирање, максимизирање и затварање (енгл. *Minimize*, *Maximize* и *Close*). Назив документа је на левој страни. На почетку, документ се увек зове **Untitled** (без назива). То ће се променити када сачуваш свој цртеж;
2. линија менија (енгл. *Menu bar*);
3. кутија с алаткама (енгл. *Toolbox*) која је подељена на два дела. У горњој половини су алатке којима се обављају различите радње у програму. У доњој половини су опције које се односе на тренутно одабрану алатку;
4. картице (енгл. *Tabs*) с додатним опцијама;
5. радна површина.



о Слика 1.3.5. Радно окружење програма GIMP

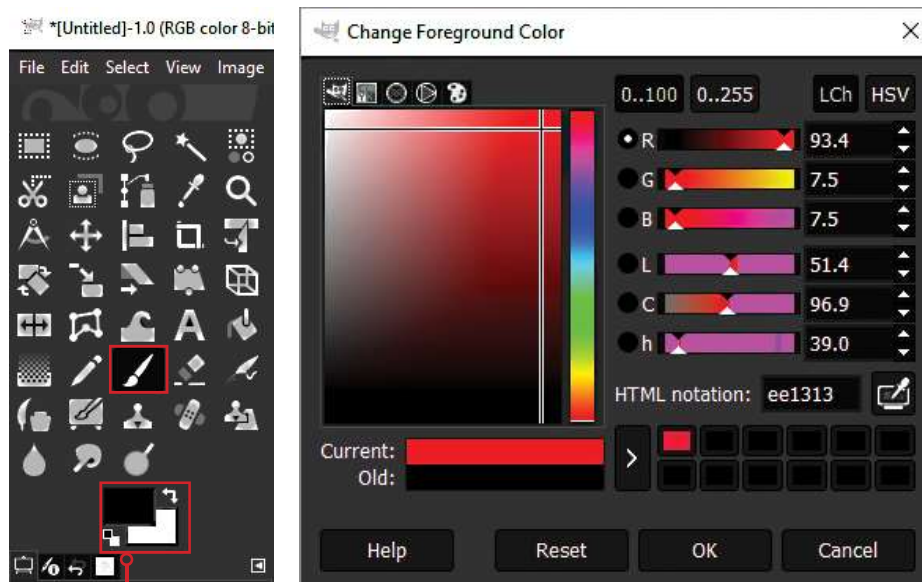
Алатке за цртање

Све алатке за цртање налазе се у менију **Tools => Paint Tools**. Најчешће коришћене налазе се и у кутији с алаткама. Поред тога, сваку можеш одабрати и помоћу тастерске пречице. Дакле, постоје по три начина за избор алатке у *GIMP*-у. Алатке служе за:

- **Pencil** (оловка) – цртање линија с тврдим ивицама;
- **Paintbrush** (четкица) – цртање линија меких ивица;
- **Eraser** (гумица) – брисање линије те делова цртежа или слике;
- **Airbrush** (спреј) – цртање веома меких линија;
- **Ink** (мастило) – писање или цртање калиграфским стилем;
- **Bucket Fill** (попуна бојом) – попуњавање подручја бојом;
- **Gradient** (градијент) – попуњавање подручја градијентом.

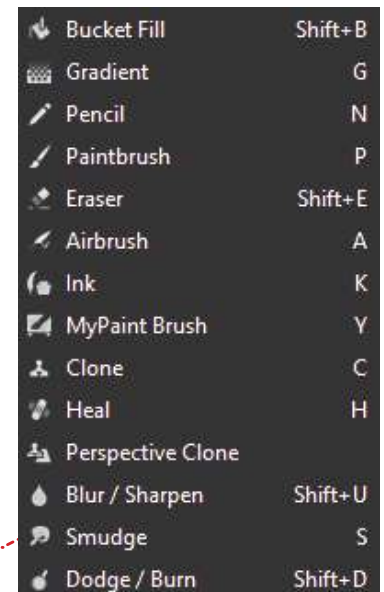
Свака наведена алатка има мноштво додатних опција. Помоћу њих бираш величину, облик, угао под којим ћеш цртати, затим боју, прозирност као и многе друге могућности које ти могу бити веома корисне.

Сада ћемо ти показати основне кораке за коришћење алатки за цртање. Исти кораци се примењују код већине алатки. Послужићемо се четкицом. Одабери алатку **Paintbrush** из кутије с алаткама. Затим одабери боју којом ћеш цртати. Подразумевана боја за све алатке је црна. Двокликом на црни правоугаоник у дну кутије с алаткама, отвара се прозор **Change Foreground Color** (промени боју првог плана). Ту изабери боју и кликни ОК.



Слика 1.3.7. Одабир алатке и боје

Опција за избор боје првог плана или позадине



Слика 1.3.6. Алатке за цртање у подменију Paint Tools менија Tools

РЕЧНИК

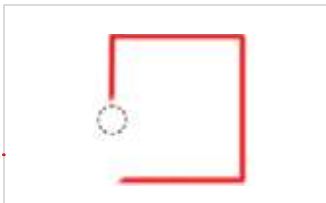
Градијент (преливање боја) представља постепени прелаз између две боје или више њих.

ИНФО ПЛУС

За сваку алатку у *GIMP*-у можеш одабрати боју првог плана (енгл. *Foreground*) и боју позадине (енгл. *Background*). Увек је активна боја квадрата који се налази на врху. Да би променио/-ла боју кликни на стрелицу која се налази уз квадрате.




о Слика 1.3.8. Две варијанте цртања четкицом



о Слика 1.3.9. Брисање дела цртежа помоћу гумице

Постоје два начина рада с четкицом. Први је да црташ слободном руком као у програму *Paint*. По другом, црташ уз притиснут тастер **Shift**. Тако ћеш начинити праве линије до места на које си кликнуо/-ла.

 Брисање делова слике се обавља гумицом – **Eraser**. Пошто изабереш ову алатку, можеш јој у додатним опцијама повећати величину (**Size**). На тај начин ћеш брже обрисати цртеж. Ако пак треба да обришеш најситније детаље на цртежу, величину гумице можеш смањити.

Алатке за селектовање

За разлику од програма *Paint*, селектовање делова слике у програму *GIMP* се обавља на различите начине. За то постоји чак седам алатки (слика 1.3.10. лево). Помоћу њих се бира део слике који ће се обрађивати. Када направиш селекцију (слика 1.3.10. десно), све алатке и опције можеш применити само над њом.

Селектовање се може вршити превлачењем алатки преко жељене области слике. При томе алатка може бити четвртастог или кружног (елипсастиог) облика. Можеш и сам/-а нацртати оквир селекције, можеш извршити селекцију према одређеној боји итд. Програм може и сам понудити да изабере област коју сматра различитом од осталих. Довољно је да одговарајућом алатком једном кликнеш у ту област. Алатке за селектовање наћи ћеш у подменију **Selection Tools** менија **Tools**. Најчешће се употребљавају:



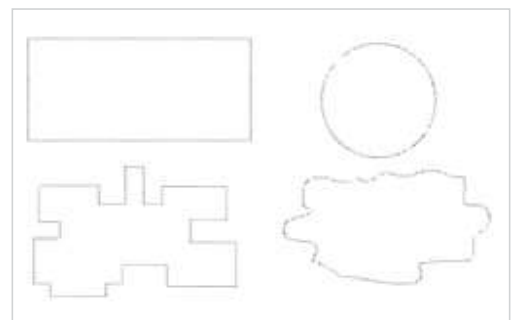
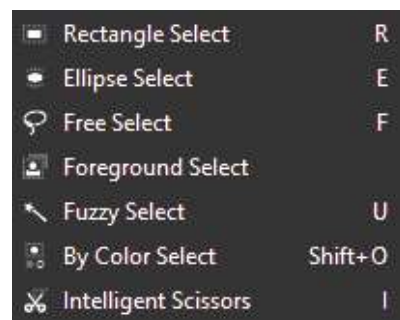
Rectangle Select (правоугаона селекција) – алатка за селектовање области правоугаоног облика.



Ellipse Select (елипсаста селекција) – алатка за селектовање области елипсастиог облика.



Free Select (слободна селекција) – алатка којом селекујеш област облика какав сам/-а нацрташ. Линије могу бити праве (као да су цртане лењиром) или као да су нацртане слободном руком.



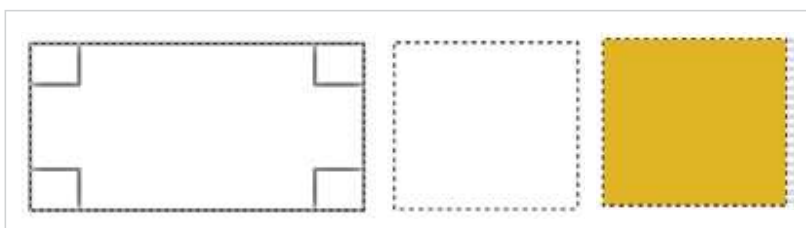
о Слика 1.3.10. Алатке и врсте селектовања

Након селектовања, елементе слике можеш преместити или копирати. За ово користиш добро познате опције – **Cut**, **Copy** и **Paste**. Можеш их позвати из менија **Edit**, или можеш употребљавати познате тастерске пречице (**Ctrl + X**, **Ctrl + C** и **Ctrl + V**). Ако се не сећаш која чему служи, врати се на претходно поглавље.

Цртање и бојење облика

Цртање облика у програму *GIMP* је мало другачије од цртања у *Paint*-у и *Word*-у. Не постоје понуђени облици као у поменутих програмима. За цртање облика се користе приказане алатке за селектовање.

Прво изабери алатку за селектовање, а затим начини облик селекције (слика 1.3.11. лево). Када направиш жељени облик, можеш га променити превлачењем сваке ивице у жељеном смеру. Ако нешто ниси добро урадио/-ла, притисни тастер **Esc** и прекини цртање. Када ти нацртани облик одговара, притисни **Enter** (слика 1.3.11. средина). Тада облик постаје коначан и не можеш му више мењати димензије. Након тога нацртани облик можеш да обојиш. Узми алатку **Bucket Fill**, одабери боју у **Toolbox**-у и кликни унутар селекције (слика 1.3.11. десно).



Слика 1.3.11. Цртање и бојење облика

Облицима нацртаним на овај начин можеш додати контуру и обојити је. Када нацрташ облик, кликни на њега десним тастером миша. Из менија **Select** изабери опцију **Border** (граница). У прозору **Border Selection** (граница селекције) (слика 1.3.12) упиши дебљину контуре у пикселима. Дебљина контуре мора бити већа од 0 px да би контура била видљива (види слику 1.3.11). Јединицу можеш променити кликом на стрелицу надолу поред ознаке пиксела (px). Потом кликни **OK**.

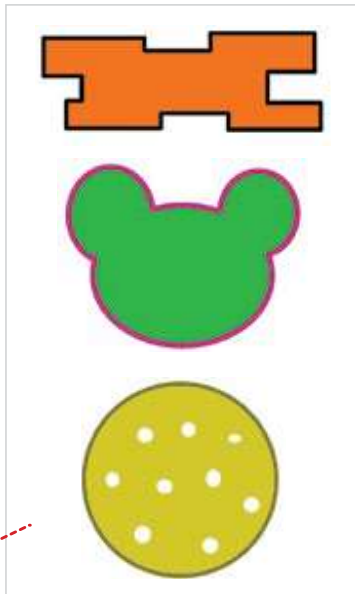
На крају помоћу алатке **Bucket Fill** обоји контуру облика.



Слика 1.3.12. Дефинисање дебљине и врсте контуре



Слика 1.3.13. Додавање и бојење контуре облика



Слика 1.3.14. Додавање и одузимање делова облицима

Комбинацијом претходна два поступка, можеш нацртати облик и подебљати му контуру. Затим можеш обојити облик једном, а контуру другом бојом.

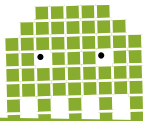
Такав пример је показан на слици 1.3.14. Ту је приказана још једна корисна опција *GIMP*-а. Наиме, облике можеш да комбинујеш. То делује као израда уније или пресека скупова. Након једног нацртаног облика држи притиснут тастер **Shift** и настави да их црташ. Када притиснеш **Enter**, облици ће бити спојени у један. Ово је изводљиво искључиво када је одабрана алатка за селектовање. Не можеш нацртати кружни облик па му додати правоугаони. Од облика можеш и да одузмеш нацртани део. Само држи тастер **Ctrl** док црташ облик који хоћеш да одузмеш.

На овај начин можеш нацртати много тога. Труди се да будеш креативан/-на. Комбинујући облике и алатке за цртање, ми смо нацртали кућицу.



Слика 1.3.15. Кућица нацртана у програму *GIMP*

ВЕЖБАЈ



Покрени *GIMP* и креирај нову слику ширине 297 mm и висине 210 mm. Покушај да израдиш слику коју смо ми направили. Сачувај фајл у фолдеру **Documents** под називом *Moja prva slika u GIMP-u*. Сада извези (**Export**) фајл на исту локацију под истим називом али у *JPEG* формату.

Чување слике

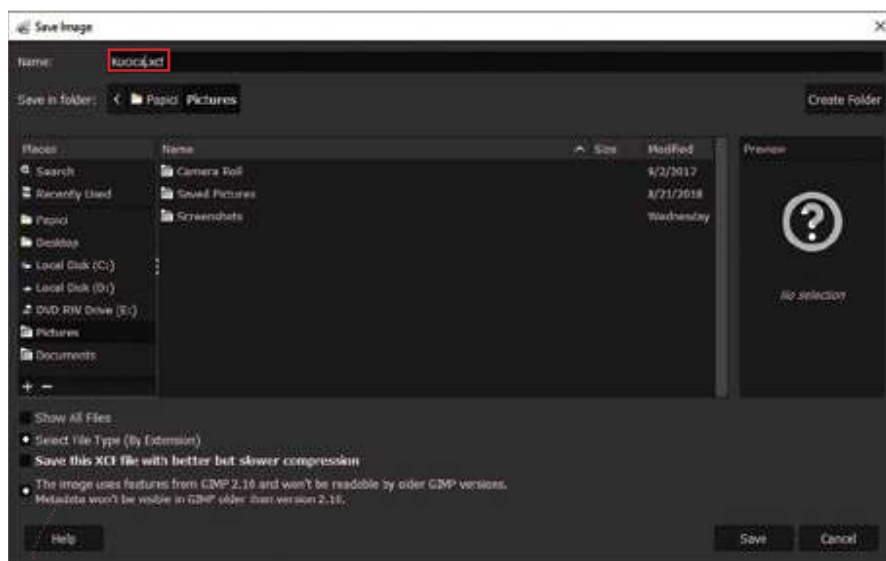
Сада ћемо нацртану слику сачувати. Сигурно мислиш да је то лако и да већ знаш то да урадиш. Делимично си у праву, пошто је чување слике у програму *GIMP* другачије него у програму *Paint*.

На слици 1.3.16. је приказан прозор који позиваш кликом на опције **Save** или **Save As**. У пољу **Name** уписујеш назив фајла и бираш локацију. Ако желиш да промениш тип фајла, видећеш да то није могуће.

Помоћу опције **Save/Save As** могуће је снимити слику искључиво у *GIMP*-овом изворном **XCF** формату. Сlike у **XCF** формату могу се отворити само употребом *GIMP*-а. Не можеш их прегледати у апликацији *Photos* ни делити преко друштвених мрежа. Не могу се ни постављати као позадина на десктопу (енгл. *Desktop Background*).

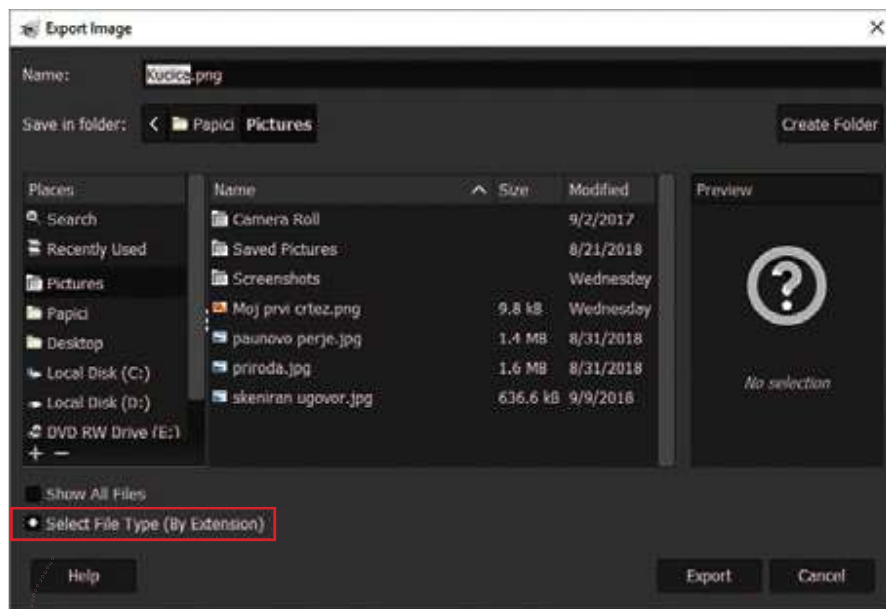
Иконице **XCF** фајлова сачуваних у *GIMP*-у, изгледају овако:





о Слика 1.3.16. Чување слике помоћу опције Save

Да би сачувао слику у неком другом формату, мораш да је **извезеш** односно **експортујеш**. Потребно је да кликнеш на ставку **Export** или **Export As** у менију **File**. Отвориће се прозор као на следећој слици. Прозор је готово исти као претходни с тим што у менију **Select File Type** (изабери врсту датотеке) можеш одабрати да фајл буде сачуван под другом екстензијом. Подсећамо те да су најчешће екстензије графичких фајлова **.jpg**, **.tiff**, **.png**, **.bmp**, **.gif** итд.



о Слика 1.3.17. Чување слике помоћу опције Export

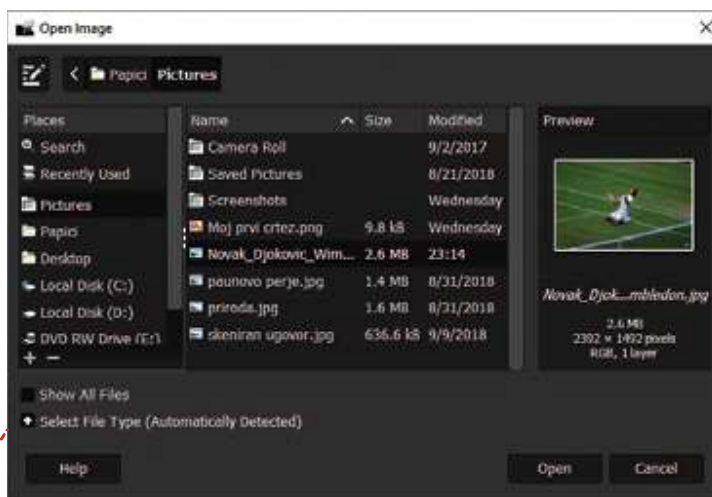
ВЕЖБАЈ

Нацртај склопни цртеж саобраћајног знака по избору. Означи делове предмета позиционим бројевима. За пример погледај наредну слику.

Обрада слика

Иако смо ти прво показали како да креираш нову слику, *GIMP* је пре свега намењен обради слика. Сврха обраде је прилагођавање слике њеној намени. Обрада слика обухвата промену величине, одсецање делова, употребу филтера, уметање текста итд.

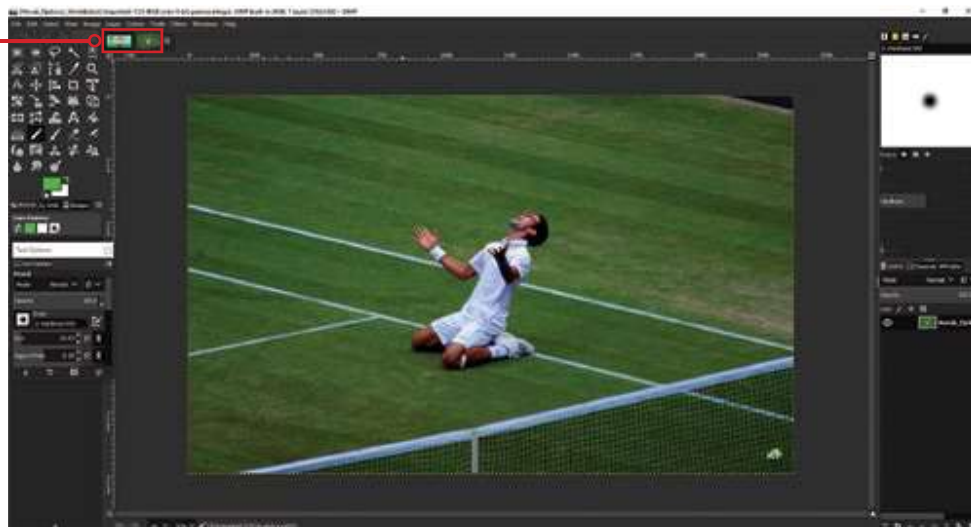
Да би могао/-ла приступити обради слике, мораш прво да је отвориш. **Отварање** се обавља као и у већини других програма – изабере се опција **Open** у менију **File** или се употреби тастерска пречица **Ctrl + O**.



о Слика 1.3.18. Прозор за отварање слике

У прозору **Open image**, из листе одабери слику коју ћеш отворити и кликни на **Open**. Након тога ће слика бити приказана на радној површини.

Картице
отворених
слика



о Слика 1.3.19. Отворена слика

У програму *GIMP* се може истовремено отворити више слика. Оне се налазе свака под својом картицом у врху прозора (испод линије менија). Одабрана картица се разликује од осталих по томе што је истакнута. Сliku, тј. картицу која је активна, можеш **затворити** притиском на дугме **x** с њене десне стране. Исти ефекат постижеш помоћу тастерске пречице **Ctrl + W** или избором опције **Close View** из менија **File**. Ако хоћеш да затвориш све отворене слике, у истом менију одабери опцију **Close All**.

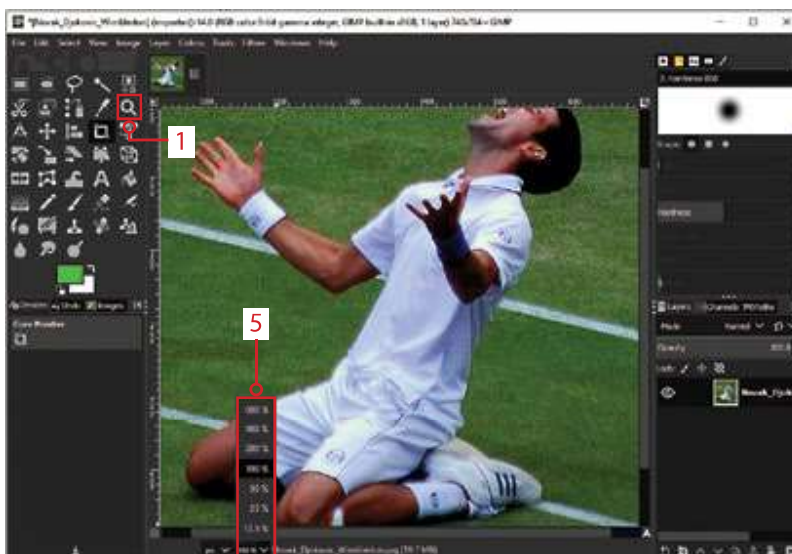
Зумирање и кретање по слици

Када отвориш слику у *GIMP*-у, величина приказа је подешена тако да слика стане на радну површину. Приликом обраде слике, често је потребно увећати (зумирати) и изблиза видети и обрадити неки детаљ. Неки од начина за увећавање делова слике су:

1. Употреба алатке за зумирање **Zoom Tool** из **Toolbox**-а. Превлачењем алатке преко одређеног дела он бива зумиран (**Zoom In**). Када држиш тастер **Ctrl** приликом превлачења, величина приказа се смањује (**Zoom Out**).
2. Окретање точића на мишу уз притиснут тастер **Ctrl**. Окретањем точића нагоре, приказ дела слике се повећава, док се окретањем надолу смањује.
3. Притискање тастера са знаком + (**Zoom In**) и знаком – (**Zoom Out**).
4. Притискање нумеричких тастера, од 1 до 5. При чему је 1 = 100% увећања, 2 = 200%, 3 = 300% итд.
5. Одабир или уписивање жељеног процента увећања/умањења из падајућег менија на дну прозора.
6. Зумирати се наравно може и преко менија **View**.



Зумирање помоћу точића на мишу и тастера **Ctrl** примењује се и у *Word*-у, *PowerPoint*-у и многим другим програмима.



Слика 1.3.20. Зумирана слика



Професионални дизајнери приликом обраде слика за кретање и зумирање најчешће користе точкић на мишу.

За кретање по слици можеш користити вертикални и хоризонтални клизач. Исто је и у свим другим програмима. Лакши и бржи начин је да се по слици крећеш држећи притиснут точкић на мишу и померајући миша.

Промена величине слике



За промену величине слике користи се алатка **Scale Tool**. Она се налази у менију **Tools => Transform Tools => Scale**. Свакако је можеш одабрати и у **Toolbox**-у. Након избора алатке, кликни на слику. Појавиће се провидни квадрати на угловима и срединама страница слике. Њиховим превлачењем ка унутрашњости слике или ка њеној спољашњости, можеш ручно променити димензије слике.

Такође ћеш у горњем десном углу добити прозорчић у који можеш уписати нове димензије. Исти прозорчић можеш добити и кликом на ставку **Scale Image** у **Image** менију.

Величина слике може се изражавати у готово свим стандардним јединицама: у пикселима, инчима, милиметрима, сантиметрима, метрима, процентима и др. Ако се сећаш опције **Resize** у **Paint**-у, видиш да је поступак за промену величине слике готово исти – само је изглед прозора и назив алатке другачији.

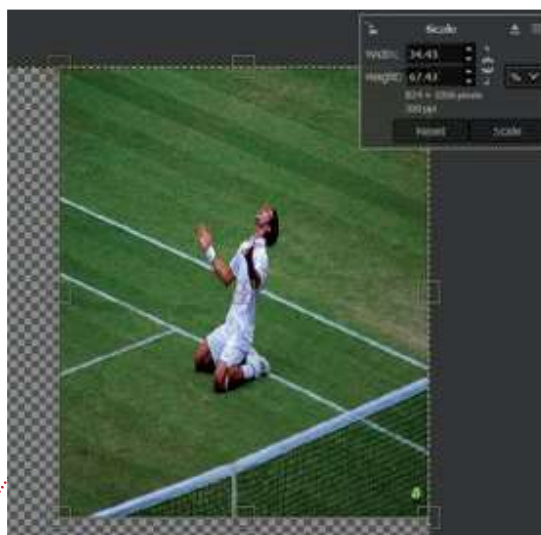


ИНФО ПЛУС

Смањење слике обично не утиче на њен квалитет. С друге стране, промена величине слике на већу од оригиналне, може утицати на њен квалитет.



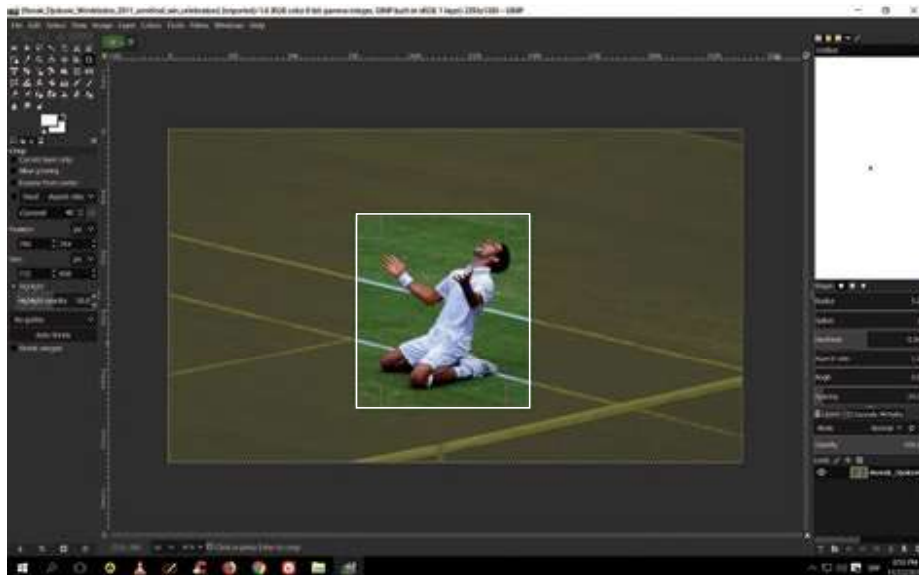
Инч (енгл. *Inch*) је англоамеричка јединица за дужину. Један инч једнак је 2,54 центиметра ($1'' = 2,54 \text{ cm}$). У инчима се и код нас изражава величина екрана дигиталних уређаја. Када чујеш да је паметни телевизор од 32 инча, то значи да је толика величина дијагонале његовог екрана.



о Слика 1.3.21. Промена величине слике

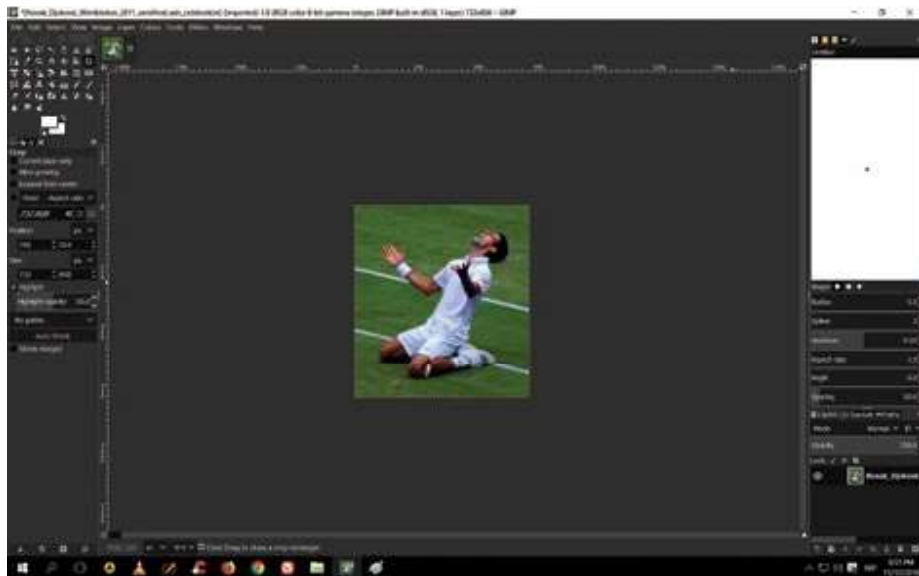
Одсецање дела слике

У *GIMP*-у се одсецање обавља помоћу алатке **Crop Tool**. Одабери алатку, па означи део слике који ће бити одсечен. Означени део слике ће посветлети, а остатак ће потамнети (слика 1.3.22).



Слика 1.3.22. Одсецање дела слике – алатка Crop Tool

Притиском на **Enter** остаће ти само означени део (слика 1.3.23). Можеш га сачувати као посебан фајл.



Слика 1.3.23. Одсечени део слике



ИНФО ПЛУС

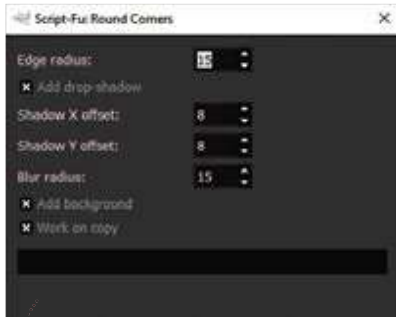
Када означиш део слике који желиш да одсечеш, можеш кориговати селекцију превлачењем ивица ка унутрашњости селекције или ван ње. На тај начин можеш смањити или проширити селектовани део за одсецање.

Употреба филтера

Појавом друштвених мрежа знатно је порасла популарност фотографисања. Већина људи воли да побољша квалитет слике пре постављања на друштвену мрежу. За то се између осталог користе филтери. Они служе за додавање специјалних ефеката на слику. *GIMP* има преко 100 различитих врста филтера. Веома се једноставно користе. Отвориш слику и преко менија **Filters** (филтери) примениш филтер по жељи.

На пример, можеш заоблити слику. За ово се користи филтер **Round Corners** (заобли ћошкове). Отворићемо слику из претходног примера, из менија **Filters**, одабрати **Decor** (украси) па **Round Corners**. У прозору који се појављује (слика 1.3.24) можеш подесити додатне опције које зависе од употребљеног филтера. Када их подесиш, кликни на ОК.

На левој страни следећег примера налази се оригинална слика, док је на десној слика с примењеним филтером.



о Слика 1.3.24. Додавање филтера Round Corners



о Слика 1.3.25. Ефекат филтера Round Corners

На слици 1.3.26. је приказан ефекат филтера **Old Photo** (стара фотографија). Он на слици прави ефекат старе црно-беле фотографије.

Филтер **Slide** (слајд) (слика 1.3.27) користи се када на слику желиш да додаш оквире какви су некад постојали на филмовима за старе фото-апарате.

ВЕЖБАЈ



Преузми слику спортисте/ спортисткиње с интернета. Отвори је у програму *GIMP* и испробај ефекте разних филтера.



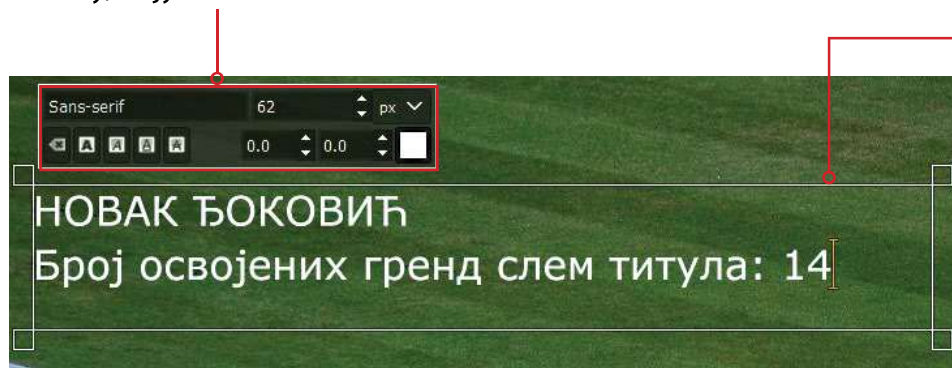
о Слика 1.3.26. Ефекат филтера Old Photo



о Слика 1.3.27. Ефекат филтера Slide

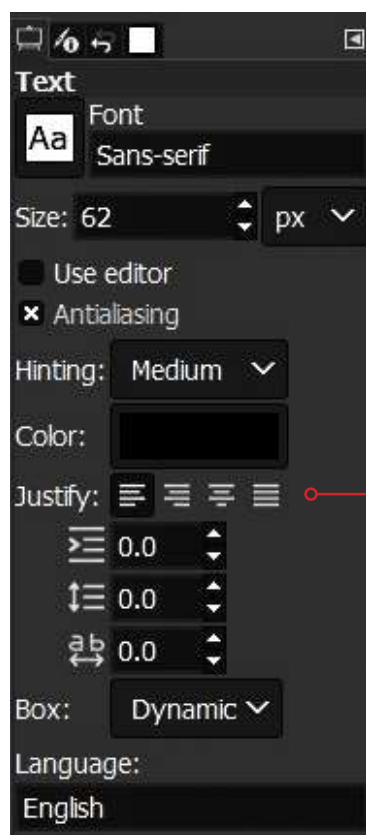
Уметање текста

A За додавање текста се користи алатка **Text Tool**. Кад је одабереш кликни левим тастером миша на место на слици где желиш да уметнеш текст. Потом откуцај текст. Изнад оквира за текст појавиће се помоћни мени у ком можеш да прилагодиш фонт, величину, боју и стил слова.



Приликом куцања, текст је означен оквиром. Овај оквир остаје и када завршиш куцање текста. Димензије оквира с текстом можеш смањивати и повећавати превлачењем ивица и углова ка унутра односно ка споља.

Слика 1.3.28. Уметање текста на слику

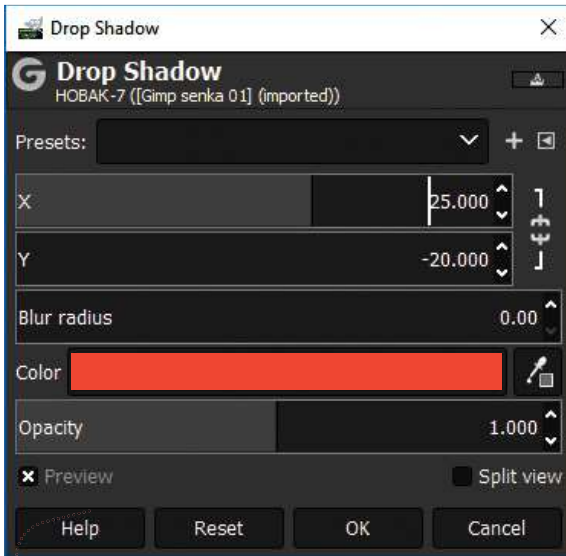


Слика 1.3.29. Додатне опције за Text Tool

Подешавања текста можеш прилагођавати и у додатним опцијама. Помоћу опције **Justify** (обострано поравнање) поравнава се текст, а у опцијама испод подешава се растојање између редова и размак између слова.

+ Када је активан, оквир с текстом можеш померати превлачењем на жељено место уз притиснут тастер **Alt**. У случају да није активан, у ову сврху можеш користити **Move Tool** (премештање).

Текст такође можеш улепшати коришћењем филтера. Ми ћемо за пример употребити филтер **Drop Shadow** (додај сенку) који се налази у менију **Filters** у опцији **Light and Shadow** (светлост и сенка). У прозору **Drop Shadow** можеш променити позицију сенке померајући је по X и Y осе. Затим, можеш мењати боју сенке (**Color**) и њену прозирност (**Opacity**). Пошто одабереш вредности кликни на **OK**.



o Слика 1.3.30. Додавање ефекта тексту



o Слика 1.3.31. Текст с филтером

Ефекти филтера **Drop Shadow** виде се на слици 1.3.31. Сенка је наранџаста, али јој је прозирност подешена на 1 (од максималних 2 што даје потпуно непровидну сенку). Сенка је у односу на текст померена по X осе за 25 пиксела. По Y осе је померена за -20 пиксела. Негативна вредност за позицију сенке по Y осе подразумева да је сенка изнад текста. Када је вредност за X осу негативна, сенка иде лево од текста. Као ликови у координатној мрежи сцене у програму *Scratch*.

Употреба слојева

За обраду било какве слике, важно је схватити како се користе **слојеви** (енгл. *Layers*). Они служе за спајање две слике или више њих у једну. Ово је највећа разлика између *GIMP*-а (и сличних програма) и *Paint*-а.

Као пример, погледај следећу слику. Она је састављена од четири засебне слике при чему је свака у свом слоју. То су слике на којима су Новак Ђоковић, Роџер Федерер, знак Вимблдона и те-

ниски терен. Када се ови слојеви споје, добија се комплетна слика. Слојеви могу бити провидни, тако да се види цела слика или само неки део слике испод или изнад тог слоја. Текст који додаш на слику се такође додаје у посебном слоју.



о Слика 1.3.32. Пример слике са више слојева

Слике се најлакше уклапају када имају провидну (транспарентну) позадину. То важи за све слике осим за ону која треба да се налази у првом слоју. У нашем случају то је терен Вимблдона. Да би пронашао/-ла слике с провидном позадином, приликом претраге интернета поред кључних речи *Djokovic*, *Federer* итд. откуцај и **png transparent**. Код оваквих слика нема позадине коју мораш да уклањаш како би слику уклапао/-ла са осталим слојевима.

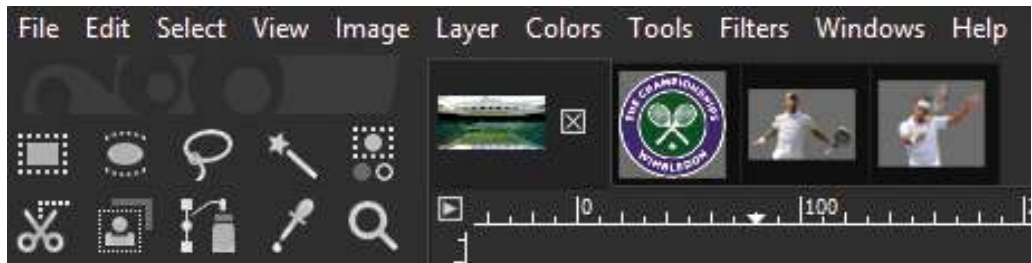


Слике у **GIF** формату такође могу имати провидну позадину.



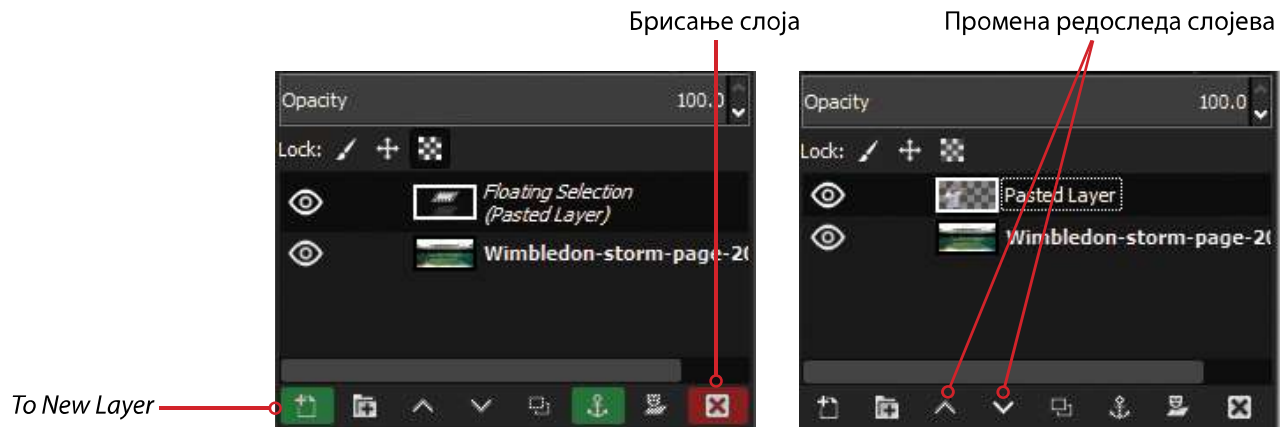
о Слика 1.3.33. Слике с провидном (транспарентном) позадином

Када пронађеш све слике које ћеш уклапати, потребно је да их отвориш у *GIMP*-у. Прво отвори ону која ће бити подлога (први слој). У нашем случају то је терен Вимблдона. Затим отвори све преостале слике које ћеш убацивати на подлогу. Пређи на неку од слика које уклапаш и кликни на њену картицу у врху прозора. Тиме ћеш је уједно селектовати. Сада је копирај, а затим пређи на подлогу и налепи копирану слику.



Слика 1.3.34. Отворене слике у програму **GIMP** спремне за уклапање

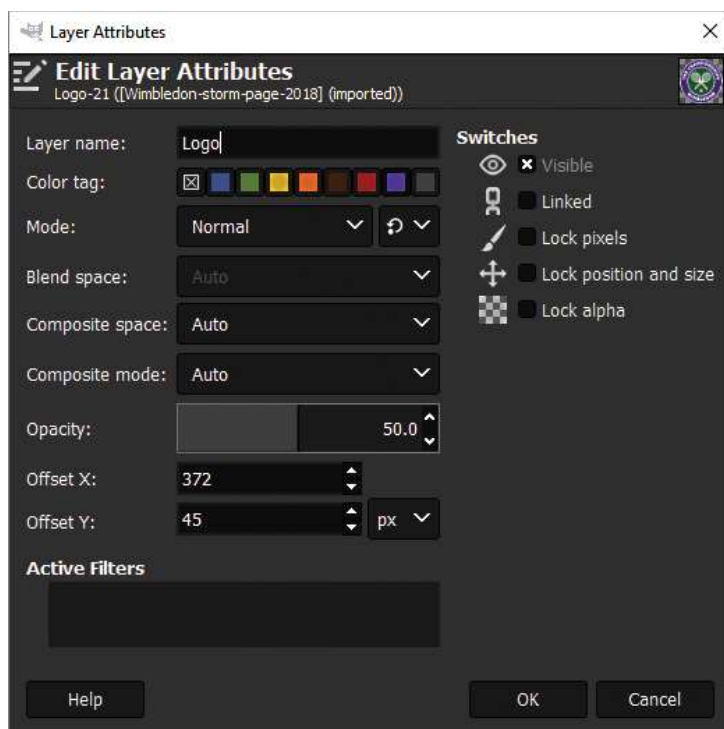
Када на овај начин додаш слику на подлогу, она ће бити налепљена у плутајућем слоју (енгл. *Floating Selection*). Потребно је да је убациш у нови слој. Најлакши начин је да кликнеш на команду **To New Layer** (у нови слој) која се налази у дну оквира са слојевима (слика 1.3.35. лево).



Слика 1.3.35. Додавање слика у слојеве

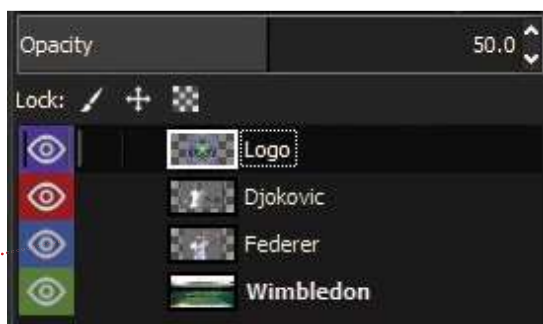
Тиме ћеш направити нови слој који добија назив **Pasted Layer** (налепљен слој). Двокликом на назив слоја, можеш му доделити нови назив. Ако хоћеш да прилагодиш још нека подешавања слоја, кликни на његов назив десним тастером миша и одабери команду **Edit Layer Attributes** (измени својства слоја).

Добићеш прозор **Layer Attributes** (својства слоја) у коме можеш да промениш назив слоја (**Layer name**), одабереш боју за ознаку слоја (**Color tag**) и по потреби подесиш још неке параметре.



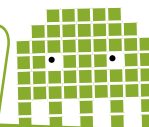
о Слика 1.3.36. Слојеви и њихова подешавања

На исти начин додај и преостале слике и прилагоди називе и евентуално боје слојева. Када радиш с више слојева, њихов редослед је веома важан. Њега најлакше мењаш превлачењем слоја на жељено место. Редослед можеш кориговати и стрелицама у дну дела прозора са слојевима (слика 1.3.35. десно). Слојеви за слику 1.3.32. су поређани тако да се у првом слоју налази **Wimbledon**, у другом **Federer**, у трећем **Djokovic**, а у четвртном **Logo**.



о Слика 1.3.37. Промењени називи и обојене ознаке слојева

ВЕЖБАЈ



1. Са интернета преузми слике фудбалског стадиона, двојице омиљених фудбалера и пехара Лиге шампиона. За сваку слику направи посебан слој. Њиховом комбинацијом начини постер за најаву финала Лиге шампиона. На постеру напиши који тимови учествују у финалној утакмици и напиши датум утакмице
2. Направи разгледницу димензија 10.5 x 15 cm чија ће тема бити Србија. Мотиви слика које можеш преузети са интернета, а потом уклопити у разгледницу су: главни град Србије, застава Србије, грб Србије, најлепша места/ насеља Србије итд.

Ако кликнеш на симбол ока с леве стране, искључићеш слој са слике. Око ће тада нестати. Вратићеш га поновним кликом. Када користиш слојеве, измене слике на једном слоју ни на који начин не утичу на слике које се налазе у другим слојевима. На пример, ако треба да смањеш лого Вимблдона, ово ћеш урадити само у том слоју.

ИНФО ПЛУС

На претходним сликама си можда приметио/-ла опцију **Opacity** (прозирност). Она се често користи у раду са сликама. Служи за прилагођавање провидности слоја. Вредност 100 значи да је слој непровидан. Како се иде ка вредности 0, слој постаје све провиднији. Providnost логотипа Вимблдона је подешена на 50.

Селектовање објекта на сликама с непровидном позадином

Ако пак не проналазиш слику с провидном позадином, ни то није проблем. Показаћемо ти како да селекујеш објекат и да га издвојиш од позадине. За ово ти је потребна алатка под називом Паметне Маказе (енгл. *Intelligent Scissors*).

Ова алатка ти омогућава да селекујеш објекат на слици тако што мишем означаваш тачке по ивици објекта. Што је већи број тачака, коначна селекција биће прецизнија. Потребно је обићи цео објекат и на крају кликнути на почетну тачку. Након тога притисни **Enter**. Да би лакше на овај начин селектовао/-ла објекте, пожељно је да зумираш слику.

Пример селектовања алатком **Intelligent Scissors** је приказан на слици 1.3.38.

Претходно селектовани део слике (Новак Ђоковић који клечи) убацили смо преко слике Вимблдона. На тај начин смо комбинацијом две слике поново добили једну.

САЗНАЈ ВИШЕ

Применом било које алатке за селектовање, врши се селекција само с активног слоја слике.



○ Слика 1.3.38.

Селектовање дела слике помоћу алатке *Intelligent Scissors*

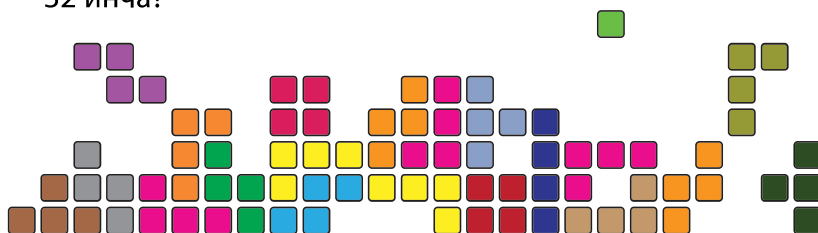


○ Слика 1.3.39. Још један пример слике с више слојева



ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

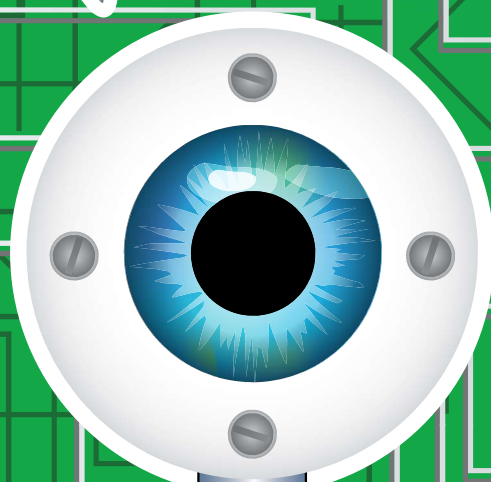
1. На које се све начине бира алатка у програму *GIMP*?
2. Како се у програму *GIMP* цртају облици?
3. Како се одсецају делови слике? Које су сличности а које разлике међу начинима одсецања у програму *Paint* и у програму *Word*?
4. Објасни како се и зашто употребљавају слојеви при обради слика.
5. Чему служи опција **Opacity**?
6. Колика је, у сантиметрима, дијагонала телевизора од 32 инча?



УКРАТКО

За цртање у програму *GIMP* се користе алатке Paint Tools (оловка, четкица, гумица, кофица, и др.). Не постоје понуђени облици као у *Paint*-у или *Word*-у. Облици се цртају помоћу алатки за селектовање (Rectangle Select, Eclipse Select, Free Select и др.). Облицима можеш додавати и одузимати делове, бојити им унутрашњост и контуру. Помоћу Save As опције могуће је сачувати слику само у XCF формату. За друге формате се користи опција Export. Када отвориш слику (File => Open) можеш је обрађивати – мењати јој величину (Scale), одсецати делове (Crop), употребљавати филтере, уметати текст итд. Слојеви (Layers) служе за спајање две слике или више њих у једну. Измене слике на једном слоју ни на који начин не утичу на слике које се налазе у другим слојевима.

2. ДИГИТАЛНА ПИСМЕНОСТ



2.1. ДИГИТАЛНИ УРЕЂАЈИ У МРЕЖНОМ ОКРУЖЕЊУ

КЉУЧНЕ РЕЧИ

мрежа, интернет, глобална мрежа, локалне мреже, мреже широког
распона, мрежни уређаји, хакер, антивирус, заштитни зид

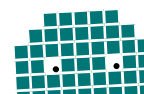


Цивилизација не би достигла данашњи степен развоја да људи с временом нису научили да размењују информације и сарађују. Тако је и с рачунарима. Све погодности које пружа примена информационих технологија не би вределе када рачунари не би могли да буду **повезани** и да **комуницирају**. Група међусобно повезаних рачунара који деле ресурсе и информације назива се **рачунарска мрежа**.



Слика 2.1.1. Интернет је највећа рачунарска мрежа

Највећа рачунарска мрежа на свету је интернет, јер повезује различите територијално удаљене мреже на планети. Зато се каже да је интернет светска, односно **глобална рачунарска мрежа**. Често га називају и „мрежом свих мрежа“. Интернет омогућава комуникацију и размену информација између рачунара, паметних телефона и других дигиталних уређаја широм света. Прошле године си научио/-ла да ти је за везу с интернетом, поред дигиталног уређаја, потребан интернет провајдер (фирма која обезбеђује везу с интернетом). Провајдери могу пружати ову услугу на различите начине (преко ADSL-а, оптичког интернета, кабловског интернета и бежичног интернета – *Wireless*).



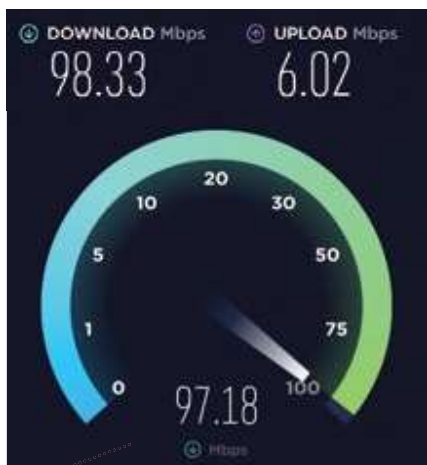
РЕЧНИК

Људи користе природне ресурсе (земљиште, воду, фосилна горива). И рачунари имају своје ресурсе. За рачунаре су ресурси све оно што им је потребно за рад (меморија, програми, штампач, скенер итд.).



ЗАНИМЉИВОСТ

Прва рачунарска мрежа на свету и претеча интернета је био ARPANET. Развил га је америчко министарство одбране крајем 60-их година. ARPANET је повезивао све важне научне и војне институције.



o **Слика 2.1.2.** Провера брзине конекције

ВЕЖБАЈ

На сајту <https://www.speedtest.net/> провери брзину своје везе с интернетом. Израчунај колико би ти времена требало да преузмеш садржај од 2 гигабајта (GB).

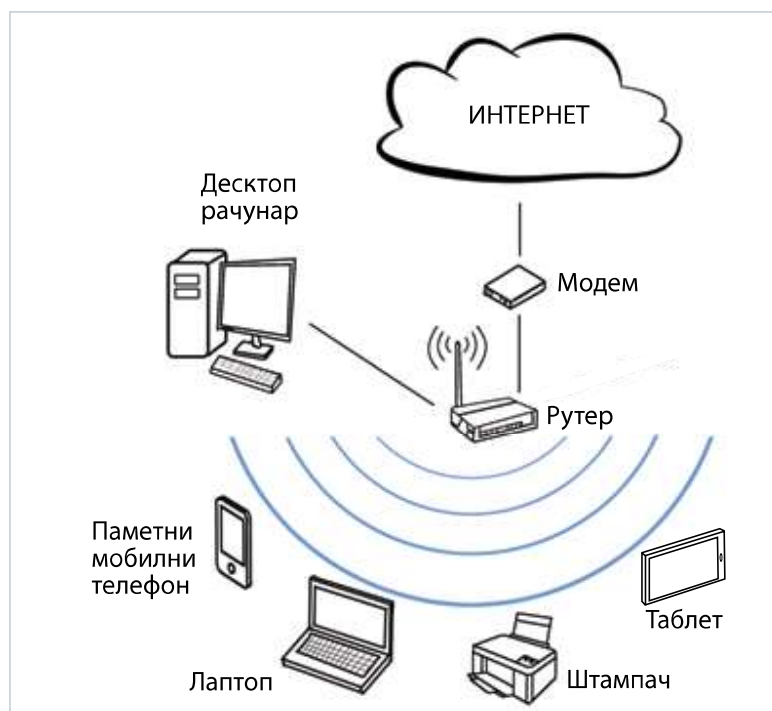


САЗНАЈ ВИШЕ

Тренутно најбржу конекцију омогућава оптички интернет. Брзине конекције могу достићи чак 1000 Mb/s.

За кориснике интернета најбитнија је **брзина конекције** (везе с интернетом). Она се изражава у **мегабитима у секунди (Mb/s)**. Притом се под брзином конекције подразумева брзина преузимања података са сервера на наш рачунар (енгл. *Download*). Брзина слања (отпремања/постављања) података с нашег рачунара на неки сервер (енгл. *Upload*) увек је знатно мања од брзине преузимања. Ако си у петом разреду научио/-ла разлику између бита и бајта, знаћеш да разликујеш мегабит (Mb) и мегабајт (MB). Када вредност брзине преузимања (**Download**) или слања података (**Upload**) поделиш са 8, добићеш брзину изражену у мегабајтима у секунди (MB/s).

Поред интернета, постоје мање рачунарске мреже. Оне повезују мањи број рачунара на мањем простору. Примера ради, такве су мреже у једној играоници, једној канцеларији или више суседних канцеларија, једној учионици или згради. Овакве мреже се називају **локалне рачунарске мреже** (енгл. *Local Area Network*, скр. *LAN*). Користе се да повежу уређаје удаљене највише до 1000m. Локална мрежа, поред повезивања, омогућава и дељење свих ресурса у мрежи. Уколико у мрежи постоји један штампач, на пример, сви рачунари могу да га користе. На следећој слици је приказана мала локална мрежа. С обзиром на то да се данас користи велики број дигиталних уређаја, свака кућа се може сматрати једном локалном мрежом.



o **Слика 2.1.3.** Локална мрежа (LAN)

Да би повезао рачунар у мрежу потребно је да имаш делове хардвера које до сада нисмо помињали. Они се убрајају у мрежну опрему. На претходној слици си видео/-ла два таква уређаја (модем и рутер).

Модем (енгл. *Modem*) јесте уређај који повезује локалну мрежу с интернетом. Он омогућава рачунару да шаље и прима податке преко телефонске линије, каблова кабловске телевизије итд. Као што видиш на претходној слици, за LAN мреже је довољан један модем преко кога је цела мрежа повезана на интернет.

Рутер (енгл. *Router*) служи да повеже уређаје у мрежи или да повеже више локалних мрежа. Везу с интернетом коју омогућава модем, рутер дели свим уређајима у мрежи. Постоје типови модема који имају уграђен бежични рутер. Тада ти посебан рутер није потребан.

Мрежна картица је основни и најважнији елемент мрежне опреме. Она омогућава повезивање рачунара у мрежу. Често се назива и **мрежни адаптер** (енгл. *Network adapter*). Мрежна картица се повезује на одговарајући слот на матичној плочи. Матичне плоче данашњих паметних телефона, таблета и лаптоп рачунара углавном имају већ уграђене мрежне картице за бежично повезивање на интернет. То само показује колико се у данашње време подразумева коришћење интернета.



Паметни телефони имају опцију **Hotspot & tethering** (хотспот и привезивање) помоћу које се могу претворити у такозване **бежичне приступне тачке**. Тада се понашају као виртуелни рутери па други уређаји могу да користе њихову везу с интернетом. Преко паметног телефона на којем је активирана ова опција, може се повезати на интернет до 10 других уређаја.



ИНФО ПЛУС

Десктоп рачунари најчешће немају мрежне картице за бежично повезивање на мрежу. Они се на рутер повезују мрежним каблом (енгл. *Ethernet cable*). Каже се још и LAN кабл. Ако хоћеш да избегнеш каблове, можеш купити бежичну мрежну картицу која се монтира у кућиште (слика 2.1.5). Уколико ти монтирање бежичне мрежне картице делује компликовано, постоје екстерне бежичне мрежне картице. Оне изгледају као USB флеш меморије с тим што имају антену. Прикључују се на USB порт.



Слика 2.1.4. Мрежни (LAN) кабл



Слика 2.1.5. Бежична мрежна картица



Слика 2.1.6. Екстерна бежична мрежна картица

Спајањем локалних мрежа у оквиру већег подручја (града, региона, државе итд.) настају **мреже широког распона** (енгл. *Wide Area Network*, скр. *WAN*). Најпознатија и највећа мрежа широког распона на свету је управо интернет. Он је сачињен од мноштва националних *WAN* мрежа. Зато се и назива мрежа свих мрежа.



Слика 2.1.7. Мрежа широког распона (*WAN*) спаја више *LAN* мрежа

Правила употребе дигиталних уређаја у мрежном окружењу



Слика 2.1.8. Заштитни зид чува безбедну од небезбедне зоне

Пошто сада знаш како функционише интернет, лакше ћеш схватити чему заправо служи заштитни зид (енгл. *Firewall*). Он има улогу мрежног саобраћајног полицајца. Главни ток мрежног саобраћаја одвија се између интернета и локалне мреже. Локална мрежа се сматра безбедном зоном (енгл. *Trusted*), а интернет небезбедном (енгл. *Untrusted*). Заштитни зид одређује шта може, а шта не може да прође с интернета у локалну мрежу.

Заштитни зид и антивирус (енгл. *Antivirus*) у оперативном систему *Windows 10* налазе се у оквиру безбедносног центра (енгл. *Windows Defender Security Center*). Подешени су да се аутоматски ажурирају и да увек буду укључени. Ово је једно од најважнијих правила безбедне употребе дигиталних уређаја у мрежном окружењу. Без обзира на заштиту коју дигитални уређаји имају захваљујући антивирусу и заштитном зиду, на интернету увек мораш бити обазрив/-а. Никома немој одавати своје личне податке и води рачуна о нетикецији. У поглављу 2.4. научићеш још нека правила. Међутим, било би добро да се подсетиш и оних која смо ти предочили прошле године.

ИНФО ПЛУС

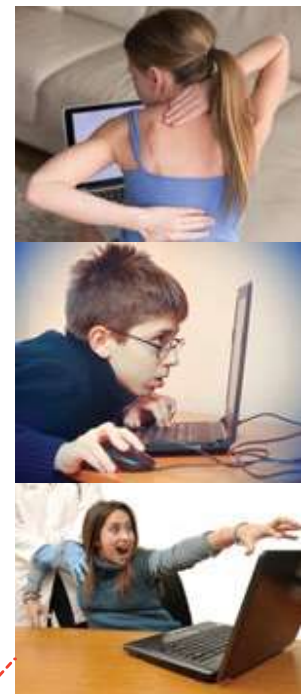
У дигиталном, као и у реалном свету, постоје људи који се не придржавају правила понашања. Склони су да праве штету другима и да угрожавају безбедност. Овакве особине се најчешће приписују хакерима. **Хакер** је особа која модификује (мења) програме, недозвољено приступа туђим рачунарима и користи туђе информације. Ипак, хакери често имају добре намере. Постоји група хакера који се зову бели хакери (енгл. *White hats*). Они раде на томе да онлајн свет постане боље и сигурније место.



Слика 2.1.9. Хакер

Подразумева се да се одговорно и безбедно понашаш према свом уређају и према другим корисницима интернета. Но, треба да се одговорно понашаш и према себи. Немој дозволити да ти паметан телефон и рачунар одузимају превише слободног времена. Забавни садржаји на интернету су направљени да буду веома привлачни али те могу одвојити од мноштва других, лепих и корисних ствари. Много би корисније било да слободно време после школе и учења проведеш на друге начине. На пример, можеш са својим другарима отићи у шетњу по природи, на пливање, вожњу бициклом, можеш играти баскет, фудбал, одбојку итд. На крају крајева, било би корисније и да одеш с друштвом у биоскоп да погледате добар филм.

Многи људи се у данашње време не раздвајају од својих паметних телефона или таблета. Уколико се томе дода и неправилно држање, могу се јавити главобоље, поремећаји вида, болови у врату, леђима или деформитети кичме. Врхунац су зависност од интернета или видео-игрица, која може проузроковати психичке проблеме. Ето додатних разлога да ограничиш коришћење дигиталних уређаја. Поред тога, док их употребљаваш, прави паузе на сваких 30 до 45 минута. Током паузе уради вежбе истезања.



Слика 2.1.10. Последице неправилног и прекомерног коришћења дигиталних уређаја

ИНФО ПЛУС

Један од уређаја за који би се могло рећи да је уз њега оправдано проводити више времена, јесте читач електронских књига (енгл. *E-book reader*). Најпопуларнији такав уређај је *Kindle*. Најновија верзија има капацитет меморије од 32 GB што значи да у њега може стати више од 20000 књига.

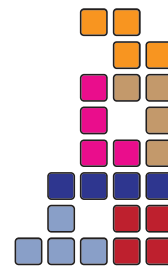


Слика 2.1.11. *Kindle* – читач књига у дигиталном формату (*E-book reader*)

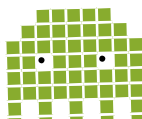


ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Чему служи наредба **print**?
2. Шта је специфично код наредбе **print** и рада с бројевима?
3. Шта је резултат реалног, а шта целобројног дељења?



ВЕЖБАЈ



1. У конзоли израчунај вредности следећих израза:
 - а) $(10 - (-6) / (-5 + 2) + (-15 - 5 / 5))$
 - б) $(((-3) * (-4) - 5 * (-6)) * (-6 + (-2) * (-3)))$
 - в) $(((-12) * (-6) + 4 * (-32)) * (-8 - 2 * (-2)))$
 - г) $((999 - 6 * (125)) * (-3 * 16 + (-7) * (-7)))$
2. У конзоли провери која је разлика у резултату израза $7*7//2-7*7\%2$, $7*7//2-(7*7\%2)$ и $(7*7//2)-(7*7\%2)$
3. Наредбом **print** на екрану испиши резултат разлике петоструке вредности броја **-13** и троструке вредности броја **15**. Све то одузми од броја **-100**. Програм сачувај под називом *Tezak izraz*.
4. Употреби наредбу **print** и на екрану испиши резултат производа бројева **-100** и **3** умањен за производ бројева **-100** и **-1**. Тој разлици додај збир бројева **61** и **-71**. Добијену вредност израза одузми од броја **-100**. Програм сачувај под називом *Jos tezi izraz*.



УКРАТКО

Основне аритметичке операције (сабирање, одузимање, множење и дељење) можеш испробати у конзоли на било ком математичком изразу. Довољно је да га откуцаш и притиснеш Enter. Поред основних аритметичких операција, можеш користити и заграде, негативне бројеве итд. Наредбом за исписивање **print** такође можеш одредити неки израз и добити његов резултат. Уколико израз ставиш под наводнике, коначно решење биће запис тог израза а не његов резултат. Програмски код који садржи дељење с нулом не може да се изврши. Коришћењем оператора */*, *//* и *%*, у *Python*-у можеш добити количнике реалног и целобројног дељења, те остатак при целобројном дељењу.

3.4. ПРОМЕНЉИВЕ

КЉУЧНЕ РЕЧИ

променљиве с додељеним вредностима, променљиве с уносом вредности, типови података, наредба *input*

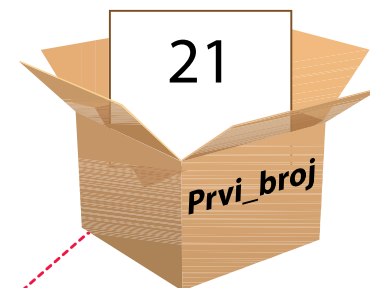


Променљиве су један од основних појмова у програмирању. Оне имају важну улогу у свим програмским језицима. Користе се на исти начин у блоковском и у текстуалном програмирању. Променљива представља именовано место у меморији рачунара у ком се чувају подаци. То могу бити нумерички (бројчани), текстуални и други типови података. Наравно, не могу се сви чувати одједном. Променљива у датом моменту може чувати само један податак.

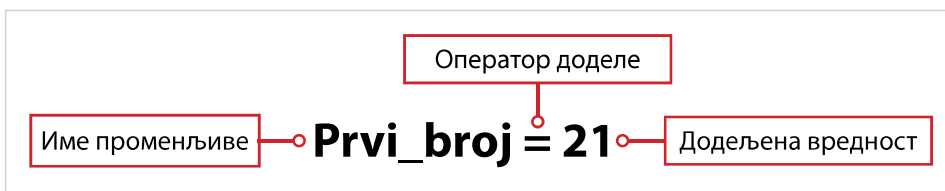
Када именујеш променљиву у *Python*-у, мораш поштовати следећа правила:

1. у имену не сме постојати размак. Ако се име састоји од више речи, уместо размака користи доњу црту;
2. име може садржати слова и бројеве. Притом, име не може почињати бројем нити бити само број.

Вредности се променљивама дају помоћу израза доделе. Он садржи: име променљиве, оператор доделе (знак једнакости) и вредност променљиве. Наредни израз доделе програму говори Променљива *Prvi_broj* сада има вредност 21.



Слика 3.4.1. Променљиву можеш замислити као кутију у коју стављаш податак



Слика 3.4.2. Израз доделе вредности променљивој

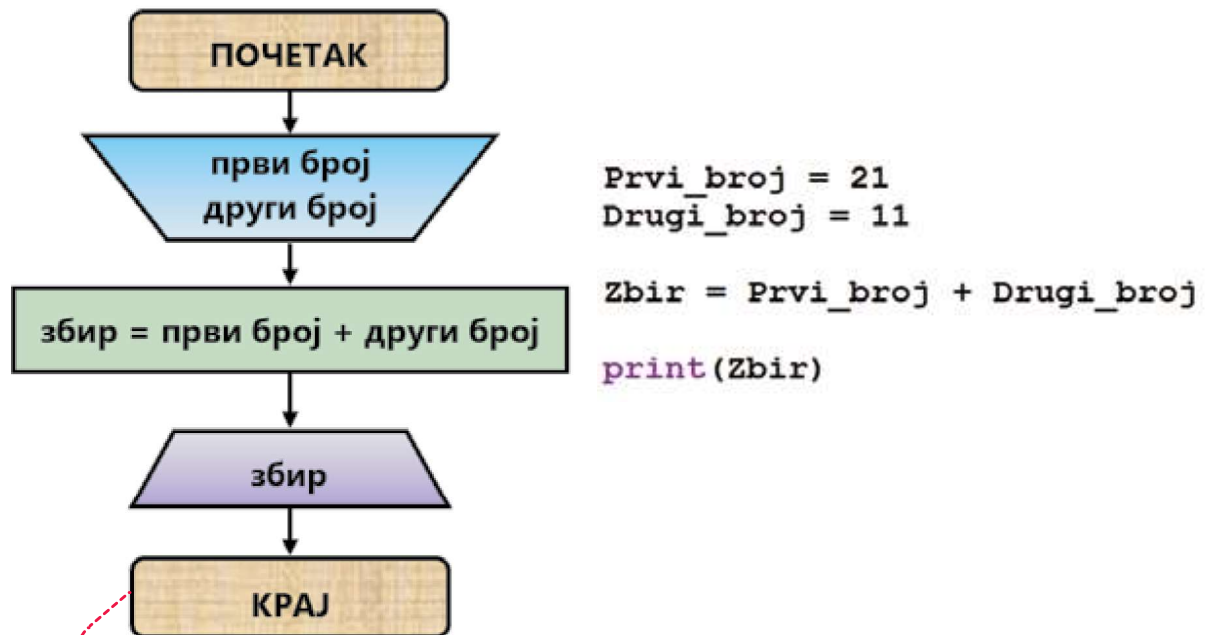
ИНФО ПЛУС



Променљивама се поред бројчаних, могу доделити и текстуалне вредности. У том случају вредност променљиве треба ставити под наводнике.

Променљиве омогућавају чување вредности улазних података, као и приказивање излазних података. Излазни подаци су најчешће резултат неког израза. У алгоритму на следећој слици,

видећеш три променљиве (*Prvi_broj*, *Drugi_broj* и *Zbir*). Такође, видиш да се оне користе у улазном и излазном блоку и у блоку обраде.



Слика 3.4.3. Променљиве у блоковима алгоритамске шеме и линијама програмског кода

У улазном блоку алгоритма су дефинисане две променљиве *Prvi_broj* и *Drugi_broj*. У блоку обраде је дефинисана променљива *Zbir*. У њој се рачуна збир вредности променљивих *Prvi_broj* и *Drugi_broj*. На крају се у излазном блоку приказује резултат израза односно вредност променљиве *Zbir*.

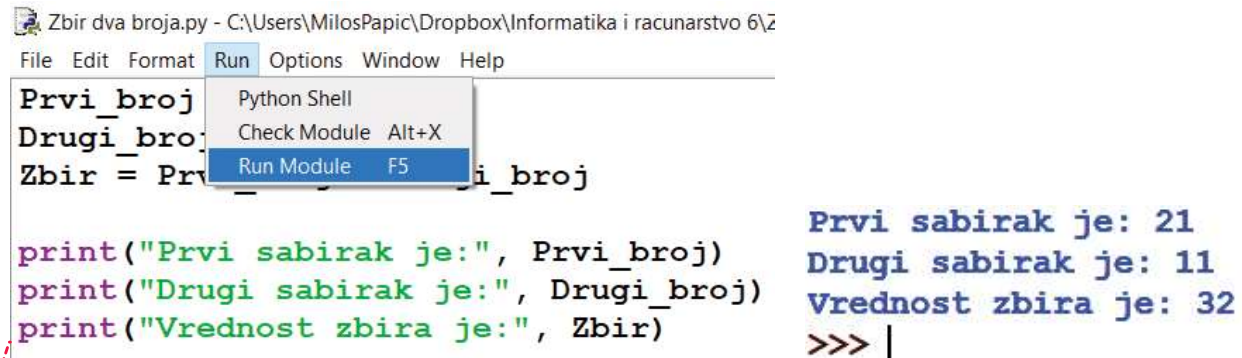
У програмском коду видећеш да су у прве две наредбе дефинисане променљиве *Prvi_broj* и *Drugi_broj* и да су им додељене вредности 21 и 11. У трећој наредби је дефинисана променљива *Zbir* чија је вредност збир вредности променљивих из претходних наредби (**Zbir = Prvi_broj + Drugi_broj**). Наредбом **print** се приказује резултат програма. Када притиснеш **F5** и покренеш овакав програм добићеш резултат као на слици 3.4.4.

Резултат претходног програма јесте тачан, али било би много јасније кад би уз њега стајала нека порука. Она може да казује на шта се дати број односи. На пример, *Vrednost zbira je 32*. Ову поруку можеш додати у оквиру наредбе **print** пре позива променљиве *Zbir*. Али зар не би било још боље када би програм приказао вредности првог и другог броја? Пре наредбе за приказ вредности збира можеш додати још две наредбе **print** којима ћеш уз поруке приказати вредности првог и другог броја. То би изгледало као на слици 3.4.5. лево.

```
32
>>> |
```

Слика 3.4.4. Резултат програма за сабирање два броја

Сада можеш сачувати фајл (**File** => **Save** или **Ctrl + S**). После тога покренити програм (**Run** => **Run Module** или **F5**). Добићеш резултат који је приказан на слици 3.4.5. десно.



```

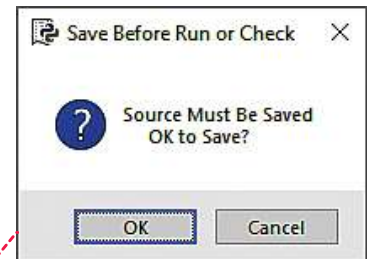
Zbir dva broja.py - C:\Users\MilosPavic\Dropbox\Informatika i racunarstvo 6\z
File Edit Format Run Options Window Help
Prvi_broj Python Shell
Drugi_broj Check Module Alt+X
Zbir = Prvi_broj + Drugi_broj Run Module F5

print("Prvi sabirak je:", Prvi_broj)
print("Drugi sabirak je:", Drugi_broj)
print("Vrednost zbira je:", Zbir)

Prvi sabirak je: 21
Drugi sabirak je: 11
Vrednost zbira je: 32
>>> |
  
```

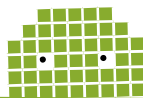
Слика 3.4.5. Уз додатне линије кода резултат програма је јаснији

Сложићеш се да је овакав резултат програма много јаснији и практичнији од резултата на слици 3.4.4. Труд се увек исплати. Сада се можеш вратити у едитор и некој променљивој изменити вредност. Када покренеш програм, *Python* ће тражити да прво сачуваш измене (слика 3.4.6), а затим ћеш у конзоли добити другачији резултат збира.



Слика 3.4.6. Свака промена у програму мора бити сачувана пре но што га покренеш

ВЕЖБАЈ



Направи програм који ће рачунати просечно трајање трке од три трке на 100 метара. Дефиниши три променљиве: *trka1*, *trka2* и *trka3* којима ћеш доделити вредности трајања трка у секундама (нпр. 9.58). Четврта променљива под називом *Prosek* треба да сабере вредности прве три променљиве и да их подели са 3. На крају прикажи вредност променљиве *Prosek* и пратећу поруку.



У уводу ове лекције смо рекли да променљиве могу чувати податке различитог типа. Основни **типови података** (енгл. *Data types*) јесу бројчани и текстуални. Текстуални тип податка се у програмирању назива **стринг**. Стринговима је посвећена једна од наредних лекција. У програмском језику *Python* ћеш углавном користити две врсте бројева. То су цели бројеви, који се означавају са **int** скраћено од *Integer* (нпр. број 2), и реални бројеви који се означавају са **float** (нпр. број 2.8).

```

>>> 2+2.8
4.8
>>> 2 + 2.8
4.8
>>> "2" + "2.8"
'22.8'
>>> int(2) + int(2.8)
4
>>> int(2) + float(2.8)
4.8
  
```

Слика 3.4.7. Различити типови података дају различите вредности

Променљиве с учитавањем вредности

Ако си успео/-ла да направиш претходни програм, сигурно си га испробао/-ла с неколико различитих вредности. Вероватно сада мислиш: „Било би добро да ме програм пита које вредности хоћу да доделим променљивама. Баш као што је то радио *Scratch*.”

У томе ти може помоћи наредба **input**. Она сама за себе не ради ништа. Увек је мораш користити уз неку променљиву. Када наиђе на наредбу **input**, програм ће стати и чекати да корисник унесе број с тастатуре и притисне **Enter**. Да би корисник знао шта треба да унесе, ти као програмер мораш му то навести уз наредбу **print** и пратећу поруку. Ево како би то изгледало на примеру програма за сабирање целог и реалног броја.

```
print("Unesi ceo broj:")          Unesi ceo broj:
Ceo_broj = input()              10
print("Unesi realan broj:")      Unesi realan broj:
Realan_broj = input()           2.8
Zbir = Ceo_broj + Realan_broj    Njihov zbir je: 102.8
print("Njihov zbir je:", Zbir)  >>> |
```

Слика 3.4.8. Сабирање целог и реалног броја с учитавањем вредности



Слика 3.4.9. Шта није у реду с наредбом **input**!?

Ово је прилично елегантно решење али постоји један проблем. Збир није тачан. Уместо да је сабрао два унета броја и дао резултат 12.8, програм их је спојио. То се десило зато што се *Python* према вредностима унетим преко наредбе **input** понаша као према тексту а не као према бројевима. Одреди типове података за унете вредности променљивих и исправићеш то. Један од начина је приказан на следећој слици. При дефинисању променљиве *Zbir* сваки сабирак се означи одговарајућим типом податка на следећи начин: **Zbir = int (Ceo_broj) + float (Realan_broj)**.

```
print("Unesi ceo broj:")          Unesi ceo broj:
Ceo_broj = input()              10
print("Unesi realan broj:")      Unesi realan broj:
Realan_broj = input()           2.8
Zbir = int(Ceo_broj) + float(Realan_broj) Njihov zbir je: 12.8
print("Njihov zbir je:", Zbir)    >>> |
```

Слика 3.4.10. Исправљен код програма за сабирање целог и реалног броја



Слика 3.4.11. Да би програм радио како треба, променљивама мораш одредити одговарајуће типове података

Овај програм можеш написати на другачији начин. Уз захтев за учитавање вредности (**input**), променљивама *Ceo_broj* и *Realan_broj* одреди тип податка (**int** односно **float**).

```
Ceo_broj = int(input("Unesi ceo broj:"))
Realan_broj = float(input("Unesi realan broj:"))
Zbir = Ceo_broj + Realan_broj
print("Zbir unetih brojeva je:", Zbir)
```

Слика 3.4.12. Другачије решење програма за сабирање целог и реалног броја

ВЕЖБАЈ

1. Повежи кодове с резултатом који дају:

```
a=input("Upiši broj: ")
print(a+a)
```

Upiši broj: **8.2**
16.4

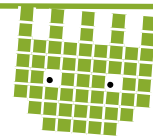
```
a=int(input("Upiši broj: "))
print(a+a)
```

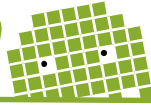
Upiši broj: **28**
2828

```
a=float(input("Upiši broj: "))
print(a+a)
```

Upiši broj: **28**
56

2. Марко има 20 бомбона. Иван има 15 бомбона више од Марка. Бранко има двапут више бомбона од Ивана. Напиши програм који ће израчунати колико укупно бомбона имају три друга. Резултат испиши наредбом **print**.
3. Напиши програм који израчунава који број треба одузети од броја 36521 да се добије број 10562.
4. Маријин ранац је тежак 1,25 kg. Иванин ранац је тежак 1,50 kg. Бранкин је тежак 1,35 kg. Колика је просечна, а колика укупна тежина свих ранчева? Резултат испиши наредбом **print**.
5. Напиши програм за израчунавање израза $(a-b)$. Вредност променљивих a и b уносиш с тастатуре, при чему је a цео број, а b реалан број. Резултат испиши наредбом **print**.
6. Напиши програм за израчунавање збира три цела броја која уносиш наредбом **input**.
7. Напиши програм за израчунавање збира три децимална броја која уносиш наредбом **input**.
8. Правоугаоник има странице дужине 10 сти 5 см. Напиши програм за израчунавање обима и површине правоугаоника. Резултате испиши наредбом **print**.
9. Напиши програм где ћеш наредбом **input** уносити вредности за странице a и b на основу којих ће програм израчунати обим и површину правоугаоника.
10. Твоја сестра на рачуну у банци има 40 000 динара. Данас јој је уплаћена плата у износу од 50 000 динара. Напиши програм који израчунава тренутно стање на рачуну твоје сестре.





Допуни блокове обраде наредних програма одговарајућим изразима да би програми могли да дају понуђене резултате.

Кодови:

- ```
broj_učenika = int(input("Koliko je učenika prisutno u školi?"))
broj_bombona = int(input("Koliko je bombona u kesici?"))

bombona_po_učeniku =
ostalo_bombona =

print("Svaki učenik će dobiti", bombona_po_učeniku, "bombone.")
print("Ostalo je još", ostalo_bombona, "bombona.")
```
- ```
centimetri = int(input("Unesi svoju visinu u centimetrima:"))

m =
cm =

print ("Tvoja visina je:", m, "metar i", cm, "centimetra.")
```
- ```
sekundi = int(input("Unesi vreme u sekundama:"))
h =
m =
s =

print("Uneto vreme je:", h, "sat/i", m, "minuta i", s, "sekundi.")
```
- ```
Dvocifren_broj = int(input("Unesi dvocifren broj:"))
Prva_cifra =
Druga_cifra =
print("Prva cifra je:", Prva_cifra)
print("Druga cifra je:", Druga_cifra)
```

Резултати:

- ```
Koliko je učenika prisutno u školi?28
Koliko je bombona u kesici?100
Svaki učenik će dobiti 3 bombone.
Ostalo je još 16 bombona.
>>>
```
- ```
Unesi svoju visinu u centimetrima:185
Tvoja visina je: 1 metar i 85 centimetra.
>>>
```
- ```
Unesi vreme u sekundama:6326
Uneto vreme je: 1 sat/i 45 minuta i 26 sekundi.
>>>
```
- ```
Unesi dvocifren broj:28
Prva cifra je: 2
Druga cifra je: 8
>>>
```

Промена вредности променљивих

У програмима где се вредности променљивих уносе преко наредбе **input**, сваки пут када се програм поново покрене мењају се вредности променљивих. Када је променљивој додељена нова вредност, стара вредност је заборављена.

Погледај како то изгледа на следећем примеру.

```
>>> Prvi_broj = 15
>>> print("Vrednost promenljive je:", Prvi_broj)
Vrednost promenljive je: 15
>>> Prvi_broj = 10
>>> print("Vrednost promenljive je:", Prvi_broj)
Vrednost promenljive je: 10
```

Слика 3.4.14. Промена вредности променљивих у програму

Променљивој *Prvi_broj* је на почетку додељена вредност 15. Ова вредност је исписана наредбом **print**. Затим је променљивој *Prvi_broj* додељена вредност 10. Након тога је и она исписана наредбом **print**. После нове доделе вредности променљивој *Prvi_broj* стара вредност више није доступна.

Међутим, пре него што постане недоступна, вредност променљиве може бити неопходна да би се добила нова вредност. На пример, цена сладоледа „капри“ је 70 динара. Када сладолед покупи или појефтине, његова цена добија нову вредност. У оваквим случајевима, нова вредност се не може одредити уколико се не зна стара.

```
>>> Cena = 70
>>> print("Cena sladoleda je:", Cena)
Cena sladoleda je: 70
>>> #sladoled je poskupeo 10 posto.
>>> Cena = Cena + Cena*10/100
>>> print("Cena sladoleda je:", Cena)
Cena sladoleda je: 77.0
>>> #sladoled je pojeftinio 10 posto! :)
>>> Cena = Cena - Cena*10/100
>>> print("Cena sladoleda je:", Cena)
Cena sladoleda je: 69.3
```

Слика 3.4.15. Нова вредност променљиве некада зависи од старе вредности



Слика 3.4.13. Нова вредност променљиве замењује стару



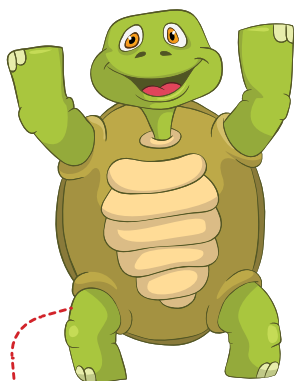
ИНФО ПЛУС

У примеру на слици 3.4.15. постоје линије кода које на почетку имају симбол # (тараба). Иако су написане црвеном бојом, ово нису грешке. То су коментари. Њих програмери остављају на одређеним местима у програму да би објаснили значење наредби или делова кода. Програмски језици коментаре не третирају као део кода већ их приликом извршавања програма занемарују.

3.7. ЦРТАЊЕ У PYTHON-У

КЉУЧНЕ РЕЧИ

библиотека, увоз, *turtle*, корњача, цртање, геометријски облици



о **Слика 3.7.1.**
Кад хоћеш да црташ у програму *Python*, позови корњачу

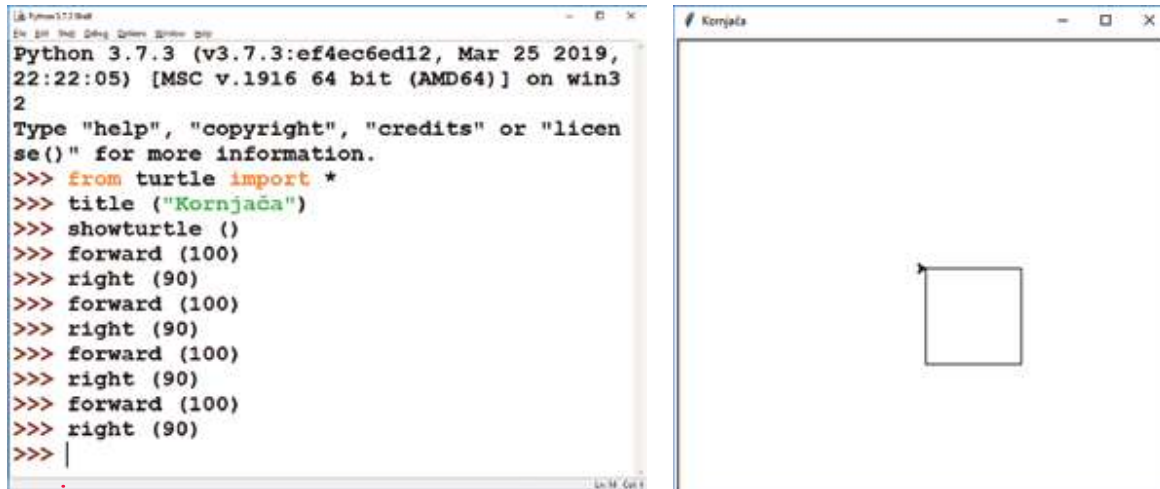
Све наредбе и функције које смо ти до сада показали део су *Python*-ове основне **библиотеке**. Сваки програмски језик, поред основне библиотеке, има и библиотеке наредби и функција које се користе по потреби. Да би употребљавао/-ла одређену библиотеку, мораш је увести (енгл. *Import*) у програм.

Осим за рачунање и рад са стринговима, *Python* можеш користити и за цртање. Потребно је увести библиотеку која садржи скуп наредби за цртање, а зове се ***turtle***.

Сада ћемо ти показати како да увезеш библиотеку *turtle* и да припремиш окружење за свој први цртеж. То се обавља у четири корака:

1. покрени *Python* (конзолу) и води рачуна да прозор не заузима цео екран. Пожељно је да га поставиш на леву половину екрана;
2. наредбом **`from turtle import*`** увези библиотеку за цртање. Увоз било које друге библиотеке врши се на исти начин. Само је потребно уместо *turtle* откуцати њен назив;
3. наредбом **`title ("Kornjača")`** отвараш прозор за цртање под називом Корњача. Ако хоћеш да се прозор другачије зове, напиши нови назив;
4. у конзоли откуцај наредбу **`showturtle()`** или скраћено **`st()`**. На средини прозора за цртање појавиће се мали троугао уперен удесно.

Сада је све спремно за цртање. За пример ћемо нацртати квадрат. Једну линију која ће представљати страницу квадрата црташ наредбом **`forward (100)`** или краће **`fd (100)`**. Број у загради представља дужину линије у пикселима. Затим окрени троугао удесно за 90 степени. Ово постижеш наредбом **`right (90)`**. За окретање улево за 90 степени користи се наредба **`left (90)`**. На исти начин нацртај остале три странице квадрата. Не заборави да после сваке наредбе притиснеш **Enter** на тастатури.



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> from turtle import *
>>> title ("Kornjača")
>>> showturtle ()
>>> forward (100)
>>> right (90)
>>> forward (100)
>>> right (90)
>>> forward (100)
>>> right (90)
>>> forward (100)
>>> right (90)
>>> |
```

о Слика 3.7.2. Цртање квадрата у програму Python

Као што видиш, у претходном примеру постоји један број наредби које се понављају. У оваквим случајевима, скраћенице за наредбе попут **fd** и **rt** за **forward** и **right** делимично ти помажу да поједноставиш куцање кода. Још једна корисна опција за скраћење кода у Python-у јесте то што можеш да напишеш две наредбе у истом реду. Потребно је да између њих ставиш тачку са запетом (;). На овај начин можеш писати и више од две наредбе у једном реду.

```
>>> from turtle import *
>>> title ("Kornjača")
>>> showturtle ()
>>> fd(100);rt(90)
>>> fd(100);rt(90)
>>> fd(100);rt(90)
>>> fd(100);rt(90)
>>>
```

о Слика 3.7.3. Краћи код за цртање квадрата у конзоли програма Python

ИНФО ПЛУС

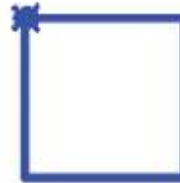


Када у конзоли имаш наредбе које се понављају у неколико линија кода, не мораш их изнова куцати у сваком реду. Довољно је да стрелицом нагоре с тастатуре доведеш курсор испред наредбе која се понавља **>>>|fd(100);rt(90)**. Затим притисни тастер **Enter**. Овим ћеш копирати наредбу у последњи ред (онај који је на реду за извршавање). Потребно је да још једном притиснеш **Enter** како би наредба била извршена. Приликом цртања квадрата понови овај поступак још двапут и квадрат је готов.

Следеће су наредбе којима подешаваш боју и дебљину линије нацртаних облика. Њих ћемо показати у *Python*-овом едитору. Програме написане у едитору можеш сачувати и касније користити и дорађивати. Едитор отвараш тако што изабереш опцију **New File** из **File** менија или преко тастерске пречице **Ctrl + N**.

На следећој слици је приказан програм за цртање квадрата. Библиотека се у едитору увози на једноставнији начин. Довољно је написати наредбу **import turtle**. Уместо стрелице, курсор је промењен у корњачу. Ово је постигнуто наредбом **turtle.shape("turtle")**. За линије је одабрана дебљина од 5 пиксела – **turtle.width(5)**. А за боју је узета плава – **turtle.color("blue")**. Наредбе за цртање и окретање курсора су сличне. Састоје се од назива библиотеке и назива функције из библиотеке, а раздвојене су тачком (**turtle.forward(100)**, **turtle.right(90)** итд.). Не заборави да провериш како програм ради када га откуцаш у едитору (*Run => Run Module* или *F5*).

```
import turtle
turtle.shape("turtle")
turtle.width(5)
turtle.color("blue")
turtle.forward(100)
turtle.right(90)
turtle.forward(100)
turtle.right(90)
turtle.forward(100)
turtle.right(90)
turtle.forward(100)
turtle.right(90)
```



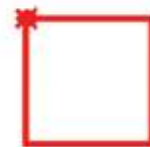
Слика 3.7.4. Програм за цртање квадрата у *Python*-овом едитору

ИНФО ПЛУС

У едитору, поред наведеног, можеш премештати и копирати делове кода помоћу познатих тастерских пречица (**Ctrl + X**, **Ctrl + C** и **Ctrl + V**).

И у едитору можеш комбиновати две наредбе у једном реду кода. То је приказано на следећој слици где је нацртан црвени квадрат. Такође су коришћене скраћенице за наредбе – **fd** уместо **forward** и **rt** уместо **right**.

```
import turtle
turtle.shape("turtle")
turtle.width(5)
turtle.color("red")
turtle.fd(100); turtle.rt(90)
turtle.fd(100); turtle.rt(90)
turtle.fd(100); turtle.rt(90)
turtle.fd(100); turtle.rt(90)
```



Слика 3.7.5. Краћи код програма за цртање квадрата у *Python*-овом едитору

На следећој слици је приказан програм за цртање троугла.

```
import turtle
turtle.shape("turtle")
turtle.width(9)
turtle.color("orange")
turtle.fd(100);turtle.rt(120)
turtle.fd(100);turtle.rt(120)
turtle.fd(100);turtle.rt(120)
```



Слика 3.7.6. Програм за цртање троугла

Код цртања геометријских облика, наредбе за промену боје линије можеш користити на више места. Тако ће линије бити другачије обојене (слика 3.7.7).

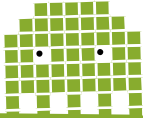
```
import turtle
turtle.shape("turtle")
turtle.width(4)
turtle.color("blue")
turtle.fd(100);turtle.rt(90)
turtle.color("green")
turtle.fd(50);turtle.rt(90)
turtle.color("blue")
turtle.fd(100);turtle.rt(90)
turtle.color("green")
turtle.fd(50);turtle.rt(90)
```



Слика 3.7.7. Програм за цртање правоугоника с линијама различитих боја

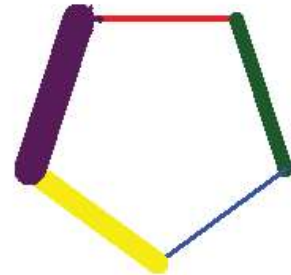
Поред различитих боја и дебљине линија се могу разликовати (слика 3.7.8). Навели смо да је у наредбама за окретање **right** и **left**, или скраћено **rt** и **lt**, у загради потребно откуцати степен угла за који се корњача окреће. Да не би рачунао/-ла колики је спољашњи угао једнакостраничног петоугла, можеш само поделити збир спољашњих углова с њиховим бројем ($360/5$). Збир спољашњих углова сваког многоугла је 360. За цртање шестоугла треба дефинисати степен окретања корњаче као $360/6$. О томе ћеш детаљно учити у математици следеће године. Овде ти само дајемо кратко упутство без објашњења да би могао/-ла да савладаш цртање геометријских облика у Python-у.

ВЕЖБАЈ



Покушај да нацрташ шестоугао тако да свака страница буде другачије обојена и различите дебљине. Програм сачувај у фолдеру *Documents* под називом *Sestougao*.

```
import turtle
turtle.shape("turtle")
turtle.width(4)
turtle.color("red")
turtle.fd(100); turtle.rt(360/5)
turtle.width(10)
turtle.color("green")
turtle.fd(100); turtle.rt(360/5)
turtle.width(3)
turtle.color("blue")
turtle.fd(100); turtle.rt(360/5)
turtle.width(14)
turtle.color("yellow")
turtle.fd(100); turtle.rt(360/5)
turtle.width(20)
turtle.color("purple")
turtle.fd(100); turtle.rt(360/5)
```



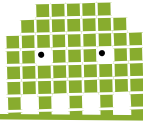
Слика 3.7.8. Програм за цртање петоугла с линијама различите дебљине

У библиотеци *turtle* постоје још неке функције за цртање које ти могу користити (табела 3.7.1). Испробај њихове функције играјући се. Цртај разне облике различитим бојама и линијама различитих дебљина. За неке цртеже биће потребно да корњачу помериш без цртања. Наредбом **`turtle.penup()`** корњача подиже своју оловку и након тога се креће не остављајући траг. Када јој издаш наредбу **`turtle.pendown()`** корњача поново почиње да оставља траг током кретања.

Табела 3.7.1. Функције за цртање из библиотеке *turtle*

Наредба	Објашњење
<code>turtle.forward(100)</code>	Корњача се помера напред за 100 корака.
<code>turtle.left(90)</code>	Корњача се окреће налево за 90 степени.
<code>turtle.right(90)</code>	Корњача се окреће надесно за 90 степени.
<code>turtle.penup()</code>	Корњача подиже оловку.
<code>turtle.pendown()</code>	Корњача спушта оловку.
<code>turtle.color("red")</code>	Боја оловке се поставља на ону која је наведена у загради под наводницима. У овом случају то је црвена боја.
<code>turtle.width(5)</code>	Дебљина оловке се поставља на вредност 5.
<code>turtle.stamp()</code>	Корњача оставља свој отисак.
<code>turtle.shape("turtle")</code>	Курсор мења свој облик (нпр. <code>"arrow"</code> , <code>"turtle"</code> , <code>"circle"</code>).
<code>turtle.speed(5)</code>	Корњача мења своју брзину кретања од 0 (најбрже) до 10 (најспорије).
<code>turtle.back(100)</code>	Корњача се помера уназад за 100 корака.
<code>turtle.reset()</code>	Цртеж се брише и корњача се враћа у средину прозора.

ВЕЖБАЈ

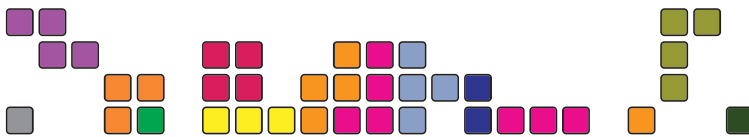


1. Уради све приказане примере у конзоли и у едитору. Покушај да их урадиш и на краћи и на дужи начин.
2. Напиши програме за цртање следећих цртежа:



ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Чему служе библиотеке у програмским језицима?
2. На које све начине можеш скратити програмски код у *Python*-у?
3. Која је разлика између писања програма у *Python*-овој конзоли и у *Python*-овом едитору?



УКРАТКО

Сваки програмски језик, поред основне библиотеке, има и библиотеке наредби и функција које се користе по потреби. Да би употребљавао/-ла одређену библиотеку, мораш је увести (*Import*) у програм. Осим за рачунање и рад са стринговима, *Python* можеш користити и за цртање. Потребно је увести библиотеку која садржи скуп наредби за цртање, а зове се **turtle**. Линију црташ наредбом **forward (100)** или краће **fd (100)**. Број у загради представља дужину линије у пикселима. Курсор окрећеш наредбом **right(90)** или **left(90)** при чему је у загради дат угао за који се курсор окреће. Боју линије мењаш наредбом **turtle.color**, док се за промену дебљине користи наредба **turtle.width**. Када хоћеш да две наредбе напишеш у истом реду између њих стави тачку са запетом (;).



3.8. ПОНАВЉАЊЕ

КЉУЧНЕ РЕЧИ

понављање, петља *for*, параметар, бројач, корак петље



Замисли да треба да напишеш програм који ће поруку *Zdravo, svete!* исписати десет пута на екрану. Сигурно ти прво пада на памет да наредбу `print("Zdravo, svete!")` поновиш десет пута па да извршиш програм (слика 3.8.1).

```
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  Zdravo, svete!  
print("Zdravo, svete!")  >>> |
```

Слика 3.8.1. Програм за исписивање 10 истих наредби на екрану

Добио/-ла си тачан резултат али да ли је ово најједноставније решење? Шта би било да треба исписати неку поруку 10000 пута?

Сви примери програма које смо ти до сада показали имали су **линијску структуру**. Наредбе су ређане једна испод друге и тим редоследом су извршаване. Често је било потребно поновити неку радњу више пута. Да би се избегло писање једне исте наредбе више пута, користе се **петље**. Петља је програмска структура која омогућава понављање једне наредбе или више наредби. Зато се каже да програми у којима се користе петље имају **цикличну програмску структуру**.

Најједноставнија врста петље јесте **for** петља. Свака **for** петља се састоји од заглавља петље и тела петље. У заглављу се налазе бројачка променљива и њена вредност (**n**). Бројачка променљива се чешће скраћено назива **бројач**. Као и сваку променљиву, бројач можеш назвати како год хоћеш. Ипак, он се у програмирању често означава латиничним малим словом **i**. У редовима испод заглавља дефинишу се наредбе. Наредбе представљају тело петље. Тело петље се извршава онолико пута колика је зада-та вредност бројача.



о Слика 3.8.2. Општа структура петље **for**

Претходни пример урађен помоћу петље **for** изгледа као на левој страни наредне слике. Задата вредност бројача је 10. Толико пута ће се тело петље извршити. Резултат видиш с десне стране. Овако би било једноставно и када би се тражио испис неке поруке 10000 пута па чак и много више од тога.

```
for i in range(10):
    print("Zdravo, svete!")
```

Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
Zdravo, svete!
>>> |

о Слика 3.8.3. Програм за исписивање једне исте наредбе 10 пута

```
for i in range(10):
    print("i")
```

```
i  
i  
i  
i  
i  
i  
i  
i  
i  
i  
i  
>>> |
```

о Слика 3.8.4. Наводници могу да направе и семантичке грешке

Како би сада написао/-ла програм за исписивање цифара од 1 до 10? Потребно је да у телу петље задаш наредбу за исписивање вредности бројача (слика 3.8.4). Али резултат приказан на слици 3.8.4. није оно што је тражено у задатку. Овде је направљена семантичка грешка. Потребно је обрисати наводнике.

ИНФО ПЛУС



Семантичке грешке спадају у грешке логичког типа. Њих програм неће пријавити јер оне не ометају његово извршење. Програм функционише али не онако како смо ми то замислили. Он не даје тражени резултат. Овакве грешке треба да открије програмер. То се најчешће дешава у фази тестирања програма. Дакле, семантика одређује значење наредби програмског језика, односно програма у целини.

```
for i in range(10):  
    print(i)
```

```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
>>>
```

o **Слика 3.8.5.** Програм за исписивање 10 цифара

Међутим, ни након уклањања наводника резултат не одговара захтеву задатка (слика 3.8.5). Тражене су цифре од 1 до 10, а приказане су цифре од 0 до 9.

Као што видиш, током извршавања петље **for** бројач почиње да узима вредности од нулте све до вредности дефинисане у загради умањене за 1. У првом извршавању бројач (променљива **i**) је узео вредност 0, у другом вредност 1, у трећем вредност 2 итд. Да би у десетом узео вредност 9.

Ово се може решити увођењем додатног параметра бројача. Ако у загради напишеш (1, 10) тада ће бројање кренути од 1. Међутим, завршиће се са 9. Бројач се увек завршава код броја који је за 1 мањи од његове вредности! То се догађа зато што се петља зауставља у последњем пролазу.

Да би као резултат биле приказане цифре од 1 до 10 за крајњу вредност бројача напиши 11 (слика 3.8.6).

Почетна вредност бројача

```
for i in range(1, 11):  
    print(i)
```

Крајња вредност бројача

o **Слика 3.8.6.** Програм за исписивање бројева с дефинисаном почетном и крајњом вредношћу бројача

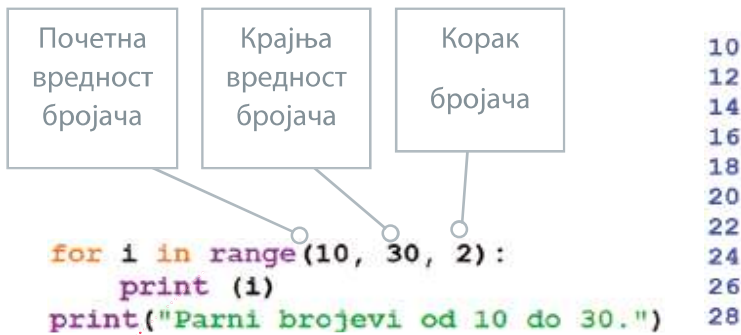
С обзиром на то да је бројач променљива, корисник може унети његове вредности. У следећем примеру погледај како изгледа програм који тражи од корисника да уноси два броја (**a** и **b**). Затим исписује све бројеве који се налазе између **a** и **b**.

```
a = int(input("Početni broj:")) Početni broj:5  
b = int(input("Završni broj:")) Završni broj:10  
for i in range(a,b):  
    print(i)  
5  
6  
7  
8  
9  
>>>
```

o **Слика 3.8.7.** Учитавање почетне и крајње вредности за бројач петље

Бројач за **for** петљу може имати и три параметра. Поред почетне и крајње вредности, можеш дефинисати и **корак** петље односно бројача. Он представља вредност за коју се бројач увећава при сваком пролазу кроз петљу.

Примера ради, треба исписати све парне бројеве од броја 10 до броја 30. Петља за решавање овог проблема је приказана на слици (3.8.8). Као што видиш, крајња вредност у резултату је 28. То се догодило зато што је корак бројача постављен на 2.



о Слика 3.8.8. Бројач с три параметра

Корак бројача може имати и негативну вредност. Тада ће бројање ићи уназад. Вредност бројача ће се смањивати. У таквим случајевима почетна вредност бројача петље мора бити већа од крајње.

Петља **for** се у великој мери употребљава при цртању геометријских облика. Сви примери из претходног поглавља се могу много лакше нацртати ако се примени петља **for**. Ево како би изгледао програм за цртање квадрата с петљом **for**.

```

import turtle
turtle.shape("turtle")
turtle.width(5)
turtle.color("red")
for i in range(4):
    turtle.fd(100)
    turtle.rt(90)

```



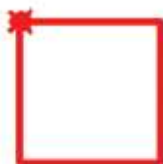
о Слика 3.8.9. Али зашто петља **for** не функционише?

Када покушаш да извршиш овако написан програм, јавиће се грешка. Она настаје зато што све наредбе у телу петље морају бити увучене.

```

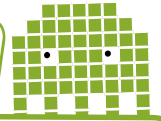
import turtle
turtle.shape("turtle")
turtle.width(5)
turtle.color("red")
for i in range(4):
    turtle.fd(100)
    turtle.rt(90)

```



о Слика 3.8.10. Наредбе испод петље **for** морају бити увучене

ВЕЖБАЈ



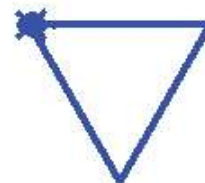
Испробај шта ће бити резултат програма ако за параметре бројача унесеш (40, 0, -3)?

САЗНАЈ ВИШЕ

Препоручује се да се наредбе у телу петље увлаче увек за исти размак у односу на први ред петље. За ово користите тастер **Tab** или четири пута притисните тастер **Space**.

Као што видиш на претходној слици, резултат програма за цртање квадрата с петљом **for** исти је као у претходном поглављу. Тада је програм имао линијску структуру. Међутим, овакав поступак је краћи и лакши. На једном месту је задато да корњача нацрта линију дужине 100 пиксела и да се окрене удесно за 90 степени. У петљи је дефинисано да се овај поступак понови 4 пута. На сличан начин се могу нацртати и остали геометријски облици.

```
import turtle
turtle.shape("turtle")
turtle.width(4)
turtle.color("blue")
for i in range(3):
    turtle.fd(100);turtle.rt(120)
```



```
import turtle
turtle.shape("turtle")
turtle.width(4)
turtle.color("purple")
for i in range(5):
    turtle.fd(100);turtle.rt(360/5)
```



```
import turtle
turtle.shape("turtle")
turtle.width(4)
turtle.color("yellow")
for i in range(6):
    turtle.fd(100);turtle.rt(60)
```



```
import turtle
turtle.shape("turtle")
turtle.width(4)
turtle.color("green")
for i in range(360):
    turtle.fd(1);turtle.rt(1)
```

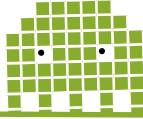


о Слика 3.8.11. Програми за цртање облика помоћу петље **for**

ИНФО ПЛУС

Петља коју смо ти у овој лекцији приказали је безусловна. Њено извршавање не зависи ни од каквог услова. Само је потребно задати колико пута петља треба да се изврши. Прешле године си научио/-ла да осим безусловне петље постоји и условна. Број понављања условне петље зависи од постављеног услова. Наредбе у телу условне петље извршаваће се све док је услов испуњен. У *Python*-у се условна петља дефинише кључном речју **while**. Она се често назива петља **while**. Њом ћеш се детаљније бавити у старијим разредима.

ВЕЖБАЈ

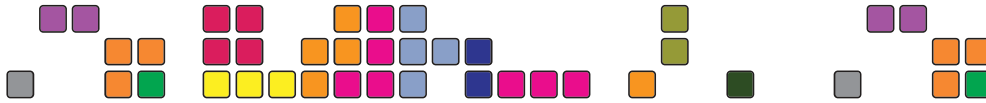


1. На десктопу направи фолдер под називом *Zadaci iz Pajtona*. Уради све примере програма из овог поглавља и сачувај их у наведеном фолдеру. Све сачуване програме из претходних поглавља премести у фолдер *Zadaci iz Pajtona*.
2. Помоћу петље **for** нацртај црвени шестоугао с линијама дебљине 10 пиксела.
3. Четврти задатак из претходне лекције уради помоћу петље **for**.
4. У игри жмурке деца броје 5, 10, 15, 20...100. Напиши програм који испишује баш те бројеве помоћу петље **for**.
5. Напиши програм који у интервалу од 5 до 85 испишује све бројеве дељиве са 6.



ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. У чему је разлика између програма линијске структуре и програма цикличне структуре?
2. Које су користи од употребе петљи у програмирању?
3. Како се користи петља **for** у раду са стринговима, а како у раду с бројевима?
4. Од чега се састоји петља **for**?



УКРАТКО

Да би се избегло писање једне исте наредбе више пута, користе се петље. Петља је програмска структура која омогућава понављање једне наредбе или више наредби. Зато се каже да програми у којима се користе петље имају цикличну програмску структуру. Најједноставнија врста петље јесте **for** петља. Свака **for** петља се састоји од бројачке променљиве и њене вредности (**n**). Бројачка променљива се чешће скраћено назива бројач. Бројач за **for** петљу може имати и три параметра. Поред почетне и крајње вредности, можеш дефинисати и корак петље односно бројача. Он представља вредност за коју се бројач увећава при сваком пролазу кроз петљу. Корак бројача може имати и негативну вредност. Тада ће бројање ићи уназад.

3.9. ГРАНАЊЕ

КЉУЧНЕ РЕЧИ

услов, одлука, ток програма,
оператори поређења, логички оператори



о Слика 3.9.1. Ток програма зависи од испуњења услова

Програми поред линијске и цикличне структуре могу имати разгранату структуру. Разграната структура подразумева да у програму постоје места где се извршавање програма наставља током који зависи од постављеног услова. Ако је услов испуњен, програм наставља једним током, а ако није, креће другим током.

Оператори поређења код наредбе гранања

Да би се испитало да ли је услов испуњен (ДА) или није (НЕ), користе се оператори поређења. Помоћу њих можеш утврдити да ли је нека вредност мања од вредности поређења у оквиру услова, већа од ње или једнака њој. Такође можеш утврдити да ли је вредност мања или једнака, већа или једнака или је различита од оне с којом се пореди.



Оператор поређења једнако се у многим програмским језицима пише као `==`. Разлог је то што се `=` користи као оператор доделе вредности променљивама.

	Математика	Python
Мање од	$<$	<code><</code>
Веће од	$>$	<code>></code>
Једнако	$=$	<code>==</code>
Мање или једнако од	\leq	<code><=</code>
Веће или једнако од	\geq	<code>>=</code>
Различито од	\neq	<code>!=</code>

о Табела 3.9.1. Оператори поређења у математици и у Python-у

ЗАНИМЉИВОСТ

Знак једнакости (`=`) изумео је 1557. године велшки математичар Роберт Рекорд.

На пример, ако је температура воде у базену већа од 25 степени, отићи ћу на пливање. Или, ако је цена сладоледа мања од 100 динара или једнака томе, купићу један. Услови исказани говорним језиком се на веома једноставан начин преводе у наредбе програмског језика.

У Python-у се услов поставља наредбом гранања **if**. Она може имати непотпуни и потпуни облик. Услов на слици 3.9.2.a представља непотпуни облик наредбе гранања – када услов није испуњен прелази се на извршавање следеће наредбе програма. Потпуни облик био би када би било предвиђено шта да се ради ако услов није испуњен, односно ако је нетачан (слика 3.9.2b). Шта ако температура воде није већа од 25 степени или ако цена сладоледа није мања од 100 динара или није једнака томе.

Веома је важно запамтити да се постављени услов у наредби гранања завршава двотачком. Исто важи за кључну реч **else** (у супротном). Двотачка у оба случаја означава крај реда и прелазак на извршавање наредби.

Сада ћемо ти на неколико примера показати како изгледа постављање и испитивање услова помоћу наредбе гранања и оператора поређења.

За почетак, хајде да направимо програм који проверава знање ћака шестака из историје. Програм прво испитује питање „Које године се одиграла Маричка битка?“ и тражи одговор. Ако је одговор = 1371, програм ће исписати поруку „Тачан одговор!“.

```
Odgovor = int(input("Koje godine se odigrala Marička bitka?"))
if Odgovor == 1371:
    print ("Tačan odgovor!")
```

Koje godine se odigrala Marička bitka?1371
Tačan odgovor!
>>> |

о Слика 3.9.3. Програм с непотпуном наредбом гранања

Али шта ако одговор није тачан? Најчешће је потребно предвидети акцију и за такав резултат. Најлогичније је да програм испише поруку о томе у случају да унети резултат није тачан (слика 3.9.4). Ову наредбу уносиш испод кључне речи **else** иза које треба да стоји двотачка.

```
Odgovor = int(input("Koje godine se odigrala Marička bitka?"))
if Odgovor == 1371:
    print ("Tačan odgovor!")
else:
    print("Netačan odgovor!")
```

Koje godine se odigrala Marička bitka?1389
Netačan odgovor!
>>> |

о Слика 3.9.4. Програм с потпуном наредбом гранања

Извршава се ако је услов испуњен

if uslov:

naredba_1

(a)

Извршава се ако је услов испуњен

if uslov:

naredba_1

else:

naredba_2

(б)

Извршава се ако услов није испуњен

о Слика 3.9.2. (a) Непотпуни облик наредбе гранања и (б) Потпуни облик наредбе гранања

Осим наредбе која се извршава у случају да услов није испуњен, у претходном примеру смо урадили још једну корисну ствар. Упореди резултате овог програма и оног са слике 3.9.2. Одговор у овом примеру унет је у нови ред. То смо постигли на следећи начин. Откуцали смо `\n` на крају питања на које очекујемо да корисник унесе одговор. Ово смо ти показали у лекцији о стринговима. Ипак, много је zgodније употребљавати ову опцију у програмима који траже од корисника да унесе неке вредности.

У наредном примеру смо користили оператор поређења `ВЕЋЕ` `ОД` како бисмо испитали да ли је температура воде погодна за купање. Ако је унета температура већа од 25, програм ће испитати поруку „Можемо на купање!”. У супротном, исписаће поруку „Voda nije dovoljno topla.”

```
Temperatura = int(input("Kolika je temperatura vode?\n"))
if Temperatura > 25:
    print ("Možemo na kupanje!")
else:
    print ("Voda nije dovoljno topla.")
```

о Слика 3.9.5. Потпуно гранање са оператором `ВЕЋЕ`

На сличан начин можеш одлучити да ли ћеш купити сладолед (слика 3.9.6).

```
Cena_sladoleda = int(input("Unesi cenu sladoleda:\n"))
if Cena_sladoleda <=70:
    print("Kupiću jedan sladoled!")
else:
    print("Nemam dovoljno novca.")
```

о Слика 3.9.6. Потпуно гранање с оператором `МАЊЕ` `ОД` ИЛИ `ЈЕДНАКО`

Оператор поређења различито од `(!=)` функционише обрнуто од оператора једнако (слика 3.9.7). За све вредности које јесу различите од оне с којом се врши поређење, услов је испуњен. Услов није испуњен само у случају када је унета вредност иста као вредност с којом се врши поређење. Тада се извршава наредба која следи иза кључне речи **у супротном** (`else`).

```
Odgovor = int(input("Koliko žica imaju drveni gudački instrumenti? "))
if Odgovor != 4:
    print("Greška!")
else:
    print("Bravo, tačan odgovor.")
```

о Слика 3.9.7. Потпуно гранање с оператором `РАЗЛИЧИТО`

У претходном примеру нисмо користили опцију `\n` која учитавање вредности пребацује у нови ред. Урадили смо нешто друго. Направили смо један размак пре затварања наводника на крају питања. Тако се, када покренеш програм, прави размак између питања и унетог одговора (слика 3.9.8).

```
Koliko žica imaju drveni gudački instrumenti? 4
Bravo, tačan odgovor.
```

```
>>> |
```

о Слика 3.9.8. Размак између питања и унетог одговора

Код наредби гранања, можеш поредити неку вредност с вредношћу променљиве на коју је примењена аритметичка операција. У примеру који следи унети број се дели бројем 2 али тако да целобројни остатак буде 0 (`Број%2 == 0`). На тај начин можеш видети да ли је било који унети број паран.

```
Broj = int(input("Unesi broj: "))
if Broj % 2 == 0:
    print("Broj je paran.")
else:
    print("Broj je neparan.")
```

о Слика 3.9.9. Приликом поређења вредности, на променљиве можеш примењивати аритметичке операције

Логички оператори

У овој лекцији је у свим примерима до сада вршено поређење вредности у оквиру једног израза. Када је потребно да испиташ тачност сложених израза, можеш користити логичке операторе. Логички оператори које ћемо ти показати су **and**, **or** и **not** (и, или и није). Операторима **and** и **or** можеш комбиновати два једноставна израза или више њих и правити сложен израз чија се тачност испитује.

- Да би израз с оператором **and** био тачан, сви изрази који га чине морају бити тачни.
- Израз с оператором **or** биће тачан, ако је и само један од израза који га чине тачан.

На пример, ако на полугодишту II, на крају VI разреда, будеш одличан/-на са свим петицама добићеш нов бицикл.

```
Polugodište = float(input("Unesi prosek na kraju I polugodišta: "))
Kraj = float(input("Unesi prosek na kraju VI razreda: "))
if (Polugodište == 5.00 and Kraj == 5.00):
    print ("Dobijam nov bicikl!")
else:
    print ("Nisam ispunio/la sve uslove...")
```

о Слика 3.9.10. Код оператора *and* сви изрази морају бити тачни

Некада за нов бицикл може бити довољно да постигнеш одличан успех на полугодишту ИЛИ на крају године. У том случају у претходном примеру уместо оператора **and** можеш користити оператор **or**.

```
Polugodište = float(input("Unesi prosek na kraju I polugodišta: "))
Kraj = float(input("Unesi prosek na kraju VI razreda: "))
if (Polugodište == 5.00 or Kraj == 5.00):
    print ("Dobijam nov bicikl!")
else:
    print ("Nisam ispunio/la sve uslove...")
```

```
Unesi prosek na kraju I polugodišta: 5.00
Unesi prosek na kraju VI razreda: 4.20
Dobijam nov bicikl!
```

о Слика 3.9.11. Код оператора *or* само један од израза мора бити тачан



Уради претходни пример тако да следи да нов бицикл добијаш ако постигнеш одличан успех на било ком тромесечју или полугодишту или на крају школске године. Притом не мораш имати све петице већ је довољно да ти просек буде већи од 4,50. Програм сачувај у фолдеру *Zadaci iz Pajtona*.



У једном програму можеш комбиновати различите логичке операторе. На пример, у биоскопу се филмови приказују увек у два термина, од 18 до 20 часова и од 21 час до 23 часа. Напишимо програм који проверава да ли је у одређеном тренутку пројекција филма у току.

```
sati = int(input("Unesi vreme u satima: "))
minuti = int(input("Unesi vreme u minutima: "))
if (sati >= 18 and sati <=20) or (sati >= 21 and sati <=23):
    print ("U toku je projekcija filma.")
else:
    print ("U bioskopu trenutno nema projekcije.")
```

```
Unesi vreme u satima: 14
Unesi vreme u minutima: 45
U bioskopu trenutno nema projekcije.
```

```
Unesi vreme u satima: 19
Unesi vreme u minutima: 30
U toku je projekcija filma.
```

о Слика 3.9.12. У програму можеш комбиновати различите логичке операторе

Логички оператор негације (**not**) сличан је оператору поређења различито од. Резултат услова с оператором **not** је тачан када је израз нетачан и обрнуто. Ево како би задатак с бројем страница квадрата могао бити урађен коришћењем оператора негације. Уколико је израз нетачан (одговор није једнак броју 4) резултат услова је тачан. Тада програм исписује прву поруку. Ако је израз тачан (одговор је једнак 4), резултат услова је нетачан и програм извршава наредбу која следи иза **else**:

```
Odgovor = int(input("Koje godine je bio Kosovski boj?"))
if not Odgovor == 1389:
    print("Greška, netačan odgovor.")
else:
    print("Bravo, tačan odgovor.")
```

о Слика 3.9.13. Оператор *not* ради на сличан начин као *!=*

Некада ти за комбиновање више услова нису потребни логички оператори. У програму можеш имати два или више услова који искључују један други. Замисли да за сваки од унетих одговора програм извршава различиту наредбу (или више њих). На пример, програм може израчунати обим троугла, квадрата или правоугаоника. За ово су ти потребне три наредбе гранања (слика 3.9.14).

```
Odgovor = input("Čiji obim želiš da izračunaš? T/K/P: ")
if Odgovor == "T":
    a = int(input("Unesi stranicu a:"))
    Obim_trougla = 3*a
    print ("Obim trougla je", Obim_trougla, "cm")
if Odgovor == "K":
    a = int(input("Unesi stranicu a:"))
    Obim_kvadrata = 4*a
    print ("Obim kvadrata je", Obim_kvadrata, "cm")
if Odgovor == "P":
    a = int(input("Unesi stranicu a:"))
    b = int(input("Unesi stranicu b:"))
    Obim_pravougaonika = 2*a + 2*b
    print ("Obim pravougaonika je", Obim_pravougaonika, "cm")
```

```
Čiji obim želiš da izračunaš? T/K/P: T
Unesi stranicu a:3
Obim trougla je 9 cm
```

о Слика 3.9.14. Три услова у једном програму

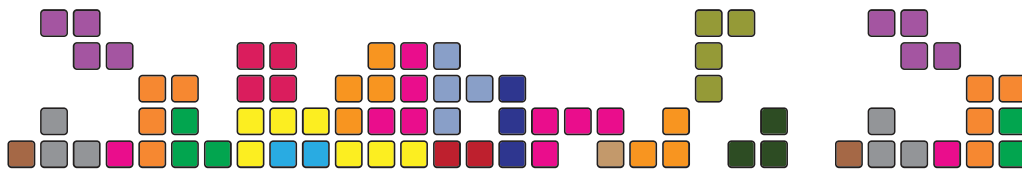
ПИТАЊА ЗА ПРОВЕРУ ЗНАЊА

1. Која је разлика између оператора поређења и логичких оператора?
2. Да ли постоји сличност између неких оператора поређења и логичких оператора?
3. Шта је мана наредбе гранања непотпуног облика?
4. Којом би се од следећих наредби добио исти резултат као наредбом на слици 3.9.15? Објасни где су грешке код осталих наредби.

a) `if not (Odgovor == "P" or Odgovor == "K" or Odgovor == "T"):`
`print ("Uneo/la si pogrešno slovo!")`

б) `if not (Odgovor = "P" or Odgovor = "K" or Odgovor = "T"):`
`print ("Uneo/la si pogrešno slovo!")`

в) `if not (Odgovor == "P" and Odgovor == "K" and Odgovor == "T"):`
`print ("Uneo/la si pogrešno slovo!")`



УКРАТКО

Разграната структура подразумева да у програму постоје места где се извршавање програма наставља током који зависи од постављеног услова. Ако је услов испуњен, програм наставља једним током, а ако није, креће другим током. Да би се испитало да ли је услов испуњен (ДА) или није (НЕ), користе се оператори поређења (<, >, =, <=, >=, !=). У *Python*-у се услов поставља наредбом гранања **if**. Она може имати непотпуни и потпуни облик. Веома је важно запамтити да се постављени услов у наредби гранања завршава двотачком. Када је потребно да испиташ тачност сложених израза, можеш користити логичке операторе (AND, OR и NOT). Логички оператор негације (**not**) сличан је оператору поређења различито од.