

Инсталирање и покретање библиотеке PyGame

3.2.

Пајтон (Python), PyGame, инсталирање, покретање.

Кључне речи

Подсети се:

- ▶ шта су програми;
- ▶ којој врсти програмских језика припада Пајтон, визуелној или текстуалној;
- ▶ на који начин и одакле програми могу безбедно да се преузму и инсталирају на рачунару;
- ▶ библиотеке уз помоћ које си цртао/цртала у Пајтону.

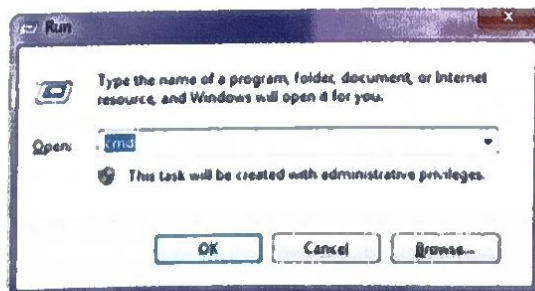
Програмски језик Пајтон нуди могућност исцртавања облика, приказа слика, креирања анимација и симулација у склопу своје стандардне библиотеке корњача графика, о којој је већ било речи. За напредније могућности и програмирање игара обично се користи нека напреднија библиотека. Једна таква библиотека, погодна за учење програмирања рачунарске графике, јесте **PyGame**. Она се развила почетком 21. века. Како јој и само име каже, служи за програмирање игара помоћу програмског језика Пајтон. Помоћу ње и њених модула можеш да црташ графичке облике, приказујеш слике, креираш анимације.

PyGame је изузетно флексибилна библиотека и лако се преноси на било коју другу платформу или у оперативни систем.

Као и програмски језик Пајтон, и његова библиотека PyGame потпуно је бесплатна. Ради на свим оперативним системима. Лако и једноставно се преузима са званичног сајта и брзо се инсталира.

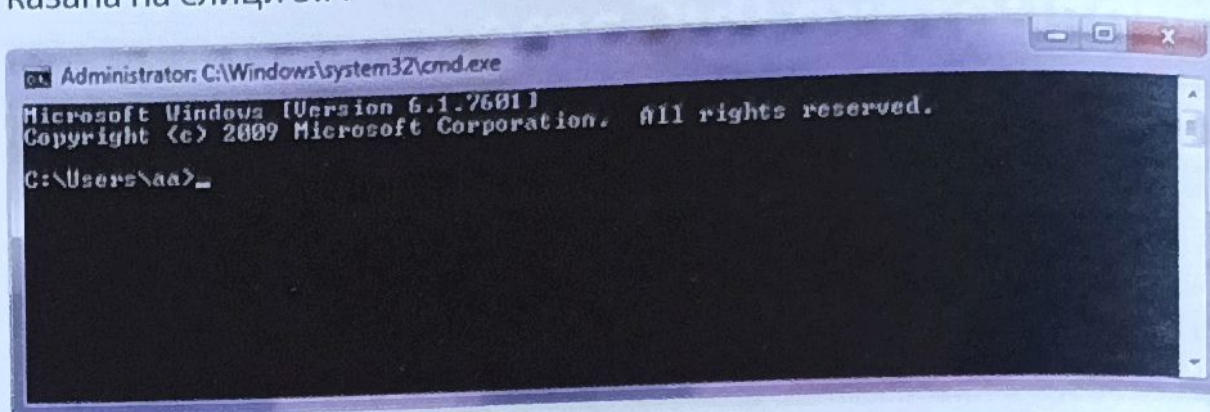
Да би на рачунару било могуће програмирати уз коришћење библиотеке PyGame, потребно ју је претходно инсталирати. Предуслов за то је да на рачунару имаш инсталиран Пајтон (пожељно је верзију 3.6 или новију), јер тек тада можеш прећи на инсталирање библиотеке PyGame.

На почетку је неопходно активирати командну линију и такошто држиш тастер **Windows** и истовремено притиснеш тастер **R**. Добијаш прозор **Run** у коме куцаш **cmd** (сл. 3.6.).



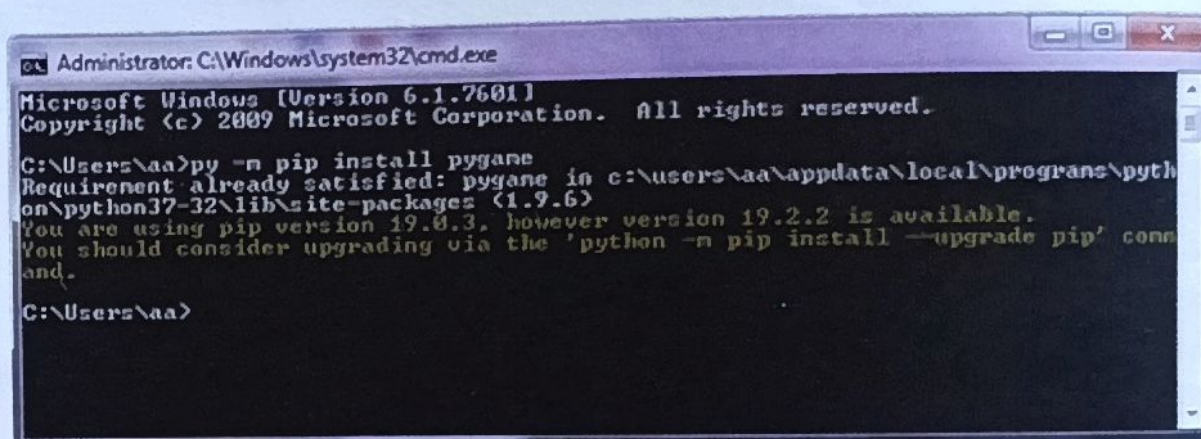
Сл. 3.6. Отварање прозора Run

Затим, притиснеш **OK** или **Enter** и отвориће се командна линија приказана на слици 3.7.



Сл. 3.7. Командна линија

У командном прозору треба да укуцаш наредбу `py -m pip install pygame`. Након покретања добићеш поруку као на слици 3.8., што значи да си PyGame успешно инсталирао/инсталирала.



Сл. 3.8. Успешно инсталирана библиотека PyGame

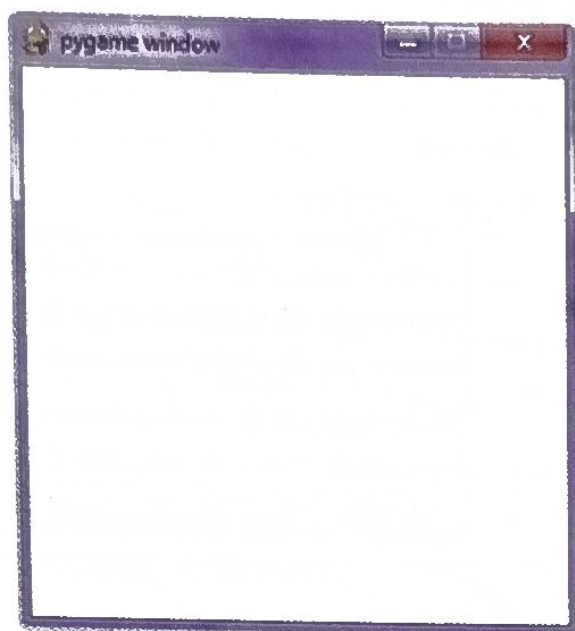
Када извршиш инсталирање, најбоље је да одмах тестираш да ли је све протекло како треба тако што ћеш:

- покренути Пајтон развојно окружење (IDLE) које је инсталирано као Windows апликација;
- у развојном окружењу IDLE отворити нови пројекат (опција File → New File);
- у едитору који се појави откуцати програм који је приказан у тексту:

```
import pygame
pygame.init()
prozor=pygame.display.set_mode((300, 300))
prozor.fill(pygame.Color("white"))
pygame.display.update()
pygame.time.wait(3000)
pygame.quit();
```

- снимити програм у датотеку пре покретања (опција File → Save as...);
- покренути програм (опција Run → Run Module, тј. притиском F5 на тастатури).

Након покретања програма треба да се појави прозор беле боје 300 x 300 пиксела. Он стоји приказан три секунде (сл. 3.9.)



Сл. 3.9. Изглед иницијалног програма

Основна структура библиотеке PyGame

иницијализација, искључивање, отварање прозора, димензије прозора, наслов прозора.

Кључне речи

Подсети се:

- ▶ шта чини основну структуру Пајтона;
- ▶ има ли разлике приликом коришћења великих и малих слова;
- ▶ да ли је битан редослед уношења програмских кодова.

PyGame омогућава да се цртају различити облици (дужи, правоугаоници, кругови, елипсе, многоуглови, кружни лукови и слично). Сви они могу бити обојени различитим бојама. Поред тога, могуће је на екранима приказивати и слике (учитане из .png или .jpg датотека), а могуће је исписивати и текст (коришћењем различитих фонтова).

Сви програми PyGame имају неколико незаобилазних корака. Програм садржи део кода:

- који отвара прозор у ком се програм извршава (задају се његове димензије и наслов);
- у којем се исцртава неки садржај на прозору;
- који анализира и реагује на акције које корисник извршава (мишем или тастатуром);
- који на крају програма искључује прозор.

Иницијализација и искључивање рада библиотеке PyGame

Програми у којима се користи библиотека PyGame обично имају следећи скелет:

Линија кода	Значење сваке линије појединачно
<code>import pygame as pg</code>	укључује се коришћење библиотеке PyGame и задаје јој се скраћени назив <code>pg</code>
<code>pg.init()</code>	укључује се, тј. иницијализује библиотека PyGame
<code>pg.quit()</code>	искључује рад библиотеке PyGame

Отварање прозора

Наредни корак у програму је отварање прозора. Да бисмо отворили прозор, неопходно је навести следеће линије програма:

Линија кода	Значење сваке линије појединачно
<code>pg.display.set_caption("Naslov prozora")</code>	Прва линија позива функцију <code>pg.display.set_caption()</code> која поставља наслов прозора на текст који се задаје као аргумент (текст се наводи у облику ниске карактера, тј. стринга, унутар једноструких или двоструких наводника, нпр. 'Наслов прозора' или "Наслов прозора").
<code>(sirina,visina)=(500, 500)</code>	Пошто ће се димензије прозора често користити за израчунавање положаја неких објеката, добра пракса је да се пре позива <code>pg.display.set_mode()</code> те димензије запамте у две променљиве (обично их називамо <code>sirina</code> и <code>visina</code>).
<code>prozor=pg.display.set_mode((sirina, visina))</code>	Функција <code>pg.display.set_mode()</code> укључује приказ прозора, а аргумент је уређени пар који садржи димензије тог прозора у пикселима (ширину и висину). Наравно, могуће је било димензије навести у облику константи <code>pg.display.set_mode(500,500)</code> . Функција <code>pg.display.set_mode</code> враћа вредност коју је веома важно упамтити (за то ће се обично користити променљива <code>prozor</code>), јер ће та вредност у наставку бити коришћена за свако цртање по прозору.

ЗАПАМТИ

Основни делови програмирања у библиотеци PyGame су:

- део програма који поређе PyGame и отвара прозор беле боје, димензија 300 x 300 пиксела, који се задаје наредбом приказаном у наставку:

```
import pygame  
pygame.init()  
prozor = pygame.display.set_mode((300, 300))  
prozor.fill(pygame.Color("white"));
```

- део програма који освежава прозор и чека 3 секунде (3000 милисекунди) пре него што се прозор сам затвори приказан је у наставку:

```
pygame.display.update()  
pygame.time.wait(3000)  
pygame.quit().
```

Скраћени назив библиотеке PyGame, којим се наглашава да ће у програму бити коришћена ова библиотека, јесте `import pygame as pg`.

Наке од основних наредби у раду са овом библиотеком су:

- `pg.init()` – за укључивање библиотеке PyGame;
- `pg.quit()` – за искључивање рада библиотеке PyGame;
- `pg.display.set_caption("Naslov prozora")` – за наслов прозора;
- `(sirina, visina) = (500, 500)` – за димензије прозора;
- `prozor = pg.display.set_mode((sirina, visina))` – за приказ прозора.

ПРОВЕРА ЗНАЊА

1. Цртање којих облика омогућава библиотека PyGame?
2. Како се покреће библиотека PyGame?
3. Наведи основне наредбе које се користе за рад у библиотеци PyGame.

3.3.

Координатни систем и цртање дужи

Кључне речи

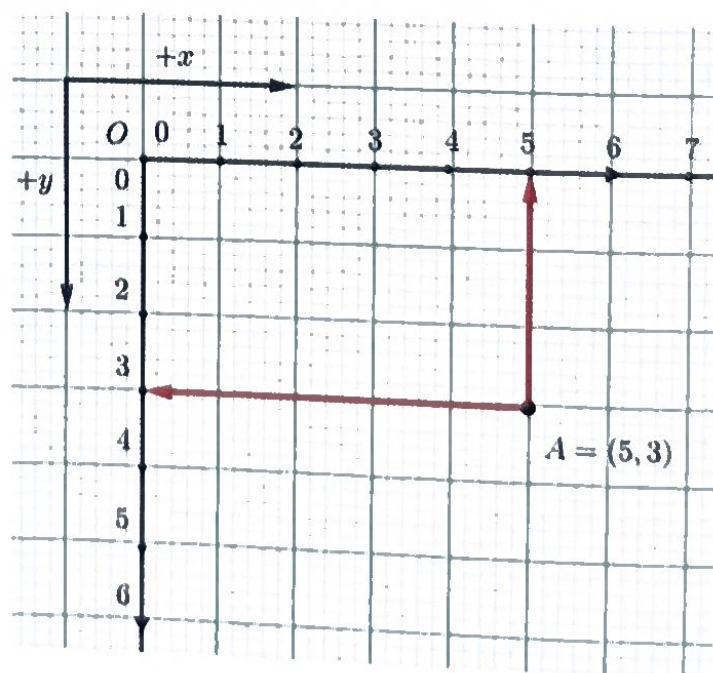
координатни систем, прозор, темена дужи, дебљина дужи, затварање прозора.

Подсети се:

- шта чини координатни систем у математици;
- распореда позитивних и негативних вредности x и y осе у математици;
- шта је дуж и чиме је дуж одређена.

Положај свих објеката (тачка, дужи, кругова и слично) на прозору одређује се њиховим координатама у **координатном систему прозора**, који је мало другачији у односу на онај са којим си се срео/срела у математици и у програмском језику Scratch у 5. разреду. Положај тачке је и у овом случају одређен уређеним паром њених координата (координатом x , тј. апсцисом, и координатом y , тј. ординатом). Јединица мере је пиксел. Координатни почетак је овде постављен у **горњем левом углу** прозора.

Координата x и овде расте када се крећемо надесно (као што смо већ учили у математици), али координата y опада када се крећемо нагоре, односно повећава се када се крећемо надоле, што је другачије у односу на координатни систем који нам је познат од раније. Нека је дата тачка $A(5,3)$ (сл. 3.10.). Прва координата тачке одређује колико је тачка удаљена од леве ивице прозора, а друга координата колико је тачка удаљена од горње ивице прозора.

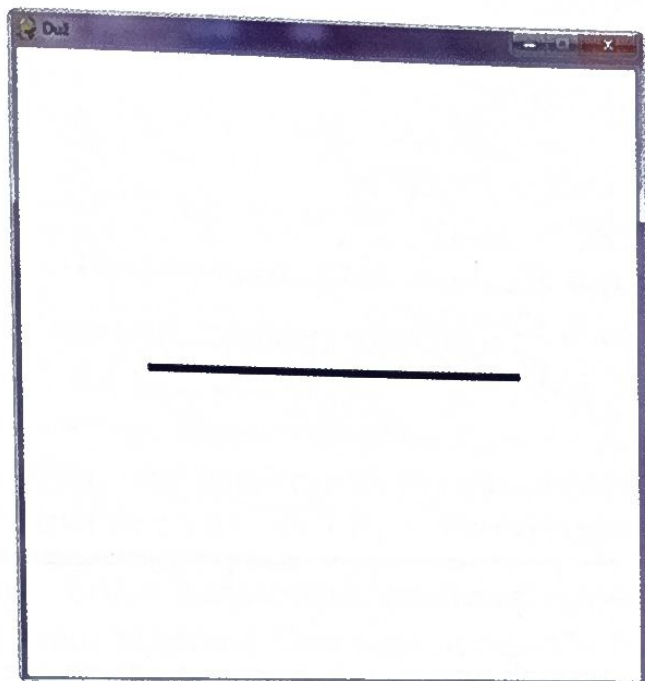


Сл. 3.10. Координатни систем

У програмском језику Пајтон пар координата тачке можемо представити двочланом торком $(3, 5)$ или двочланом листом $[3, 5]$.

Цртање дужи

За цртање дужи, поред већ наведених елемената, неопходно је додати још један ред који ће управо омогућити исцртавање задате дужи. У следећем примеру дата је хоризонтална дуж приказана на слици 3.11.



Сл. 3.11. Хоризонтална дуж

Линија кода	Значење сваке линије појединачно
<code>prozor.fill(pg.Color("white"))</code>	Омогућава да се позадина прозора обоји у белу боју (функција <code>prozor.fill()</code> проузрокује бојење позадине, док је <code>pg.Color("white")</code> позив функције који враћа белу боју).
<code>pg.draw.line(prozor, pg.Color("black"), (100,250), (400,250), 5)</code>	Ова функција има пет параметара који се наводе у загради. Први параметар је прозор на коме желимо да нацртамо дуж (његу добијамо као резултат позива функције <code>pg.display.set_mode()</code>). Други параметар је боја, наредни параметри су координате почетног темена дужи и крајњег темена дужи. Последњи параметар је дебелина линије којом се исцртава дуж, задата у пикселима.
<code>pg.display.update()</code>	Да би се слика приказала на екрану, потребно је позвати функцију <code>pg.display.update()</code> , која проузрокује освежавање прозора. Уместо <code>pg.display.update()</code> , могуће је позвати и <code>pg.display.flip()</code> .
<code>pg.time.wait(2000)</code>	Након исцртавања налази се наредба <code>pg.time.wait(2000)</code> којом се програм пазира на 2 секунде (ова функција очекује број милисекунди, па, пошто у једној секунди има 1000 милисекунди, 2 секунде се задају као 2000). Након паузе долази се до наредбе за искључивање рада библиотеке (самим тим, и до искључивања рада прозора).

Комплетан пример програма, у којем се у прозору 500 x 500 пиксела црта хоризонтално постављена дуж дебљине 5 пиксела, а прозор се затвара након две секунде, изгледа овако:

ПРИМЕР 1

```
import pygame as pg
pg.init()
pg.display.set_caption("Duž")
(sirina, visina)=(500, 500)
prozor=pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))
pg.draw.line(prozor, pg.Color("black"), (100, 250), (400, 250), 5)
pg.display.update()
pg.time.wait(2000)
pg.quit()
```



Током паузе не може чак ни да се искључи прозор.

Током паузе од 2 секунде програм је потпуно замрзнут и не реагује ни на какве акције корисника. Много боље решење од овога било би да корисник сам искључи прозор када то пожели. Управо из тог разлога приказаће се структура таква да ће се након иницијализације прозора и исцртавања садржаја у њему чекати да корисник искључи прозор, након чега ће се завршавати рад програма. Најкраћи начин да се то уради приказан је у наставку.

```
while pg.event.wait().type!=pg.QUIT:
    pass
```

У склопу услова петље позива се функција `pg.event.wait()`, која чека да се догоди наредни догађај, односно да се кликне на екран. Када се то догоди, односно када се кликне, упоређује се клик са типом `pg.QUIT`, који представља догађај искључивања прозора. Ако је тип догађаја који се десио различит од `pg.QUIT` (ако је кликнуто на други део прозора), то значи да корисник још није искључио прозор и извршава се тело петље које је, заправо, празно, па се поново прелази на проверу услова петље у ком се чека нови догађај. Ако је тип догађаја баш `pg.QUIT`, то значи да је корисник управо искључио прозор.

Комплетан програм који уместо паузе од 2 секунде садржи реаговање на догађаје приказан је у наставку.

ПРИМЕР 2

```
import pygame as pg
pg.init()
pg.display.set_caption("Duž sa klikom na X")
(sirina, visina) = (500, 500)
prozor = pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))
pg.draw.line(prozor, pg.Color("black"), (100, 250), (400, 250), 5)
pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

Ово је најједноставнији пример употребе библиотеке PyGame и захтева неколико корака. Велики број програма које ћеш писати у наставку имаће потпуно идентичну структуру, а једини део који ће се мењати биће онај у ком се врши цртање (у овом програму то је део који се састоји од позива `prozor.fill(...)` и `pg.draw.line(...)`).

Како би ово што боље запамтио/запамтила, провежбаћеш цртање дужи кроз неколико задатака. Ови задаци помоћи ће ти да боље упознаш координатни систем који користимо за позиционирање тачака на прозору.

ЗАДАТАК

1. Допуни програм (уместо звездица) који исцртава симбол плус у центру прозора димензија 400 x 400 пиксела. Дужина обе дужи је по 200 пиксела, а дебљина је 10 пиксела. Уведена је променљива `debljina` да би дебљина обе линије могла да се контролише само на једном месту.

```
import pygame as pg
pg.init()
pg.display.set_caption("Simbol plus")
(sirina, visina) = (400, 400)
prozor = pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))
debljina = 10

#vertikalna crna linija dužine 200 piksela
pg.draw.line(prozor, pg.Color("black"), (200, 100), (200, 300), debljina)
# horizontalna crna linija dužine 200 piksela
*****

#osvežavamo sadržaj prozora i tako prikazujemo ono što smo nacrtali
pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

Након знака `#` пишу се коментари који служе за лакше сналажење у програмском коду – то није линија кода.

2. Допуни програм који помоћу две дужи исцртава слово Т у центру прозора димензија 300 x 300 пиксела. Дужина обе дужи је по 200 пиксела, а дебљина је 15 пиксела.

```
import pygame as pg
pg.init()
pg.display.set_caption("Slovo T")
(sirina, visina) = (300, 300)

prozor=pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))
debljina=15

#vertikalna linija dužine 200 piksela
pg.draw.line(prozor, pg.Color("black"), (***,***), (***,***), debljina)

#horizontalna linija po sredini
pg.draw.line(prozor, pg.Color("black"), (***,***), (***,***), debljina)

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

ЗАПАМТИ

Координатни почетак у библиотеци PyGame поставља се у горњем левом углу прозора.

Тачка се представља уређеним паром, при чему прва вредност одређује колико је тачка удаљена од леве ивице прозора, а друга вредност колико је тачка удаљена од горње ивице прозора.

За цртање дужи користимо функцију:

```
pg.draw.line(prozor, pg.Color("black"), (100, 250),
              (400, 250), 5).
```

Ова функција садржи пет параметара који се наводе у загради. Први параметар је прозор на коме се црта дуж, други параметар је боја, наредни параметри су координате почетног темена дужи и крајњег темена дужи, док је последњи параметар дебљина дужи задата у пикселима.

Коришћењем наредбе

```
while pg.event.wait().type != pg.QUIT:
    pass
```

дозвољава се кориснику да затвори прозор кад жели, без временског ограничења.

ПРОВЕРА ЗНАЊА

1. Наведи програмски код који покреће PyGame и отвара прозор произвољних димензија.
2. Наведи програмски код који освежава прозор и чека 5 секунди (5000 милисекунди) пре но што се прозор сам затвори и искључи.
3. У чему је разлика између дела програма који се искључује самостално после одређеног (задатог) времена и програма који корисник затвара у жељено време?
4. Где се налази почетак координатног система у програму PyGame?
5. Колико има параметара функције који служе за цртање дужи и шта они представљају?

3.4. Задавање боја

Кључне речи

боја, боја дужи, називи боја, задавање боја коришћењем модела RGB.

Подсети се:

- које су основне боје и како се комбинују;
- како се задаје боја оловке, а како боја позадине у Пајтону;
- важности редоследа наведених боја.

Сви облици могу се цртати у различитим бојама. **Боју** можеш добити на основу имена које се наводи као параметар функције `pg.Color`. Можеш користити већ готове боје навођењем одговарајуће ниске: `'black'` за црну, `'white'` за белу и слично. Рекли смо већ да се ниске наводе између једноструких или двоструких наводника (што значи да се равноправно могу користити `'blue'` и `"blue"`).

На пример, ако позовеш функцију `pg.draw.line(prozor, pg.Color('blue'), (100, 100), (200, 200), 3)`, на прозору ће се приказати дуж плаве боје чија су темена тачке са координатама (100, 100) и (200, 200) дебљине 3 пиксела.

Имена основних боја које се користе у програмима

<code>pg.Color("red")</code>	црвена
<code>pg.Color("green")</code>	зелена
<code>pg.Color("blue")</code>	плава
<code>pg.Color("yellow")</code>	жута
<code>pg.Color("cyan")</code>	тиркизна
<code>pg.Color("magenta")</code>	циклама
<code>pg.Color("purple")</code>	љубичаста
<code>pg.Color("white")</code>	бела
<code>pg.Color("black")</code>	црна
<code>pg.Color("grey")</code>	сива
<code>pg.Color("orange")</code>	наранџаста
<code>pg.Color("pink")</code>	розе

Пример бојења позадине у наранџасту боју.

```
prozor.fill(pg.Color("orange"))
```

ПРИМЕР 1

Ради бољег упознавања и вежбе, пробај да замениш све понуђене боје уместо звездица.

```
import pygame as pg
pg.init()
pg.display.set_caption("Boje-nazivi")
(sirina, visina) = (500, 500)
prozor=pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("*****"))

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

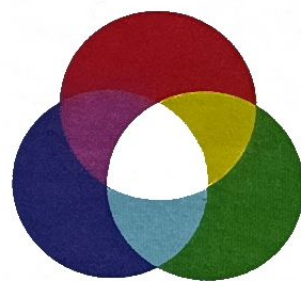
Основне боје могу се добити и коришћењем модела боје RGB (енг. *red-green-blue*) (сл. 3.12.). Свака боја може се добити комбиновањем црвене (*Red*), зелене (*Green*) и плаве (*Blue*) боје. На пример, комбиновањем црвеног и зеленог добија се жута, комбиновањем црвеног и плавог – љубичаста, а комбиновањем плавог и зеленог – тиркизна. Комбиновањем све три основне боје добија се бела, док се црна добија када се све боје искључе. Сива се добија када се измеша подједнака количина црвене, зелене и плаве боје.

У библиотеци PyGame боју можеш представити и тројком бројева, и то у облику трочлане уређене торке, нпр.: (100, 150, 200), или трочлане листе, нпр.: [100, 150, 200]. Торку или листу можеш навести директно као аргумент функције који одговара боји, а можеш је упамтити у променљивој и касније користити више пута.

На пример, доделом `TIRKIZNA=(0, 255, 255)` дефинишеш тиркизну боју наводећи одговарајуће количине црвене, зелене и плаве боје коју садржи (пошто је то мешавина плаве и зелене боје, у њој нема нимало црвене, а плава и зелена компонента су на максимуму). Након тога, ту боју можеш употребити и у позиву функције (нпр. `prozor.fill(TIRKIZNA)`).

RGB вредности неких карактеристичних боја

(255, 0, 0)	црвена
(0, 255, 0)	зелена
(0, 0, 255)	плава
(255, 255, 0)	жута
(0, 255, 255)	тиркизна
(255, 0, 255)	љубичаста
(255, 255, 255)	бела
(0, 0, 0)	црна
(128, 128, 128)	сива
(255, 128, 0)	наранџаста
(255, 128, 128)	розе



Сл. 3.12.
Комбинација боја

ПРИМЕР 2

Ради бољег увежбавања, пробај да замениш понуђене боје уместо звездица, а можеш и да се поиграш произвољним вредностима.

```
import pygame as pg
pg.init()
pg.display.set_caption("Боје - RGB")
(sirina, visina) = (500, 500)
prozor = pg.display.set_mode((sirina, visina))

prozor.fill([***, ***, ***])

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```



Отвори програм (Задатак 2 у претходној лекцији) који си већ урадио/урадила и само промени боје и преименуј програм.



ЗАДАТАК

1. У претходном поглављу имао/имала си задатак да напишеш слово Т. Преправи програм тако да исцртава слово Т зелене боје на жутој позадини.
2. Допуни програм који исцртава црно слово И на црвеној позадини.

```
import pygame as pg
pg.init()
pg.display.set_caption("Slovo I")
(sirina, visina) = (200, 200)
prozor = pg.display.set_mode((sirina, visina))

# bojimo pozadinu u crveno
****

# debljina linija je 10 piksela
debljina = 10

# leva crna linija
pg.draw.line(prozor, pg.Color("*****"), (***, ***), (***, ***), debljina)

# kosa crna linija
pg.draw.line(prozor, pg.Color("*****"), (50, 150), (150, 50), debljina)
```

```
# desna crna linija
pg.draw.line(prozor, pg.Color("*****"), (***, ***), (***, ***), debljina)

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

3. Напиши програм који исцртава бело слово М на плавој позадини. Прозор је 200 x 200 пиксела. Дужина вертикалних дужи је по 100 пиксела, а дебљина је 10 пиксела. Можеш користити координате дате за слово И.

4. Напиши програм који исцртава љубичасто слово Е на сивој позадини. Прозор је 150 x 200 пиксела. Дужина вертикалне дужи је 100 пиксела, хоризонталне горње и доње по 50 пиксела, средње 40 пиксела, а дебљина је 10 пиксела. Можеш користити координате дате за слово И.

ЗАПАМТИ

Боју можеш добити на основу имена које се наводи као параметар. Функција за задавање боја је `pg.Color`.

Ниске се наводе између једноструких или двоструких наводника.

Основне боје могу се добити и коришћењем модела боје RGB, и то тројком бројева у облику трочлане уређене торке или трочлане листе.

Функција `prozor.fill` омогућава бојење позадине, односно површине прозора.

ПРОВЕРА ЗНАЊА

1. Да ли се ниске које представљају име боје наводе између једноструких или двоструких наводника?
2. Како би на исправан начин написао/написала линију кода која исцртава дуж црвене боје коришћењем имена боје?
3. Која би била разлика уколико би претходни задатак био урађен коришћењем RGB модела?
4. Која се функција користи за бојење позадине?



Слово треба да изгледа као на слици 3.13. Треба водити рачуна о месту спајања горње и доње хоризонталне дужи са вертикалном дужи.



Сл. 3.13.

3.5.

Цртање квадрата и правоугаоника

Кључне речи

димензије и боја оквира правоугаоника/квадрата, димензије и боја целог (обојеног) правоугаоника/квадрата.

Подсети се:

- ▶ шта је квадрат;
- ▶ шта је правоугаоник;
- ▶ у чему је разлика између квадрата и правоугаоника;
- ▶ на који начин се обележавају странице квадрата, а на који начин странице правоугаоника.

За цртање квадрата и правоугаоника у библиотеци PyGame користимо функцију `pg.draw.rect`. Посматрајмо следећи пример.

```
pg.draw.rect(prozor, pg.Color("red"), (50, 50, 200, 100), 1)
```

Параметар функције `pg.draw.rect` јесте прозор на коме се црта (код нас ће то увек бити прозор настао позивом функције `pg.display.set_mode`).

Наредни параметар је боја коју можемо добити на основу њеног назива, функцијом `pg.Color`, или на основу тројке RGB вредности, како смо описали у поглављу о представљању боја.

Трећи параметар је уређена четворка бројева:

- прва два броја представљају координате горњег левог темена правоугаоника;
- друга два представљају редом његову ширину и висину.

Тако се у примеру прво црта црвени правоугаоник чије се горње лево теме налази на координатама (50, 50) (прва је координата x, која одређује удаљеност тог темена од леве ивице прозора, а друга је координата y, која одређује удаљеност тог темена од горње ивице прозора), чија је ширина 200 пиксела, а висина 100 пиксела.

Ова функција садржи и четврти параметар, број 1, који представља дебљину оквира правоугаоника у пикселима.

Ако се тај параметар не наведе или ако се наведе нула, тада се не црта оквир правоугаоника, већ се цео правоугаоник испуњава бојом.

```
pg.draw.rect(prozor, pg.Color("black"), (75, 75, 150, 50))
```

У овом случају црта се црни правоугаоник:

- горње лево теме налази се на координатама (75, 75);
- ширина је 150 пиксела;
- висина је 50 пиксела.

Овај правоугаоник цео је обојен црном бојом, пошто нема четвртог параметра (сл. 3.14.).



Сл. 3.14.

Целокупан код задатка који отвара прозор димензија 300 x 200 пиксела и изгледа као на слици 3.14. приказан је у наставку.

ПРИМЕР 1

```
import pygame as pg
pg.init()
pg.display.set_caption("Pravougaonik")
(sirina, visina) = (300, 200)
prozor = pg.display.set_mode((sirina, visina))
#bojimo pozadinu u belo
prozor.fill(pg.Color("white"))

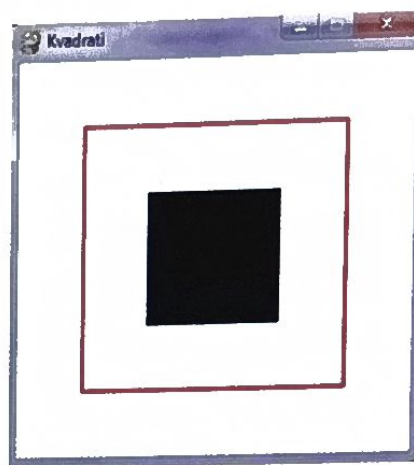
#crtamo crveni pravougaonik
pg.draw.rect(prozor, pg.Color("red"), (50, 50, 200, 100), 1)

#crtamo crni pravougaonik
pg.draw.rect(prozor, pg.Color("black"), (75, 75, 150, 50))

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

ЗАДАТАК

1. Допуни линије које недостају, уколико треба да нацрташ два квадрата (види слику 3.15.), при чему је црвена линија квадрата 200×200 пиксела, дебљине 3, а црни квадрат је 100×100 пиксела. Како би квадрати били видљивији, прозор ће бити димензија 300×300 пиксела. Води рачуна о почетним координатама црног квадрата.



Сл. 3.15.

```
import pygame as pg
pg.init()
pg.display.set_caption("Kvadrati")
(sirina, visina) = (300, 300)
prozor=pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))

#crtamo crveni kvadrat
*****

#crtamo crni kvadrat
*****

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

Пример за цртање квадрата и примену боја можеш видети на следећем урађеном примеру, који ће исцртати 5 квадрата у 5 боја. На почетку се дефинишу боје које ће се користити на основу RGB вредности. Након тога, прелази се на цртање једног по једног квадрата. Сви квадрати су димензија 50×50 пиксела. Стога је прави прозор чија је димензија 250×50 пиксела. Координата сваког горњег левог теме на квадрата је 0 (пошто се та темена налазе на горњој ивици екрана). Координата x сваког наредног темена је 50 пиксела већа од претходне. Дакле, горња лева темена тих пет квадрата редом имају координате (0, 0) (50, 0) (100, 0) (150, 0) и (200, 0).

ПРИМЕР 2

```
import pygame as pg
pg.init()

pg.display.set_caption("Pet kvadrata u pet boja")
(sirina, visina) = (250, 50)

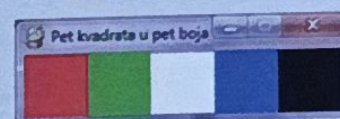
prozor = pg.display.set_mode((sirina, visina))

# bojimo pozadinu u belo
prozor.fill(pg.Color("white"))

# definišemo 5 osnovnih boja pomocu sistema RGB
CRVENA = (255, 0, 0)
ZELENA = (0, 255, 0)
BELA = (255, 255, 255)
PLAVA = (0, 0, 255)
CRNA = (0, 0, 0)

# crtamo 5 kvadrata
pg.draw.rect(prozor, CRVENA, (0, 0, 50, 50))
pg.draw.rect(prozor, ZELENA, (50, 0, 50, 50))
pg.draw.rect(prozor, BELA, (100, 0, 50, 50))
pg.draw.rect(prozor, PLAVA, (150, 0, 50, 50))
pg.draw.rect(prozor, CRNA, (200, 0, 50, 50))

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```



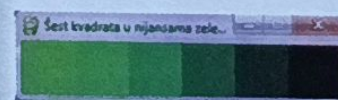
Сл. 3.16.

ЗАДАТАК

2. По узору на претходни пример, допуни програм који црта шест квадрата обојених у различите нијансе зелене боје (све дефинисати помоћу RGB система).

Помоћ: Нијансе зелене боје одређене су тиме да садрже само зелену компоненту боје, док су црвена и плава на нули. Боје иду од чисте зелене ([0, 255, 0]), па до црне ([0, 0, 0]), као што је приказано на слици 3.17.

Претпоставићемо да је разлика између сваке две суседне нијансе иста. Та разлика може се добити на више начина, а један од тих начина је $255 / 5 = 51$. Дакле, боје су редом одређене смањењем за 51 друге вредности, односно са [0, 255, 0], [0, 204, 0], [0, 153, 0], [0, 102, 0], [0, 51, 0] и [0, 0, 0]. Опет претпостављамо да су димензије квадрата 50 x 50 пиксела, тако да квадрате редом распоређујемо дуж прозора димензија 300 x 50 пиксела.



Сл. 3.17.

```

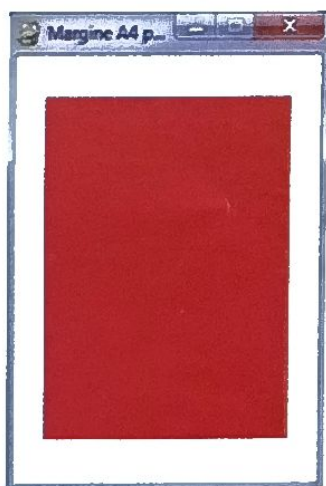
import pygame as pg
pg.init()

pg.display.set_caption("Šest kvadrata u nijansama zelene boje")
(sirina, visina) = (300, 50)
prozor = pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))

# crtamo 6 kvadrata
pg.draw.rect(prozor, [0, ***, 0], (***, 0, 50, 50))
pg.draw.rect(prozor, [0, ***, 0], (***, 0, 50, 50))
pg.draw.rect(prozor, [0, ***, 0], (***, 0, 50, 50))
pg.draw.rect(prozor, [0, ***, 0], (***, 0, 50, 50))
pg.draw.rect(prozor, [0, ***, 0], (***, 0, 50, 50))
pg.draw.rect(prozor, [0, 0, 0], (***, 0, 50, 50))

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()

```



Сл. 3.18.

3. Допуни програм који црвеном бојом обележава садржај папира увучен за маргине, као на слици 3.18. Наиме, папир формата A4 има димензије 210 x 297 mm. На његовој средини штампа се текст, при чему је са све четири стране оставља маргина која је 10 процената димензија папира (лева и десна маргина су 10 процената ширине, а горња и доња маргина су 10 процената висине папира). Програм треба да ради исправно и када се димензије папира, односно прозора, промене. Један милиметар цртати као један пиксел.

Помоћ: Ако проценат који заузима свака маргина обележимо са p (у нашем случају $p = 10$), ширина леве и десне маргине могу се израчунати као p пута ширина папира, а висина горње и доње маргине као p пута висина папира. Координата x горњег левог темена правоугаоника који представља садржај једнака је ширини леве маргине, а координата y висини горње маргине. Ширина садржаја једнака је разлици ширине странице и збира ширина леве и десне маргине, тј. може се израчунати као $1 - 2p$ x ширина папира. Слично се израчунава и висина папира.

У складу са претходним објашњењем, допуни наредни пример.

```

import pygame as pg
pg.init()
pg.display.set_caption("Margine A4 papira")
# uključujemo prozor dimenzije 210x297 piksela
(sirina, visina) = (210, 297)
prozor = pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))

# crtamo sadrzaj stranice
procenat_margine = 0.1

```

```

pocetak_sadrzaja_x = *****
pocetak_sadrzaja_y = *****
sirina_sadrzaja = *****
visina_sadrzaja = *****
pg.draw.rect(prozor, pg.Color("red"),
              (pocetak_sadrzaja_x, pocetak_sadrzaja_y, sirina_sadrzaja,
               visina_sadrzaja))

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()

```

ЗАПАМТИ

За цртање квадрата и правоугаоника у библиотеци PyGame користи се функција `pg.draw.rect`. Функција `pg.draw.rect` иза наведене функције садржи 3 елемента, у случају да се боји цела површина, односно 4 елемента, уколико се црта само оквир.

Функцијом се:

- позива прозор за цртање (нпр. `pg.draw.rect (prozor)`);
- позива боја (нпр. `pg.Color("red")`);
- одређују димензије квадрата или правоугаоника (нпр. ако напишемо (50, 50, 200, 100), прва два броја су координате горњег и левог темена, а друга два броја су ширина и висина правоугаоника; код цртања квадрата ширина и висина имају исте вредности);
- одређује дебљина оквира у пикселима (нпр. 1), а уколико овај елемент не постоји, обојиће се цела површина правоугаоника или квадрата).

ПРОВЕРА ЗНАЊА

1. Наведи функцију за цртање квадрата и правоугаоника у библиотеци PyGame.
2. Шта се добија ако иза наведене функције `pg.draw.rect` постоје 4 елемента?
3. Уколико се у неком примеру изостави последњи елемент, шта се добија?
4. Која је разлика између цртања квадрата и правоугаоника?

3.6.

Цртање троуглова и многоуглова

Кључне речи

троуглови, многоуглови, полигон.

Подсети се:

- ▶ шта је троугао;
- ▶ какве врсте троуглова постоје;
- ▶ шта се подразумева под појмом „многоугао”.

Цртање троуглова изводљиво је на више начина, а један од најчешће коришћених начина је задати координате сва три темена, при чему се препознаје целина (полигон). Погледај пример жутог троугла са сивим оквиром (сл. 3.19.).

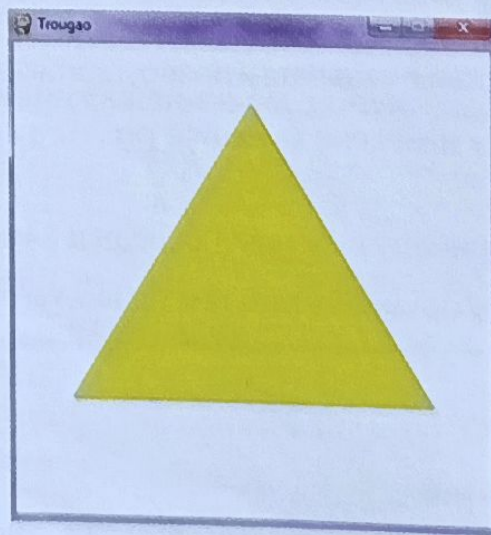
ПРИМЕР 1

```
# bojimo pozadinu prozora u belo
prozor.fill(pg.Color("white"))

# temena trougla
temena = [(50, 300), (200, 50), (350, 300)]

# crtamo trougao popunjen žutom bojom
pg.draw.polygon(prozor, pg.Color("yellow"), temena)

# crtamo sivi okvir oko trougla
pg.draw.polygon(prozor, pg.Color("grey"), temena, 3)
```



Сл. 3.19.

Многоугао је део равни који је ограничен затвореном изломљеном кривом линијом. Библиотека PyGame даје могућност за директно цртање многоуглова (каже се и полигона) (сл. 3.20.). То је приказано на следећем примеру.

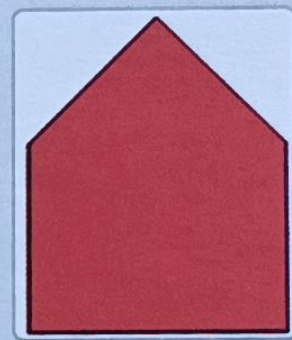
ПРИМЕР 2

```
# bojimo pozadinu ekrana u belo
prozor.fill(pg.Color("white"))

# temena mnogougla
temena = [(50, 300), (50, 150), (150, 50), (250, 150), (250, 300)]

# crtamo mnogougao popunjen crvenom bojom
pg.draw.polygon(prozor, pg.Color("red"), temena)

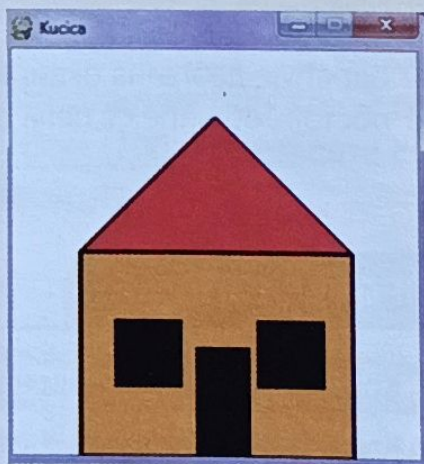
# crtamo crni okvir oko mnogougla
pg.draw.polygon(prozor, pg.Color("black"), temena, 3)
```



Сл. 3.20.

ЗАДАТАК

1. Напиши програм којим се црта кућица приказана као на слици 3.21. у прозору димензија 300 x 300 пиксела, користећи горенаведене линије програмског кода и самостално рачунајући темена и величине прозора и врата



Сл. 3.21.

2. Напиши програм којим се црта јелка приказана као на слици 3.22. у прозору димензија 300 x 300 пиксела, користећи наведене линије програмског кода.

```
# bojimo pozadinu u belo
prozor.fill(pg.Color("white"))

# boje koje cemo koristiti
ZELENA = (0, 100, 36)
BRAON = (97, 26, 9)
```



Сл. 3.22.

```
# stablo
pg.draw.rect(prozor, BRAON, (130, 250, ???, ???))

# krošnja
pg.draw.polygon(prozor, ZELENA, [(50, 250), (150, 150), (250, 250)])
pg.draw.polygon(prozor, ZELENA, [(75, ???), (150, 100), (???, 200)])
pg.draw.polygon(prozor, ???, [(100, 150), (???, ???), (200, ???)])
pg.draw.polygon(prozor, ???, [(???, 100), (???, 50), (175, ???)])
```

ЗАПАМТИ

За цртање полигона (троуглова, многоуглова) у библиотеци PyGame користи се функција `pg.draw.polygon`. Функција `pg.draw.polygon` садржи 3 елемента, у случају да се облик боји у потпуности, или 4, уколико се црта само оквир датог облика.

Функцијом се:

- позива прозор за цртање (нпр. `pg.draw.polygon(prozor)`);
- позива боја (нпр. `pg.Color("red")`), али се боја може задати и помоћу RGB вредности;
- задају темена као уређени парови *x* и *y* осе (сваки уређени пар представља једно теме многоугла (полигона) [нпр. `temena`]);
- одређује дебљина оквира у пикселима (нпр. 3), а уколико овај елемент не постоји, обојиће се цела површина полигона).

ПРОВЕРА ЗНАЊА

1. Који је најједноставнији начин за цртање троуглова?
2. Како се задају темена многоуглова?
3. Која је разлика између цртања оквира многоуглова и обојеног многоугла?

Цртање кругова и кружница

3.7.

круг, кружница.

Кључне речи

Подсети се:

- ▶ шта је круг, а шта кружница;
- ▶ где се налази центар круга/кружнице;
- ▶ шта је полупречник, а шта пречник круга/кружнице;
- ▶ шта представља реално, а шта целобројно дељење;
- ▶ која је функција за заокруживање бројева.

Библиотека PyGame омогућава једноставно цртање кругова коришћењем функције `pg.draw.circle`. Цртање обојеног круга омогућено је коришћењем наредбе:

```
pg.draw.circle(prozor, pg.Color("red"), (100, 100), 50).
```

Параметри функције који су задати након `pg.draw.circle` уносе се у заграду и означавају прозор на коме се црта, затим боју којом се црта (у нашем случају црвена); уређеним паром (100, 100) задате су координате центра круга, а 50 је полупречник круга.

ПРИМЕР 1

Пример кода програма у коме је прозор димензија 200 x 200 пиксела, а круг црвене боје је у центру прозора са полупречником од 50 пиксела, приказан је у наставку.

```
import pygame as pg
pg.init()

pg.display.set_caption("Krug crvene boje")
(sirina, visina) = (200, 200)
prozor = pg.display.set_mode((sirina, visina))

# bojimo pozadinu u belo
prozor.fill(pg.Color("white"))

# crtamo crveni krug
pg.draw.circle(prozor, pg.Color("red"), (100, 100), 50)

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```



Приликом цртања кружнице, користи се иста функција `pg.draw.circle`. Разлика између цртања круга и кружнице јесте у још једном додатом параметру којим се задаје дебљина кружнице у пикселима.

Цртање кружнице дебљине 3 пиксела, плаве боје, омогућено је коришћењем наредбе:

```
pg.draw.circle(prozor, pg.Color("blue"), (100, 100), 50, 3).
```

Прилагодићемо раније коришћени програм, са малим изменама. Прозор је истих димензија (200 x 200 пиксела), а кружница је плаве боје, са полупречником 50 пиксела у центру прозора. Кружница је дебљине 3 пиксела. Програм је у целини исписан у наставку.

ПРИМЕР 2



```
import pygame as pg
pg.init()

pg.display.set_caption("Plava kružnica")
(sirina, visina) = (200, 200)
prozor = pg.display.set_mode((sirina, visina))
prozor.fill(pg.Color("white"))

# crtamo plavu kružnicu
pg.draw.circle(prozor, pg.Color("blue"), (100, 100), 50, 3)

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

За разлику од функција за цртање линија и правоугаоника, функција за цртање кругова захтева да су координате центра круга и дужина полупречника цели бројеви. Зато се често у задацима у којима се цртају кругови уместо реалног дељења (оператора `/`) користи цело-бројно дељење (оператор `//`) или заокруживање реалних бројева (функцијом `round` или функцијом `int`).

Када видиш поруку `TypeError: integer argument expected, got float`, тада знај да је проблем у томе што је функцији за цртање круга уместо целог прослеђен неки реалан број. Тај проблем можеш лако решити коришћењем неког облика заокруживања бројева.

ЗАДАТАК

1. Напиши програм који исцртава 4 круга чији је центар у центру прозора. Прозор треба да је димензија 250 x 250 пиксела беле боје. Највећи круг треба да је црн и да има полупречник 125 пиксела, као на слици 3.23.

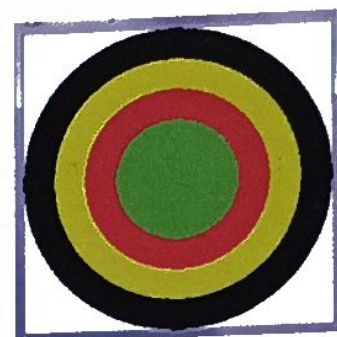
```
prozor.fill(pg.Color("white")) # bojimo pozadinu u belo
(cx,cy)=(sirina//2,visina//2) # centar krugova je u sredini ekrana
pg.draw.circle(prozor,pg.Color("black"), (cx, cy), 125)
```

Следећи круг треба да буде жут и да има полупречник 100 пиксела.

```
pg.draw.circle(prozor, pg.Color("yellow"), (cx, cy), 100)
```

Затим следе два круга, један је црвене боје полупречника 75 пиксела, а други зелене боје полупречника 50 пиксела.

```
pg.draw.circle(prozor,pg.Color("red"),(cx,cy),75)
pg.draw.circle(prozor,pg.Color("green"),(cx,cy),50)
```



Сл. 3.23.

2. Допуни програм којим се исцртава Чича Глиша као на слици 3.24.

Приликом решавања задатка, пажљиво треба одредити координате свих кругова. Кругови треба да буду испуњени одређеном бојом и затим уоквирени танком црном кружницом. То ћеш постићи тако што ћеш два пута позвати функцију `pg.draw.circle`, први пут без прослеђеног петог параметра који одређује дебљину кружне линије (чиме ћеш постићи да се црта испуњен круг), а други пут са прослеђеним параметром дебљине (чиме ћеш постићи да се нацрта кружница).

Да би избегао/избегла дуплирање кода, дефинисаћеш и помоћну функцију `uokviren_krug` коју ћеш позивати приликом цртања Чича Глише. Параметре ћеш наводити на исти начин као код функције `pg.draw.circle`.

```
import pygame as pg
pg.init()
pg.display.set_caption("Čiča Gliša")
(sirina, visina) = (300, 300)
prozor = pg.display.set_mode((sirina, visina))
# bojimo pozadinu prozora u belo
prozor.fill(pg.Color("white"))

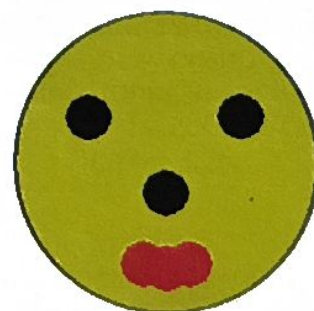
# procedura kojom se crta obojen krug okruzen crnom kruznicom
def uokviren_krug(prozor, boja, centar, poluprecnik):
    # crtamo ispunjen krug
    pg.draw.circle(prozor, boja, centar, poluprecnik)
    # uokvirujemo ga crnom kruznicom
    *****

# crtamo glavu
uokviren_krug(prozor, pg.Color("yellow"), (150, 150), 100)

# crtamo levo oko
uokviren_krug(prozor, pg.Color("black"), (100, 120), 15)
# crtamo desno oko
*****
```



Треба истаћи је да је веома важан редослед цртања кружница, јер се сваки нови круг „лепи“ преко претходног. Значи, увек треба кренути од највеће, па до најмање кружнице.



Сл. 3.24.

```
# crtamo nos
uokviren_krug (prozor, pg.Color("black"), (150, 170), 15)

# crtamo usta sastavljena od 3 kruga
*****
*****
pg.draw.circle(prozor, pg.Color("red"), (150, 220), 15)

pg.display.update()
while pg.event.wait().type != pg.QUIT:
    pass
pg.quit()
```

ЗАПАМТИ

За цртање круга у библиотеци PyGame користи се функција `pg.draw.circle`. Функција `pg.draw.circle` садржи 4 елемента у случају да се црта обојени круг, или 5 елемената, уколико се црта само кружница.

Функцијом се:

- позива прозор за цртање (`pg.draw.circle (prozor)`);
- позива боја (нпр. `pg.Color ("blue")`);
- одређују координате центра круга (нпр. `(100, 100)`);
- одређује полупречник круга у пикселима (нпр. 50);
- одређује дебљина оквира у пикселима (нпр. 3), а ако овај елемент не постоји, обојиће се цела површина круга.

За цртање кругова, у задацима треба користити целобројно дељење или заокруживање реалних бројева, пошто се захтева да су координате центра круга и дужина полупречника цели бројеви.

Редослед цртања кругова је важан, пошто се сваки нови круг „лепи“ преко претходног.

ПРОВЕРА ЗНАЊА

1. Наведи функцију за цртање круга у библиотеци PyGame.
2. Колико елемената садржи функција за цртање круга уколико се црта обојен круг, а колико уколико се црта кружница?
3. Да ли је важан редослед цртања кругова? Објасни.
4. Зашто се код цртања кругова користи целобројно дељење или заокруживање реалних бројева?