Корњача графика

Веома леп начин да се прикажу неки основни концепти програмирања је такозвана корњача-графика у којој се неки замишљени лик креће по екрану остављајући притом траг који чини цртеж. Уколико си програмирао у Скречу, онда си већ имао прилику да црташ коришћењем корњача графике. Тема ове лекције је упознавање са корњача графиком у Python програмском окружењу.

За почетак одгледај следећу видео-лекцију

Python – корњача графика

Да бисмо у нашим програмима могли користити цртање уз помоћ корњаче, потребно је прво да укључимо библиотеку за рад са корњачом под називом `turtle.` Укључивање библиотеке се изводи тако што се пре позива било које наредбе из бибилотеке наводи **`import turtle`**. Након тога можемо издавати наредбе нашој корњачи.

Свака наредба корњача бибилотеке почиње са **`turtle.`** На пример, наредба којом се корњачи говори да се помери напред (у смеру у ком је тренутно окренута) 100 корака гласи овако **`turtle.forward(100)`**.

Цртање корњача графиком се заснива на идеји да корњача, којој можемо издато команде за кретање на екрану, подразумевано са собом носи оловку којом црта. Оловка може бити спуштена, што значи да ће на екрану остављати траг којим исцртава путању којом се корњача креће. Уколико је оловка подигнута, не оставља траг. На почетку програм корњача се налази у центру екрана и окренута је надесно (у смеру истока), па од те тачке почиње свако њено кретање.

За почетак испробај следећи једноставан програм.

```
1 import turtle
2 turtle.forward(100)
```



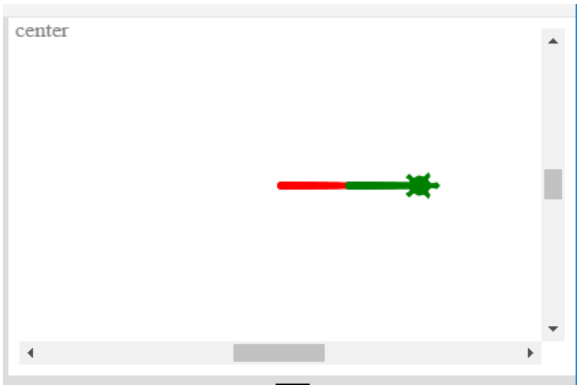
Као што видиш резултат програма је исцртавање једне линије дужине 100 пиксела, почевши од центра поља у којем се врши исцртавање ка истоку.

Могуће је поставити различите параметре који одређују начин на који се врши исцртавање. Да бисмо видели нашу корњачу (уместо стрелице која се подразумевано приказује) можемо употребити наредбу **`turtle.shape("turtle")`**. Помоћу **`turtle.color`** можемо променити боју корњаче и њене оловке (и тако променити и боју трага тј. линија које се цртају). Као параметар ове наредбе у заградама под наводницима наводимо име жељене боје на енглеском језику. На пример, наредба **`turtle.color("red")`** поставља црвену боју. Помоћу **`turtle.width`** постављамо дебљину трага који корњача оставља, при чему се дебљина задаје као параметар. На пример, наредбом **`turtle.width(5)`** постижемо да линије које се цртају буду дебеле 5 пиксела. Да бисмо видели јасније како корњача ради, можемо јој променити брзину кретања. **`turtle.speed(0)`** нам даје најбржу корњачу (након покретања програма се одмах види готов цртеж), док turtle.speed(10) даје најспорију корњачу (могуће је навести и било коју целобројну вредност између 0 и 10).

Да ли можеш да предвидиш шта ће се добити покретањем наредног програма?

```
1  import turtle
2  turtle.speed(5)
3  turtle.shape("turtle")
4  turtle.width(5)
5  turtle.color("red")
6  turtle.forward(50)
7  turtle.color("green")
8  turtle.forward(50)
```

Да ли се твоје предвиђање разликује од резултата који можеш видети на следећој слици?



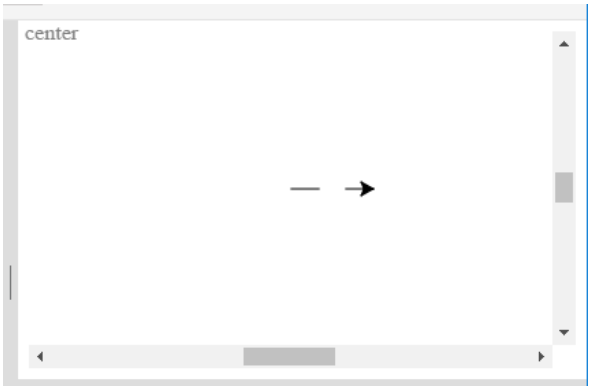
Уколико ниси био у праву, проанализирај са наставником и осталим ученицима где си погрешио у закључивању.

Корњача током свог кретања оставља траг. Међутим, некада је згодно да корњачу померимо без цртања. Наредбом **`turtle.penup()`** корњача подиже своју оловку и након тога се креће по екрану не остављајући траг све док јој се не изда наредба **`turtle.pendown()`** након чега поново почиње да оставља траг током кретања.

Можеш ли да предвидиш шта ће се исцртати наредним програмом?

```
1  import turtle
2  turtle.speed(10)
3  turtle.forward(20)
4  turtle.penup()
5  turtle.forward(20)
6  turtle.pendown()
7  turtle.forward(20)
```

Своје предвиђање упореди са резултатом извршавања датог програма приказаном на слици испод.



Као што видиш, корњача се три пута кретала напред у правцу истока, али је при другом померању оловка била подигнута, па није остављала траг. Зато је резултат испрекидана линија.

Уколико те занима да научиш још неку корњача команду можеш да их потражиш у интерактивном уџбенику.

Понављање

У програмима које задајемо корњачи често су облици правилни и неке се наредбе понављају. На пример у задатку који следи одређени кораци у извршавању програма су исти, тј. понављају се.

Задатак 1.
Напиши програм у којем корњача црта квадрат чија је дужина страница 100 корака.

Предлог решења

Квадрат се може нацртати тако што се четири пута корњачи зада да иде 100 корака напред и да се затим окрене за 90 степени (на пример, налево). Програм који исцртава квадрат се може написати овако:

```
1  import turtle
2  turtle.forward(100)   # иди напред 100 корака
3  turtle.left(90)       # окрени се 90 степени налево
4  turtle.forward(100)   # иди напред 100 корака
5  turtle.left(90)       # окрени се 90 степени налево
6  turtle.forward(100)   # иди напред 100 корака
7  turtle.left(90)       # окрени се 90 степени налево
8  turtle.forward(100)   # иди напред 100 корака
9  turtle.left(90)       # окрени се 90 степени налево
```

Претходни програм можемо да скратимо ако уведемо наредбу којом постижемо да се неки задати низ наредби више пута понови. У језику Python најлакши начин да се то уради је наредба **`for`**. О њој ћемо много детаљније говорити касније.

За сада само упамти да наредбу
ponovi n puta можеш записати као **`for _ in range(n):`**
Наредбе које се понављају наводиш увучене неколико размака у односу на положај наредбе **`for`**.

Погледај како претходни пример изгледа уз коришћење наредбе понављања.

```
1  import turtle
2  for _ in range(4):      # ponovi 4 puta:
3    turtle.forward(100)   #   idi napred 100 koraka
4    turtle.left(90)       #   okreni se nalevo za 90 stepeni
```

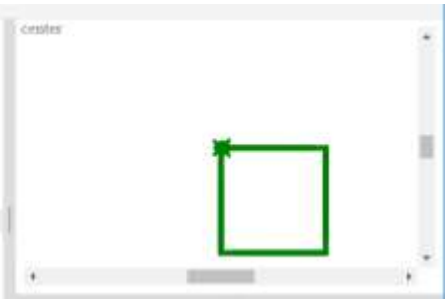
Задатак 2.
Уз коришћење наредбе понављања, напиши програм у којем корњача црта квадрат чија је дужина страница 100 корака, контура зелене боје и дебљине 5, а сама корњача оставља отисак корњаче.

Да напоменемо, корњача показује отисак корњаче, а не стрелице, након позива команде **`turtle.shape("turtle")`**. Опис команде можеш пронаћи у интерактивном уџбенику.

Предлог решења

Решење и резултат покретања програма могу изгледати овако:

```
1  import turtle
2  turtle.shape("turtle")
3  turtle.width(5)
4  turtle.color("green")
5  for _ in range(4):      # ponovi 4 puta:
6    turtle.forward(100)   #   idi napred 100 koraka
7    turtle.right(90)      #   okreni se nalevo za 90 stepeni
```



Обрати пажњу још једном на следеће на чињеницу да све наредбе које се понављају у оквиру једне for наредбе мораш записати увучене неколико размака у односу на положај наредбе for. Погледај какав резултат даје претходни програм са малом изменом у којој није наредба у 7. линији програма увучена.

```
1  import turtle
2  turtle.shape("turtle")
3  turtle.width(5)
4  turtle.color("green")
5  for _ in range(4):      # ponovi 4 puta:
6    turtle.forward(100)   #   idi napred 100 koraka
7  turtle.right(90)        #   okreni se nalevo za 90 stepeni
```



Да ли можеш да закључиш шта се десило па је резултат исцртавања линија, а не квадрат?

Линија је добијена тако што је корњача четири пута направила корак дужине 100 у истом смеру.

Дакле, неправилно назубљивање кода је дало нежељен резултат.

Задатак 3.
Са и без коришћење наредбе понављања, напиши програм у којем корњача црта једнакостранични троугао са страницом дужине 100.

Покушај да решиш овај задатак. Задатак можеш пронаћи и проверити сопствено решење у окружењу интерактивног уџбеника.