Процедурално програмирање

У досадашњем изучавању програмирања у Пајтону, користили су се програми креирани помоћу функција (процедурално програмирање).
У процедуралном програмирању се користе објекти, најчешће предефинисаних типова података.
Објекти су структурне јединице у којима је дефинисан одређени тип података.
Такође, објекти могу дефинисати и скуп метода које се могу применити над тим објектима.
Други начин организовања програма је комбиновање података и метода унутар објеката, чиме се креирају нови апстрактни типови података.
Овакво програмирање се назива објектно орјентисано програмирање (ООП).

Дефиниција објекта у програмирању

Објекат је материјална ствар која се може физички осетити, којом се може манипулисати и која може изазвати одређене ефекте.
Дефиниција објекта у развоју софтвера (software development) није превише различита од овакве дефиниције.
Софтверски објекти су модели нечега што може извести неке радње, над којима се могу извести неке радње и који на тај начин изазивају одређене ефекте.
Објекат се састоји од скупа података (data) и придружених понашања (behaviors).
Објектно орјентисано значи функционалност усмерена према моделовању објеката.
ООП је само једна од многих техника које се користе за моделовање комплексних система преко описа великог броја међусобно интерагујућих објеката, употребом њихових података и понашања.

Однос између објекта и класе

Објекти се међусобно могу разликовати по различитим понашањима.
Ако је могуће по неким сличностима груписати објекте у посебну врсту објеката, добија се класа (class).
Објекти су инстанце класа.

Основни појмови ООП

ООП се заснива на четири појма: енкапсулација, апстракција, наслеђивање и полиморфизам.
Објекти обично имају интеракцију између себе.
Врло често други објекти немају потребе да приступају унутрашњим активностима одређеног објекта.
Процес сакривања детаља унутрашњег функционисања одређеног објекта је **енкапсулација** (у Пајтону енкапсулација није значајније заступљена).
Апстракција је процес креирања и употребе начина за представљање особина и понашања објеката преко употребе унутрашњих детаља одређеног објекта.
Пример два различита нивоа апстракције за класу Ауто: возач интерагује са аутом преко употребе кочница и папучице за гас; ауто-механичар ради на другом нивоу апстракције са истом класом Ауто подешавајући кочнице и сам мотор.
У ООП класа (подкласа) може **наследити** особине и понашања из друге класе (надкласа) што је налик породичном стаблу код људи.
Полиморфизам је могућност третирања класе на другачији начин у зависности од тога која се подкласа имплементира.

Класе

Ако је могуће по неким сличностима груписати објекте у посебну врсту објеката, добија се класа (class).
Класа је програмерски код којим се групишу подаци и понашања (методе) одређеног типа објекта.
Класа је шема или опис по којем се могу креирати објекти.
Креирани објекат који је направљен према датој класи је инстанца објекта описаног у класи.
Могуће је направити више истих инстанци по датом истом опису.
Свака од оваквих инстанци је посебна инстанца истог објекта направљеног према једној класи.
Пример: направити најједноставнију класу у Пајтону

```
class MojaPrvaKlasa:
    pass
```

Дефиниција класе

Дефиниција класе се састоји од линије хедера (заглавља) класе и садржаја (тело) класе.
Линија хедера класе започиње са резервисаном речи class, иза ње је име класе којим се идентификује класа а на крају је двотачка.
Име класе мора да одговара Пајтон стандарду за давање имена промењивима.
Препорука је давање имена класе у нотацији камиле (Camel Case) чиме се име започиње са великим словом а свака следећа реч у имену такође почиње великим словом, без знака између речи.
Као и усвим осталим структурама у Пајтону, увлачење се користи за ограничавање садржаја класе.
У датом примеру, у телу класе се налази само службена реч pass.
Ова службена реч указује да не постоји никакав други садржај у класи.
Овакав једноставан пример не пружа ништа посебно, али и код овакве класе је могуће креирати инстанце објеката изведене према тој класи.
Пример:

```
class MojaPrvaKlasa:
    pass

a = MojaPrvaKlasa()
b = MojaPrvaKlasa()
print(a)
print(b)
```

На излазу се добија:

```
<__main__.MojaPrvaKlasa object at 0x000001A9587F9B70>
<__main__.MojaPrvaKlasa object at 0x000001A9588005F8>
```

У примеру се инстанцирају два објекта из класе MojaPrvaKlasa, а то су објекти a и b.
Креирање инстанце класе се реализује куцањем имена класе и пара заграда, што личи на позивање функције.
Пајтон зна да је у питању рад са класом и да је тражено да се креира нов објекат.
Приликом приказа инстанци на екрану, оба објекта дају информације о томе којој класи припадају и која је њихова почетна адреса што указује да су заиста у питању два различита објекта.

Задатак 029: креирати класу Ptica и проверити да ли је тиме креиран и објекат класе

```
class Ptica:
    pass

print(Ptica)
```

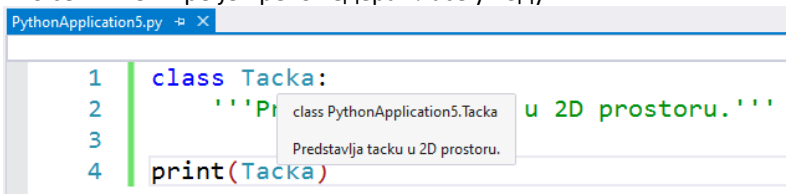
Даје: <class '__main__.Ptica'>

Задатак 030: креирати класу Tacka, описати докстрингом смисао класе

```
class Tacka:
    '''Predstavlja tacku u 2D prostoru.'''

print(Tacka)
```

Ако се мишем пређе преко хедера класе у коду:



Задатак 031: креирати класу Tacka па инстанцирати класу Tacka објектом tackica (да објекат tackica постане инстанца класе Tacka)

```
class Tacka:
    '''Predstavlja tacku u 2D prostoru.'''

print(Tacka)                #<class '__main__.Tacka'>
tackica = Tacka()
print(tackica)              #<__main__.Tacka object at 0x000001C0ACF48438>
```

Задатак 032: креирати класу Tacka па инстанцирати класу Tacka објектима tacka_1 и tacka_2 са и проверити да ли заиста представљају два различита објекта у меморији рачунара

```
class Tacka:
    '''Predstavlja tacku u 2D prostoru.'''

print(Tacka)                #<class '__main__.Tacka'>
tacka_1 = Tacka()
tacka_2 = Tacka()
print(tacka_1)              #<__main__.Tacka object at 0x000002825CF58470>
print(tacka_2)              #<__main__.Tacka object at 0x000002825CF58358>
```

Провежбати следеће задатке :

25. Креирати класу NebeskoTelo и проверити да ли је тиме креиран и објекат класе.

26. Креирати класу Vozilo, у телу класе унети докстринг описа класе па инстанцирати класу Vozilo објектом auto.

27. Креирати класу NebeskoTelo, у телу класе унети докстринг описа класе па инстанцирати класу NebeskoTelo објектима planeta и zvezda и проверити да ли заиста представљају два различита објекта у меморији рачунара.

28. Креирати класу Tacka и класу Linija, где се обе користе као део 2Д простора. Инстанцирати класу Tacka објектом tacka_1 и класу Linija објектом linija_1 и проверити да ли заиста представљају два различита објекта у меморији рачунара.

29. Креирати два објекта над класом Biblioteka па проверити да ли заиста представљају два различита објекта у меморији рачунара.

30. Инстанцирати класу Zivotinja са три објекта.