Иницијализација објеката

Пример:

```
class Tacka:
    def pomeranje(self, x, y):
        self.x = x
        self.y = y

tacka = Tacka()
tacka.x = 5
print(tacka.x)
print(tacka.y)
```

Ако се није експлицитно поставила позиција објекта класе Таска, креира се тачка у простору без правилне позиције.
Покушај приступа непостојећем атрибуту ће изазвати:

```
Traceback (most recent call last):
File "<stdin>", line 9, in <module>
AttributeError: 'Tacka' object has no attribute 'y
```

Види се да би све било у реду да се поставила некаква дифолт вредност за атрибут у.
Највећи број ООП програмских језика има концепт конструктора, посебне методе која креира и иницијализује објекат који се жели створити.
Пајтон има и конструктор и иницијализатор.
Функција конструктора се ретко користи осим ако се не ради нешто потпуно нестандардно, док се метода иницијализације увек користи.
Метод иницијализације је исти као и сваки други метод, осим што има посебно име: __init__.
Две ундерскор црте на почетку и на крају имена указују да је то посебан метод који Пајтон интерпретер посматра као посебан случај.
Савет је да се никада не користе ундерскор црте у називу корисничких метода јер се може десити да Пајтон програмери додају функције са посебним улогама а које су баш тог имена.
Пример:

```
class Tacka:
    def __init__(self, x, y):      #pozvani su argumenti tacka, 3, 5
        self.pomeranje(x, y)
    def pomeranje(self, x, y):
        self.x = x
        self.y = y
    def resetovati(self):
        self.pomeranje(0, 0)
tacka = Tacka(3, 5)
print(tacka.x, tacka.y)
```

У примеру се користи иницијализација методе на класи Таска, која захтева да постоје прецизне x и y координате при инстанцијацији објекта класе.
Сада се не може десити да тачка постоји без y координате.

Метод конструктор

Пример:

```
class Tacka:
    def __init__(self, x = 0, y = 0):
        self.pomeranje(x, y)
```

Синтакса службених речи као аргумената се заснива на употреби оператора доделе после имена сваке промењиве.
Ако позивајући објекат нема аргумент, онда се користи дифолт параметар уместо њега.
Промењиве ће и даље бити доступне методи, али ће имати вредности које се предвиђене у листи параметара.
Најчешће се у методи __init__ смештају искази за иницијализацију.
По дифолту, у иницијализатор се смешта инстанца која се назива self, иако се може по жељи назвати и другачије.
Али, Пајтон има конструктор као додатак овој методи иницијализације.

Метод конструктор се назива __new__ и прихвата само један параметар, класу која се конструише (она се позива пре прављења објекта, тако да се не користи self параметар).
Као резултат је новокреирани објекат.
У Пајтону, конструктор се никада не користи па ће метод __init__ бити сасвим довољан.

Документовање кода
Често је неопходно писати документацију о сваком објекту и методи коришћењем докстрингова који су увучени као и остатак тела метода.
Ако се објашњење простире на више редова, користи се докстринг са три апострофа или наводника за ограничавање садржаја.
Докстринг сумира сврху класе и метода коју описује, треба да објасни параметре.
Пример:

```
class Tacka:
    'Predstavlja tacku u 2D kordinatnom sistemu'
    def __init__(self, x = 0, y = 0):
        '''Inicijalizuje poziciju nove tacke. Koordinate x i y
        se mogu predodrediti. Ako nisu predodredjene, tacka je
        postavljena po difoltu na pocetne koordinate.'''
        self.pomeranje(x, y)
    def pomeranje(self, x, y):
        "Pomera tacku na novu lokaciju u 2D prostoru."
        self.x = x
        self.y = y
    def resetovanje(self):
        'Resetuje tacku na koordinatni pocetak: 0, 0'
        self.pomeranje(0, 0)
```

Израда лабораторијских вежби:
Задатак 038: Креирати класу Papagaj са једним атрибутом класе vrsta. Инстанцирати класу са објектом poli и проверити да ли објекат има атрибут класе.

```
class Papagaj:
    vrsta = "ptica"    #atribut klase

poli = Papagaj()
print("Poli je", poli.vrsta)                      #pristup atributu klase
print("Poli je {}".format(poli.__class__.vrsta)) #pristup atributu klase
```

Даје:
Poli je ptica
Poli je ptica

Задатак 039: Креирати класу Papagaj са атрибутом класе vrsta и методом __init__ чији су атрибути инстанце име папагаја и година старости папагаја. Креирати објекат poli са одговарајућим аргументима. Проверити да ли објекат има приступ атрибуту класе и атрибутима инстанце.

```
class Papagaj:
    vrsta = "ptica"                              #atribut klase

    def __init__(self, ime, godine):             #atributi instance
        self.ime = ime
        self.godine = godine

poli = Papagaj("Poli", 44)
print("Poli je {}.".format(poli.__class__.vrsta)) #pristup atributu klase
print("Moj papagaj se zove", poli.ime, "i ima", poli.godine, "godine.")
print("Moja {} se zove {} i ima {} godine.".format(poli.vrsta, poli.ime, poli.godine))
```

Задатак 040: Креирати класу Tacka, иницијализатор са атрибутима инстанце x и y и методама pomeranje и resetovanje. Метода pomeranje има два аргумента, x и y, који као атрибути указују сами на себе. Метода resetovanje нема аргументе, већ позива преко саме себе методу pomeranje са вредностима 0, 0 као аргументима. Објекат tacka_1 има координате 3, 5 а затим се врши ресетовање њене позиције.

```
class Tacka:
    def __init__(self, x, y):
        self.pomeranje(x, y)

    def pomeranje(self, x, y):
        self.x = x
        self.y = y

    def resetovanje(self):
        self.pomeranje(0, 0)

tacka_1 = Tacka(3, 5)
print("Pocetak tacka_1: {}, {}".format(tacka_1.x, tacka_1.y))
tacka_1.resetovanje()
print("Kraj tacka_1: {}, {}".format(tacka_1.x, tacka_1.y))
```

Задатак 041: На основу задатка 002, креирати објекат tacka_2 који ће на почетку имати координате 0, 0, а затим добија координате објекта tacka_1 са почетка.

```
class Tacka:
    def __init__(self, x, y):
        self.pomeranje(x, y)

    def pomeranje(self, x, y):
        self.x = x
        self.y = y

    def resetovanje(self):
        self.pomeranje(0, 0)

tacka_2 = Tacka(0, 0)
print("Pocetak tacka_2: {}, {}".format(tacka_2.x, tacka_2.y))
tacka_1 = Tacka(3, 5)
print("Pocetak tacka_1: {}, {}".format(tacka_1.x, tacka_1.y))
tacka_2 = tacka_1
print("Kraj tacka_2: {}, {}".format(tacka_2.x, tacka_2.y))
```

Задаци за самосталан рад:

35. Креирати класу Papagaj са једним атрибутом класе vrsta. Инстанцирати класу са два објекта poli и kiki па проверити да ли оба објекта имају атрибут класе.

36. Креирати класу Papagaj са атрибутом класе vrsta и методом __init__ чији су атрибути инстанце име, година старости, боја перја и воће којим се храни. Креирати објекте poli и kiki са одговарајућим аргументима. Проверити да ли објекти имају приступ атрибуту класе и атрибутима инстанце.

37. Коришћењем дефиниције класе из задатка 040 написати програм који генерише случајне целе бројеве (у опсегу од -3 до 3) као координате три тачке, објеката из класе. Програм даје на излазу стринг који исписује колико генерисаних тачака се налази у координатном почетку 2Д простора.

38. Коришћењем дефиниције класе из задатка 040 написати програм који генерише случајне целе бројеве као координате две тачке, објеката из класе. Програм исписује растојање између тачака и координатног почетка 2Д простора.