ALGEBRA

PROGRAMIRANJE

Predavanje 09

Ishod 4

1

KLASE I OBJEKTI

ALGEBRA

2

Objektno orijentirano programiranje

- Do sada smo naše programe organizirali na proceduralni način (engl. *procedural programming*)
  - Podatke čuvamo u lokalnim i globalnim varijablama
  - Kreiramo i pozivamo funkcije (procedure) s parametrima i povratnim vrijednostima
- Objektno orijentirano programiranje (OOP) je pristup u kojem povezane podatke i funkcije čuvamo zajedno
  - Objekt sadrži i podatke (varijable, atribute, svojstva) i funkcije (metode, ponašanja, operacije)
  - Objekt nastaje na temelju klase
  - (Svu) manipulaciju podacima radimo preko funkcija na istom objektu

ALGEBRA

3

Primjer proceduralnog vs OOP pristupa

- Proceduralni pristup

```
def izracunaj_povrsinu(sirina, visina):
    p = sirina * visina
    print('Površina je:', p)

s = 10
v = 5
izracunaj_povrsinu(s, v)
```

- OOP pristup

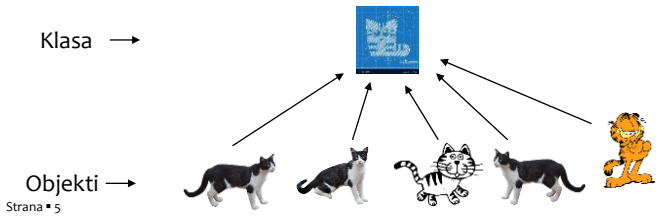
```
p = Pravokutnik()
p.sirina = 10
p.visina = 5
p.izracunaj_povrsinu()
```


ALGEBRA

4

Uvod

- Klasa je nacrt za objekte (koji se još nazivaju i instance)
  - Klasa definira svojstva (varijable) i ponašanja (metode)
  - Izradom klase Mačka opisujemo njena svojstva (ime, boja, godina rođenja) te ponašanja (mijauče, lovi, spava)
- Objekt je konkretna izvedba klase u memoriji
  - Na temelju klase Mačka možemo napraviti puno mačaka
  - Svaka mačka ima svoje vrijednosti svojstava



Slika klase uzeta s: https://www.vectorstock.com/royalty-free-vector/cats-elements-vector-6075826

ALGEBRA

5

Klase i objekti u Pythonu

- Za definiranje klase koristimo ključnu riječ **`class`**
  - Iza nje slijedi naziv klase pa dvotočka
- Za izradu objekta koristimo naziv klase iza kojeg slijede okrugle zagrade
  - Kao da pozivamo funkciju

```
class MojaPrvaKlasa:
    pass

moj_prvi_objekt = MojaPrvaKlasa()
moj_drugi_objekt = MojaPrvaKlasa()
moj_treci_objekt = MojaPrvaKlasa()
```


ALGEBRA

6

Metode

- Klasa može sadržavati proizvoljan broj metoda
  - Svaka funkcija unutar klase se naziva metoda
- Prvi parametar svake metode se obavezno mora zvati `self`
  - Python automatski u njega stavlja objekt na kojemu je metoda pozvana
  - (u stvari, može se zvati kako god želimo)
- Kad želimo pozvati metodu, pišemo objekt, točku pa metodu
  - Kao da zovemo funkciju, ali koja se nalazi na objektu
  - Parametar `self` preskačemo, njemu vrijednost dodjeljuje Python automatski

Primjer metoda

`self` je uvijek prvi parametar svake metode kad dizajniramo klasu

```
class Kalkulator:
    def zbroji(self, a, b):
        print(a, '+', b, '=', a + b)

    def mnozi(self, a, b):
        print(a, '*', b, '=', a * b)

calc = Kalkulator()
calc.zbroji(17, 5)
calc.mnozi(5, 6)
```

Kad zovemo metodu, nikad ne smijemo pisati `self`

Primjeri

1. Napišite klasu Crtanje koja sadrži dvije metode. Neka prva metoda primi *n* i nacrta kvadrat širine i visine *n*. Neka druga metoda primi *a* i *b* i nacrta pravokutnik širine *a* i visine *b*. Napravite objekt i demonstrirajte rad.
2. Napišite klasu Raspon koja sadrži tri metode. Neka prva metoda primi dva broja i ispiše sve brojeve između njih, uključujući i njih. Neka druga metoda primi dva broja i ispiše deset jedinstvenih slučajnih brojeva između ta dva broja. Neka treća metoda ispiše sve brojeve djeljive sa 37 između 1 i 10.000. U svim metodama ispis ispisujte brojeve odvojene razmacima. Napravite objekt i demonstrirajte rad.

ALGEBRA

9

Varijable na klasi

- Osim metoda, klasa može sadržavati i varijable
  - Svaki objekt u njima čuva svoje podatke
- S varijablama na klasi postupamo jednako kao s metodama:
  - Ako varijabli pristupamo iz klase, moramo pisati ispred naziva varijable prefix `self.`
  - Ako varijabli pristupamo izvan klase, ne smijemo pisati prefiks `self.` jer će ga Python dodati automatski
- U svakom zadatku će pisati smijemo li uopće izvana pristupati varijablama iz klase ili ne
  - Programeri ne smiju, mi ćemo smjeti ☺

ALGEBRA

10

Primjer varijabli na klasi

```
class Kalkulator:
    def postavi(self, a, b):
        self.a = a
        self.b = b

    def zbroji(self):
        print(self.a, '+', self.b, '=', self.a + self.b)

    def mnozi(self):
        print(self.a, '*', self.b, '=', self.a * self.b)

calc = Kalkulator()
calc.postavi(7, 3)
calc.zbroji()
calc.a = 1000
calc.b = 7
calc.mnozi()
```

Kreiramo varijable iz klase pa moramo pisati `self`

Koristimo varijable iz klase pa moramo pisati `self`

Koristimo varijable izvan klase, ne smijemo pisati `self`

ALGEBRA

Primjeri

3. Napišite klasu koja može čuvati podatke o imenu i prezimenu studenta. Napišite metodu koja ispisu ime i prezime ovako: **prezime, ime**
   Napravite jedan objekt pa pozovite metodu. Smijete izravno pristupati varijablama klase.
4. Promijenite prethodni zadatak tako da izvana ne pristupate varijablama klase.
5. Napišite klasu za čuvanje širine i visine pravokutnika te metode za ispis opsega i površine. Napravite objekt na temelju učitane širine i visine pa ispišite njegovu površinu i opseg. Smijete izravno pristupati varijablama klase.
6. Promijenite prethodni zadatak tako da izvana ne pristupate varijablama klase.

ALGEBRA

Konstruktor

- Posebna, opcionalna metoda se zove konstruktor
  - Obavezno se izvršava u trenutku izrade objekta
- Naziv konstruktora je obavezno `__init__`
  - Ispred i iza riječi `init` se nalaze dvije podvlake
  - Također kao prvi parametar mora primiti `self`
- Konstruktor se koristi za postavljanje početnog stanja objekta
  - Najčešće se koristi za definiranje varijabli na klasi

ALGEBRA

13

Primjer jednostavnog konstruktora

```
class Pravokutnik:
    def __init__(self):
        print('Pozdrav iz konstruktora')

p = Pravokutnik()
```

U ovoj liniji se poziva konstruktor

ALGEBRA

14

Primjer primjene konstruktora

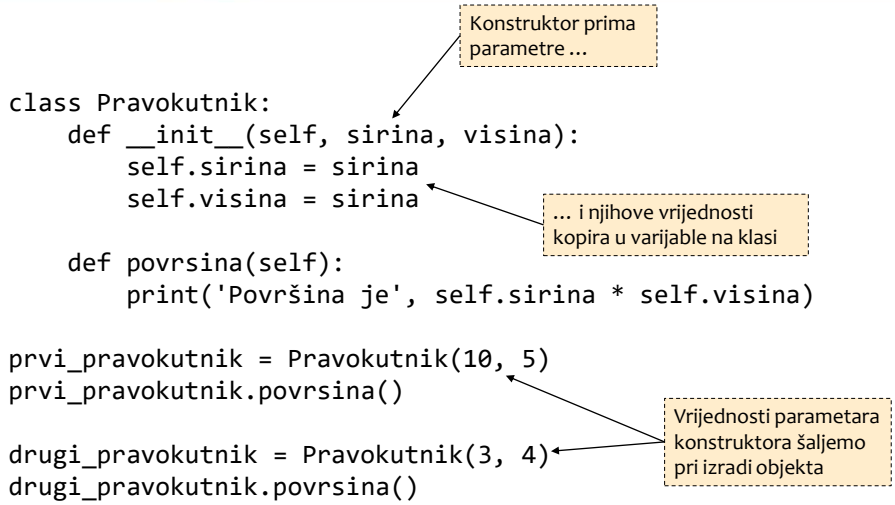
```
class Pravokutnik:
    def __init__(self):
        self.sirina = 10
        self.visina = 5

    def povrsina(self):
        print('Površina je', self.sirina * self.visina)

    def opseg(self):
        print('Opseg je', 2 * self.sirina + 2 * self.visina)

p = Pravokutnik()
p.povrsina()
p.opseg()
print(p.visina)
```

Jedini zadatak ovog konstruktora je kreiranje varijabli na klasi

Parametri konstruktora

- Konstruktoru možemo definirati parametre kao i svakoj metodi
  - Najčešće služi korisniku objekta da postavi početno stanje
- Pri izradi objekta šaljemo vrijednosti parametara

Primjer konstruktora i parametara




```
class Pravokutnik:
    def __init__(self, sirina, visina):
        self.sirina = sirina
        self.visina = sirina

    def povrsina(self):
        print('Površina je', self.sirina * self.visina)

prvi_pravokutnik = Pravokutnik(10, 5)
prvi_pravokutnik.povrsina()

drugi_pravokutnik = Pravokutnik(3, 4)
drugi_pravokutnik.povrsina()
```


ALGEBRA

Primjeri

7. Napišite klasu koja može čuvati podatke o imenu i prezimenu studenta. Napišite konstruktor i metodu koja ispisu ime i prezime ovako: prezime, ime
   Napravite jedan objekt pa pozovite metodu.
8. Napišite klasu za čuvanje širine i visine pravokutnika te konstruktor i metode za ispis opsega i površine. Napravite objekt na temelju učitane širine i visine pa ispišite njegovu površinu i opseg. Smijete izravno pristupati varijablama klase.

ALGEBRA

Primjeri

9. Napišite klasu za čuvanje podataka o knjigama (naslov, autor, godina izdanja). Napišite sljedeće metode:
   - Konstruktor
   - Metodu za ispis podataka o knjizi u formatu:
     Knjigu *x* napisao je *y* godine *z*
   - Metodu koja ispisuje „Vrijedna knjiga" ako je knjiga izdana prije 1901. godine, odnosno „Standardna knjiga" ako nije.

   Napravite objekt s vrijednostima učitanim od korisnika. Demonstrirajte rad metoda.

Recept

- Dobar pristup rješavanju zadataka je ovaj:
  1. Napišite klasu
  2. Napišite konstruktor koji prima `self` i sve ostale parametre koji vam trebaju
  3. U konstruktoru prekopirajte vrijednosti iz parametara u varijable na klasi
  4. Napišite sve ostale metode pazeći da prvi parametar bude `self`
  5. Napišite kôd izvan klase kojim izrađujete objekte i pozivate njihove metode

Primjeri

10. Napišite klasu za čuvanje podataka o 2D točkama. Napišite i metodu koja prima drugu točku i ispisuje udaljenost našeg objekta do druge točke. Učitajte dvije točke od korisnika i ispišite njihovu udaljenost koristeći formulu:
    d = $\sqrt{(x_2 - x_1)^2+(y_2 - y_1)^2}$
11. Napišite klase za podataka o kružnici (x koordinata središta p, y koordinata središta q, polumjer) i točkama (x i y). Dodajte na kružnicu metodu koja prima točku i ispisuje je li točka na kružnici. Učitajte od korisnika jednu kružnicu i jednu točku i pozovite metodu. Točka (x, y) je na kružnici sa središtem u točki $S(p,q)$ i polumjerom r ako je zadovoljeno sljedeće: $(x - p)^2+(y - q)^2 = r^2$

ALGEBRA

Kontejneri objekata

- Kao i ugrađeni tipovi podataka, i objekte možemo staviti u listu, rječnik ili n-torku

ALGEBRA

Objekti u listi

```
class Osoba:
    def __init__(self, ime, prezime):
        self.ime = ime
        self.prezime = prezime

    def ispisi(self):
        print('Pozdrav, ja sam', self.ime, self.prezime)

ekipa = [
    Osoba('Maja', 'Majić'),
    Osoba('Iva', 'Ivić')
]

ekipa.append(Osoba('Juro', 'Jurić'))

o = Osoba('Ivana', 'Ivanić')
ekipa.append(o)

for o in ekipa:
    o.ispisi()
```

Poziv dva konstruktora

Poziv jednog konstruktora

Poziv još jednog konstruktora

ALGEBRA

Objekti u rječniku

```
class Osoba:
    def __init__(self, ime, prezime):
        self.ime = ime
        self.prezime = prezime

    def ispisi(self):
        print('Pozdrav, ja sam', self.ime, self.prezime)

petorka = {
    'play': Osoba('Miro', 'Mirić'),
    'bek šuter 1': Osoba('Perica', 'Perić'),
    'bek šuter 2': Osoba('Jurica', 'Jurić'),
    'krilo': Osoba('Marko', 'Markić'),
}

petorka['centar'] = Osoba('Josip', 'Josić')

for pozicija, igrac in petorka.items():
    igrac.ispisi()
```


ALGEBRA

Primjeri

12. Napišite klasu za čuvanje podataka o pravokutnicima i metodu za ispis površine. Napravite listu od 3 objekta i ispišite njihove površine.
13. Nadopunite prethodni program tako da klasa može ispisati i opseg. Ispišite sve pravokutnike čija je površina ili opseg veći od 20.
14. Napišite klasu za čuvanje podataka o osobama (ime, prezime, godina). Napravite listu od 3 osobe i učitajte ih od korisnika. Ispišite prosjek godina svih osoba.
15. Napišite klasu za čuvanje podataka o osobama (ime, prezime, godina). Napravite listu od 3 osobe i učitajte ih od korisnika. Ispišite ime i prezime najstarije osobe.

ALGEBRA

25

Primjeri

16. Omogućite čuvanje podataka o studentima (ime, prezime, JMBAG) i kreirajte listu od 5 studenata. Omogućite korisniku pretraživanje studenata po JMBAG-u. Ako traženi JMBAG postoji, ispišite o kojemu se studentu radi. Ako ne postoji, napišite to.
17. Omogućite čuvanje podataka o proizvodima (naziv, boja, cijena). Učitavajte od korisnika proizvode dok on to želi, a onda ispišite prosječnu cijenu svih proizvoda.
18. Omogućite čuvanje podataka o prirodnim brojevima (struktura treba imati sadržavati broj o kojemu se radi i podataka je li prost ili ne). U listu stavite sve brojeve između 1 i 1000 i za svaki izračunajte je li prost ili ne. Nakon toga, u jednoj petlji ispišite proste, a u drugoj one koji nisu prosti.

ALGEBRA

26

Primjeri

19. Omogućite čuvanje podataka o 2D točkama. Neka korisnik učita točaka koliko želi. Ispišite udaljenosti između svih parova točaka. Formula za udaljenost dvije točke je: d = $\sqrt{(x_2 - x_1)^2+(y_2 - y_1)^2}$. Primjerice, ako korisnik unese točke $T_1(0,0)$, $T_2(3, 4)$ i $T_3(-5, 5)$:

    Tocke (0, 0) i (3, 4) su udaljene 5
    Tocke (0, 0) i (-5, 5) su udaljene 7.07107
    Tocke (3, 3) i (-5, 5) su udaljene 8.06226

ALGEBRA

Za sljedeće predavanje

- Ponoviti sve iz ovog predavanja
- Pročitati i isprobati primjere:
  - https://www.programiz.com/python-programming/object-oriented-programming
  - https://www.programiz.com/python-programming/class
  - https://www.learnpython.org/en/Classes_and_Objects
  - https://docs.python.org/3/tutorial/classes.html

ALGEBRA