VEŽBA 8

Pregled

Python omogućuje i objektno orijentisano programiranje – OOP. Suštinu ovog pristupa predstavljaju klase i objekti. Prvi su apstraktna definicija objekata – specifikacija koja opisuje ponašanje objekata određenog tipa tj. klase, dok su drugi konkretne realizacije klasa. Klase se definišu tokom razvoja programa, a u trenutku izvršavanja programa kreiraju se objekti u skladu sa definisanim klasama. Objekti su nosioci realnih podataka koji se u programu obrađuju.

Specijalne metode objekata za definisanje specijalnih operacija

`__str__` - konverzija sadržaja objekta u string. `print` koristi ovu funkciju za ispis.

`__repr__` - slično kao i prethodna funkcija, ali je koristi komandni interpreter za ispis

`__len__` - izračunavanje dužine objekta. Ovu funkciju poziva funkcija `len`.

`__add__` - funkcija koja se poziva za obavljanje binarne operacije `+`

`__mul__` - funkcija koja se poziva za obavljanje binarne operacije `*`

Zadaci

1. Razviti klasu `poly` koja omogućuje da se definišu objekti tipa polinoma. Koeficijenti polinoma treba da se skladište kao privatni podatak klase tipa n-torke, gde nulti član n-torke sadrži nulti koeficijent polinoma. Koeficijenti se zadaju prilikom kreiranja objekta – pozivom konstruktora. Obezbediti metode za:

   a) vraćanje koeficijenata polinoma u vidu n-torke – metoda `getcoefs`,

   b) sabiranje dva polinoma operatorom `+`,

   c) ispis polinoma (funkcionalnost koju obezbeđuje funkcija `str`) u obliku:

   ```
   (a0, a1, .... , an)
   ```

   Klasu treba definisati u posebnom modulu u fajlu `polynomial.py`. Testiranje raditi iz drugog Python programa (modula) ili iz interaktivnog moda Python interpretera uz prethodni uvoz modula `polynomial`.

2. Klasi `poly` dodati metode za

   a) množenje dva polinoma (operatorom `*`)

   b) izračunavanje vrednosti polinoma za zadatu vrednost nezavisne promenljive (`x`) - metoda `calc`.

   Testirati kao i u prethodnom zadatku.

3. Mehanizmom nasleđivanja klase `poly` napisati klasu `polysin` koja se može koristiti za izračunavanje aproksimacije funkcije sin(x) pomoću polinoma (Maclaurinov red). Prilikom kreiranja objekta konstruktor ne zahteva nikakve parametre. Polinom za aproksimaciju treba

da bude 5. reda. Klasu `polysin` definisati u fajlu `polysin.py`. Neophodno je da ova definicija klase importuje modul `polynomial`.

Testirati novu klasu poređenjem rezultata izračunavanja sin(x) pomoću objekata tipa `polysin` sa rezultatima funkcije `math.sin`.

4. Koristeći objekte tipa `polysin` izračunavati aproksimaciju sin(x) polinomom 5. stepena. Napisati program koji ispisuje rezultat za ugao izražen u stepenima (kao ceo broj) zadat preko komandne linije. Za konverziju stepena u radijane imati u vidu da je 1° = (π/180°) rad.
5. **Dodatni (neobavezni) zadatak:** dopuniti klasu `poly` tako da ispisivanje polinoma radi u obliku:

```
7*x^3 + 4*x^2 - 2*x^1 + 1*x^0
```

Dodaci

Polinom

Red polinoma odgovara najvišem stepenu uz nezavisne promenljive x. Polinom m -tog stepena ima oblik:

$$P(x)=a_m x^m+a_{m-1}x^{m-1}+\cdots+a_1 x+a_0$$

Jasno je da polinom m -tog reda ima $(m+1)$ koeficijenata zbog čega je za njihovo skladištenje potrebna n-torka sa $(m+1)$ elemenata.

Izračunavanje proizvoda dva polinoma

$$P_A P_B=\sum_{i=0}^{n}\sum_{j=0}^{m} a_i b_j x^{i+j}$$

n je red polinoma A, a m red polinoma B.

Izračunavanje vrednosti polinoma za nezavisnu pomenljivu x

$$P(x)=\sum_{i=0}^{n} a_i x^i$$

n je red polinoma.

Koeficijenti polinoma za izračunavanje funkcije sin(x) Maclaurinovim redom

$$a_i=\begin{cases} 0, & za\ i\ parno\ ili\ 0 \\ \dfrac{(-1)^{(i-1)/2}}{i!}, & za\ i\ neparno \end{cases}$$

Konačni polinom treba da bude sledećeg oblika:

$$\sin x=x-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}+\ldots$$

Obavezno pogledati objašnjenja u vežbi 7.

Algoritam za izračunavanje polinoma za nezavisnu promenljivu x

U ovom pseudokodu `a` je niz sa koeficijentima polinoma `(n-1)`-tog stepena, a `x` vrednost nezavisne promenljive. Koeficijenti a_i sadržani su u elementima niza `a` i dostpni su funkciji kao `a[i]` u trenutku njenog pozivanja (i izvršavanja).

```
function calc(x)
{
  r = 0
  i = 0
  xc = 1
  while (i<n)
  {
    r = r + a[i]*xc
    xc = xc * x
    i = i + 1
  }
  return r
}
```