Praktikum iz programiranja 1

2024/25

Funkcije

Potprogram

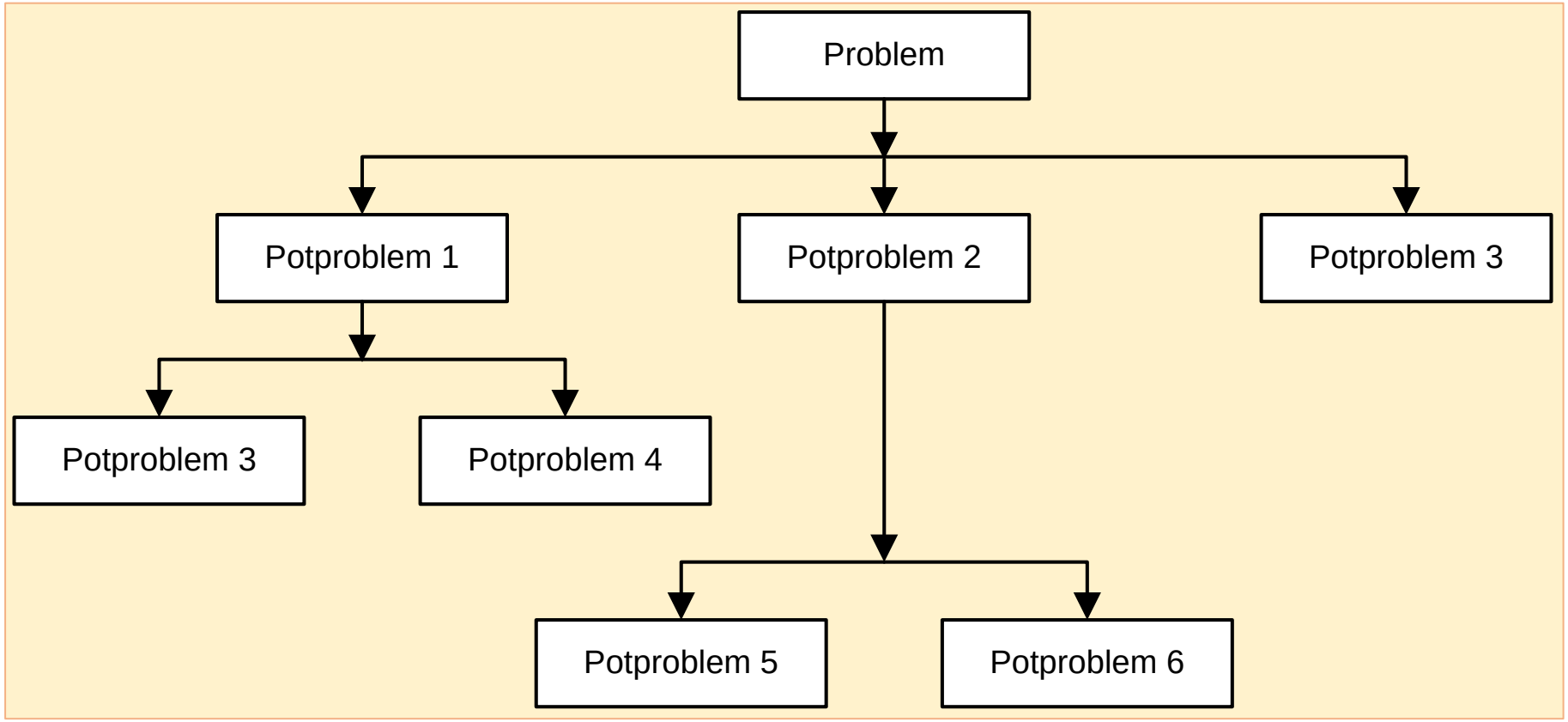
- Složeniji programi se mogu podeliti na odgovarajuće logičke i funkcionalne celine, takozvane `potprograme`.
- Potprogram omogućava organizovanje određenog broja instrukcija u celine koje obavljaju određene zadatak.
- Potprogram preuzima ulazne veličine i vrši njihovu obradu u cilju dobijanja određenih rezultata
- Potprogram možemo pozivati više puta u toku izvršavanja programa.

Prednosti korišćenja potprograma

- Omogućava koncentrisanje na izvršavanje samo određenog zadatka
- Različiti članovi razvojnog tima mogu razvijati različite potprograme koji se kasnije sklapaju u jednu celinu
- Omogućava izvršavanje istog zadatka na više mesta u programu jednostavnim pozivanjem odgovarajućeg potprograma

Programiranje “odozgo na dole”

- Funkcije i procedure omogućavaju realizaciju programiranja "odozgo na dole". Ovaj metod podrazumeva razbijanje problema na potprobleme



Funkcije

- Funkcija je samostalan deo programa koji obavlja određeni zadatak
- Preko svog naziva i liste parametara, delu programa iz koga je pozvana vraća odgovarajuće rezultate
- Svaka funkcija ima jedinstveni naziv preko koga se može pozvati proizvoljan broj puta iz bilo kog dela programa u cilju izvršenja tog zadatka.

Funkcije

- Python pruža mogućnost definisanja korisničkih funkcija

```
def ime_funkcije(parametri_funkcije) :
    telo funkcije
```

- Funkcija mora biti definisana pre prvog korišćenja(poziva)

```
def stampaj_izjavu():
    print ("Moje ime je Ana i OK sam.")
    print ("Spavam celu noc i radim ceo dan.")

stampaj_izjavu()

print (stampaj_izjavu)
print (type(stampaj_izjavu))
```

```
Moje ime je Ana i OK sam.
Spavam celu noc i radim ceo dan.
<function stampaj_izjavu at 0x0341FE40>
<class 'function'>.
```

Poziv funkcije iz funkcije

- U telu funkcije može se naći poziv ranije definisane funkcije

```
def stampaj_opet():
    stampaj_izjavu()
    stampaj_izjavu()


stampaj_opet()
```

```
Moje ime je Ana i OK sam.
Spavam celu noc i radim ceo dan.
Moje ime je Ana i OK sam.
Spavam celu noc i radim ceo dan.
```

Funkcije sa parametrima

- Funkcija može biti definisana sa jednim ili više parametara.
- Poziv funkcije mora odgovarati definiciji funkcije po broju i redosledu parametara.
- Parametri navedeni u opisu funkcije nazivaju se formalni parametri.
- Parametri navedeni pri pozivu funkcije nazivaju se stvarni parametri.
- Formalni i stvarni parametri se moraju podudarati po broju i po tipu.
- Pri izvršavanju funkcije, formalnim parametrima se dodeljuju vrednosti stvarnih parametara.

Funkcije sa parametrima

```
def stampaj_2puta(param):
    print(param)
    print(param)

stampaj_2puta('Spam')
stampaj_2puta(math.pi)
stampaj_2puta('Spam'*4)
stampaj_2puta(17)
```

```
Spam
Spam
3.141592653589793
3.141592653589793
SpamSpamSpamSpam
SpamSpamSpamSpam
17
17
```

Lokalne promenljive funkcije

- Promenljive definisane i korišćene unutar funkcije, nisu vidljive van nje

```
def saberi(a,b):
    c = a + b
    print (c)

saberi(3,5)
print(c)
```

```
8
NameError: name 'c' is not defined
```

Lokalne promenljive funkcije

- Promenljive definisane I korišćene unutar funkcije, nisu vidljive van nje

```
c = 10
def saberi(a,b):
    c = a + b
    print (c)

saberi(3,5)                8
print(c)                   10
```

Globalne promenljive funkcije

- Promenljive definisane I korišćene unutar funkcije, nisu vidljive van nje

```
c = 10
d = 15
def saberi(a,b):
    c = a + b + d   #globalne promenljive mogu da koristim
ne i da ih menjam
    print (c)

saberi(3,5)                23
print(c)                   10
```

Globalne promenljive funkcije

- Promenljive definisane I korišćene unutar funkcije, nisu vidljive van nje

```
c = 10
d = 15
def saberi(a,b):
    d = d + 1 #local variable 'd' referenced before assignment
    print (d)

saberi(3,5)
print(c)
```

Globalne promenljive funkcije

- Promenljive definisane I korišćene unutar funkcije, nisu vidljive van nje

```
c = 10
d = 15
def saberi(a,b):
    global d
    global c
    c = a + b
    d = d + 1
    print (d)
saberi(3,5)                16
print(c)                   8
```

Vraćanje vrednosti iz funkcije

- Funkcija kao rezultat može da vrati jednu vrednost

```
a = saberi (12,26)          38
print (a)                   None

def saberi2 (a,b):
    c = a + b
    return c

saberi2 (3,5)
a = saberi2 (12,26)
print (a)                   38
```

Funkcije sa više rezultata

- U nekim situacijama funkcija treba da vrati više vrednosti. Na primer, želimo da pretvorimo centimetre u metre i preostale centimetre.

```
def cm_u_mcm(cm):
    return (cm // 100, cm % 100)

(m, cm) = cm_u_mcm(178)
print(178, "cm", "=", m, "m", "i", cm, "cm")
```

Funkcije sa više rezultata

- Napiši funkciju koja na osnovu broja sekundi proteklih od prethodne ponoći izračuna trenutno vreme u satima, minutima i sekundama, vodeći računa da broj sati bude između 0 i 23.

```
def sek_u_satminsek(s):
    sek = (s // (60*60)) % 24
    min = (s // 1) % 60
    sat = (s // 60) % 60
    return  (sat,min,sek)

(sat, min, sek) = sek_u_satminsek(1000)
print(sat, ":", min, ":", sek)
(sat,min,sek) = sek_u_satminsek(7200)
print(sat, ":", min, ":", sek)
```

Odvojeni fajlovi

Pomocne_funkcije.py

```
def pozdrav(ime):
    return f"Zdravo, {ime}!"

def zbir(broj1, broj2):
    return broj1 + broj2
```

Glavni_program.py

```
# Importovanje funkcija iz drugog fajla
import Pomocne_funkcije

# Korišćenje funkcija
ime = "Milica"
print(Pomocne_funkcije.pozdrav(ime))

broj1, broj2 = 10, 15
print(Pomocne_funkcije.zbir(broj1, broj2))
```

```
from Pomocne_funkcije import pozdrav, zbir

print(pozdrav("Milica"))
print(zbir(10, 15))
```

```
import Pomocne_funkcije as pf

print(pf.pozdrav("Milica"))
print(pf.zbir(10, 15))
```