Liste - Jednodimenzionalni niz

Liste su skupovi podataka (varijablā) koji predstavljaju jednu cjelinu. Elementi u listi odvojeni su zarezom, a mogu biti različitih tipova, za razliku od nekih drugih programskih jezika.

Također, lista se prepoznaje po uglatim zagradama unutar kojih se nalaze njezini elementi.

Primjer liste s različitim elementima u listi

lista = [1, "bicikl", 3.14, "auto"]

Primjer stvaranja jedne liste:

lista=[3,6,9,12,23] – za liste obavezno koristimo uglate zagrade (desni Alt+f , desni Alt+g)

Za ispis i pristup elementima liste koristimo **indekse** :

	3	42	12	19	30	59
Index	0	1	2	3	4	5

Prvi element liste ima vrijednost 0 ,ako listu čitamo s lijeva na desno

Lista se može čitati i s desna prema lijevo , ali tada indeks ima negativnu vrijednost, prvi član liste s desne strane ima vrijednost -1.

Indeks	-6	-5	-4	--3	-2	-1

Ispis elemenata liste:

brojevi = [3, 42, 12, 19, 30, 59] print(brojevi[0])	Rezultat je 3
print(brojevi[1])	Rezultat 42

Brisanje pojedinog elementa s liste

brojevi = [59, 42, 30, 19, 12, 3, 199] print(brojevi[0])	Rezultat 59
brojevi.pop(0) print(brojevi)	Rezultat [42, 30, 19, 12, 3, 199]

Korištenje više elemenata liste odjednom

Pravilo : [početni_index:završni_index+1]

lista= ['a', 'b', 'c', 'd']	**Rezultat ['b']**
print(lista[1:2])	

Određeni niz elemenata možemo zamijeniti drugim elementom

lista= ['a', 'b', 'c', 'd']	Zamjenjuje ['a', 'b'] sa ['z']
lista[0:2] = 'z'	
print(lista)	['z', 'c', 'd']

Važno je znati

lista = ['a', 'b', 'c', 'd'] print(lista[:])	['a', 'b', 'c', 'd']

Ukoliko ne definiramo niz, već stavimo samo [:], ispisat će se svi elementi liste

Članove liste možemo ispisivati i pomoću petlje:

lista=	[	3	6	9	12	23	]

for i in range(len(lista)): print(lista[i])	funkcija len(lista) daje nam brojčanu vrijednost koliko imamo članova liste. Pri ispisu liste obavezno moramo navesti naziv liste i u uglate zagrade staviti naziv brojača pomoću kojeg će se petlja kretati po listi.

Primjer: Programski primjer korištenja negativnog indeksa

Napiši program koji unosi ime i ispisuje poruku je li upisano ime muško ili žensko. Ako ime završava na a, ispisat će se poruka 'Žensko ime', inače će se ispisati poruka 'Muško ime.	a= input ('Upisi ime: ')
	if a[-1]=='a':
	print 'Zensko ime'
	else:
	print 'Musko ime'

Provjera da li je neki element u listi ili ne

Sjetimo se četvrtog oblika petlje **for** i korištenje naredbe **in ili not in** umjesto naredbe **range**.

Primjer:

lista=	[	3	6	9	12	23	]
d							

for d in lista: print("Elementi liste",d) rezultat: 3 ,6, 9, 12, 23	Zadatak: Napišite riječ "SLAVICA" i ispišite svako slovo jedno ispod drugoga. x=" SLAVICA" for k in x: print(k) rezultat: ispisat će slovo po slovo riječi SLAVICA

Umetanje elemenata liste na kraj liste

niz=[3,5,7,8,12,13] niz.append(15) print(a)	Metoda append dodaje broj 15 na kraj liste imena niz. Pa će naša lista izgledati ovako niz=[3,5,7,8,12,13,15]

Liste uz samo dodavanje elemenata na kraj liste imaju još mnogo mogućnosti, a najčešće su sljedeće:

Naredba	Objašnjenje	Primjer	Rješenje
max(a)	Vraća najveći element u listi	a=[3,5,7,8,3,2,4] print max(a)	8
min(a)	Vraća najmanji element u listi	a=[3,5,7,8,3,2,4] print min(a)	2
sum(a)	Zbraja sve elemente u listi	a=[3,5,7,8,3,2,4] print sum(a)	32
a.sort()	Sortira elemente u listi	a=[3,5,7,8,3,2,4] a.sort() print (a)	2,3,3,4,5,7,8
del (a)	Briše element liste	a=[3,5,7,8,3,2,4] del a[0] print (a)	5,7,8,3,2,4
a.append()	Dodaje element na kraj liste	a=[3,5,7,8,3,2,4] a.append(6) print (a)	3,5,7,8,3,2,4,6
len(a)	Brojčano prikazuje duljinu liste	a=[3,5,7,8,3,2,4]	7

Primjer s objašnjenjem – stvaranje prazne liste i unos elemenata u listu

Želimo napraviti program za unošenje nekoliko imena u listu, korisnik sam bira koliko elemenata želi unijeti.

Prvo trebamo jednu varijablu u koju će se spremiti unos od strane korisnika s pomoću koje ćemo znati koliko imena on želi unijeti te praznu listu u koju ćemo unositi vrijednosti:

```
unos = int(input("Koliko imena želite unijeti? "))
lista = []
```

Nakon toga možemo krenuti na for petlju, gdje ćemo popunjavati listu:

```
for i in range(0,unos):
    x = input("Unesite ime: ")
    lista.append(x)
```

Nakon što smo u for petlji napravili varijablu x u koju se sprema ime i nakon toga se dodaje na kraj liste, potrebno je nakon izvršenja petlje ispisati cijelu listu, a nakon toga pitati korisnika koje ime želi da se ponovo ispiše.

Imena nećemo ispisivati svako posebno, već ćemo ispisati listu, a to radimo ovako:

```
print(lista)
broj = int(input("Unesite broj za ispis imena "))
```

Sada nam samo preostaje ispisati ime iz liste. Međutim, ovdje se treba sjetiti da elementi počinju od 0, a ne od 1, stoga se vrijednost mora umanjiti za 1:

```
print(lista[broj-1])
```

Na ovaj način napisali smo program, a kako bi to trebalo izgledati pogledajte na sljedećoj slici:

```
unos = int(input("Koliko imena želite unijeti? "))
lista = []
for i in range(0,unos):
    x = input("Unesite ime: ")
    lista.append(x)
print(lista)
broj = int(input("Unesite broj za ispis imena "))
print(lista[broj-1])
```

rješenje: unesimo imena [Ana, Marica, Slavica]

ako odaberem da želimo drugog člana liste , što će se ispisati ?

rješenje: Marica , zašto ?– zbog **print(lista[broj-1])** **tj. zbog -1.**

Primjeri zadataka:

Zadatak_1:

Napiši program koji provjerava duljinu lozinke. Ako upisana lozinka ima manje od 8 znakova, ispisat će se poruka: Slaba lozinka. Ako lozinka ima 8 ili vise znakova ispisat će se poruka: Jaka lozinka.	**a=input("Upiši lozinku:")** **if len(a)<8:** **print ("Slaba lozinka")** **else:** **print ("Jaka lozinka")** **npr:** **Upiši lozinku: luka123** **Slaba lozinka**

Zadatak_2:

Napiši program koji za upisanu riječ ispisuje samo samoglasnike.	**a=input("Upiši riječ:)** **for i in a:** **if i in "aeiouAEIOU":** **print (i)** **npr:** **Upiši riječ: informatika** **i o a i a**

Zadatak_3:

Stvorimo niz točno određene duljine (maksimalno 5 mjesta) i unesimo podatke preko tipkovnice niz=[0]*5 , *5 -koristimo da bi smo listu ograničili na maxsimalno pet mjesta za unos podataka u listu.	niz=[0]*5 for i in range(0,5): niz[i]=int(input("Unesite broj")) for i in range(0,5): print(niz[i])

Pojašnjenje :

niz=[0]*5 for i in range(0,5): niz[i]=int(input("Unesite broj")) for i in range(0,5): print(niz[i])	– definiramo niz za unos 5 elemenata - Za unos elemenata u niz moramo definirati varijablu **niz[i].** - Ispis svih elemenata niza

Zadatak_4:

Kreirajmo niz od maksimalno 10 elemenata koji će se sam popuniti i zbrojite sve parne članove niza.
 	
Ispravite pogreške u programu
	niz=[0]*10
	zbroj=0
	#automatsko popunjavanje niza
	for i in range(0,10):
	niz[i]=niz[i]+1
	#zbroj parnih brojeva u nizu
	for i in range(0,10):
	if niz[i]%2==0:
	zbroj=zbroj +niz[i]
	#ispis niza
	for i in range(0,10):
	print(niz[i])
	print(" zbroj parnih brojeva",zbroj)

Primjeri zadataka:

Primjer_1:

Unesite 6 elemenata u listu i ispišite sve parne brojeve u listi.

```
niz=[0]*6 #deklaracija niza
#unos podataka u niz
for i in range (0,6):
   niz[i]=int(input("unesite  broj: " ))

for i in range (1,6,2):
   if niz[i]%2==0:
      print(niz[i])
```

Zadaci za vježbu :

a) Preuredite program tako da se ispišu svi neparni brojevi u nizu.
b) Ispišite sve brojeve niza djeljive sa 3.
c) Ispišite sve brojeve djeljive sa 3 i 4.
d) Ispišite sve brojeve djeljive sa 3 ili 4.

Primjer_2: Ispis i unos podataka iz liste u novu listu , korištenjem funkcije append

```
lista=[3,6,9,12,15]
for index in range(len(lista)):
        print("lista[",index,"]==",lista[index])
```

Uređeni ispis podataka iz liste

```
lista[ 0 ]== 3
lista[ 1 ]== 6
lista[ 2 ]== 9
lista[ 3 ]== 12
lista[ 4 ]== 15
```

Zadatak:_1 :

```
lista=[3,6,9,12,15]
nova_lista=[]
for i in lista:
   if i%2==0:
      nova_lista.append(i)

print(nova_lista)
```

Ispis parnih brojeva iz liste (lista) i njihovo spremanje u novu listu (nova_lista)

Zadatak_2:

```
lista=[]
broj=1
while broj>0:
    broj=int(input("Unesi broj:  "))
    lista.append(broj)
lista.sort()

parni,neparni=[],[]
for br in lista:
   if br%2==0:
      parni.append(br)
   else:
      neparni.append(br)
print("lista: ",lista)
print("Parni brojevi: ",parni)
print("Neparni brojevi: ",neparni)
```

Kreirat ćemo praznu listu u koju ćemo unositi brojeve sve dotle dok je broj veći od 0. Kada unesemo negativan broj prestat ćemo s unosom i kreirat ćemo dvije liste:

1) **za parne brojeve**
2) **za neparne brojeve**

Zadaci:

a) **Ispišite sve brojeve niza djeljive sa 3 i rezultat pohranite u novu listu.**
b) **Ispišite sve brojeve djeljive sa 3 i 4.i rezultat pohranite u novu listu**
c) **Ispišite sve brojeve djeljive sa 3 ili 4. i rezultat pohranite u novu listu**

PRIMJERI ZADATAKA ZA PROVJERU ZNANJA

Pitanje 1.

Kako bi ste definirali listu koja sadrži brojeve 3, -5, 4, -1 ?

a) lista = [3, -5, 4, -1]
b) lista = (3, -5, 4, -1)
c) lista = {3, -5, 4, -1}

Pitanje 2.

Kako bi ste definirali listu koja sadrži brojeve -1, 63 i ime Petar?

a) list = [-1, 63, "Petar"]
b) list = [-1, 63, Petar]
c) list = {-1, 63, "Petar“}
d) Nije moguće istovremeno definirati list s brojevima i tekstom.

Pitanje 3.

Koja je vrijednost šestog elementa liste?

lista = [76, 187, 163, 193, 141, 103, 163]

Odgovor:

Pitanje 4.

Čemu je jednak indeks šestog elementa liste?

lista = [103, 154, 71, 120, 189, 185, 71]

a) 5
b) 6
c) 185
d) 71

Pitanje 5.

Što će biti rezultat izvršenja sljedećeg koda?

```
lista = [4, -4, 1, 3]
print( lista[ 2 ] )
```

a) 1
b) -4
c) [4, -4]
d) [-4, 3]

Pitanje 6. *

Što će biti rezultat izvršenja sljedećeg koda?

```
lista = [10, 10, 9, 12, 15]
lista[ -1 ] = 0
print( lista )
```

a) [10, 10, 9, 12, 0]
b) [10, 10, 9, 0, 15]
c) [-1, 10, 9, 12, 15]
d) Popis će ostati nepromijenjen. Element s indeksom -1 ne postoji, tako da se naredba neće izvršiti.

Pitanje 7.

Što će biti rezultat izvršenja sljedećeg koda?

```
lista = [-3, -1, 1, 1]
print( len( lista ) )
```

Odgovor:

Pitanje 8. *

Što će biti rezultat izvršenja sljedećeg koda?

```
lista1=[4, -2, "pero", "1"]
print( len( lista1 ) )
```

a) 2
b) 3
c) 4

Pitanje 9. *

Naveden je popis cijena sladoleda u jednoj trgovini. Kojom naredbom možete dobiti najnižu cijenu sladoleda?

```
lista = [87, 74, 25, 26, 46]
```

a) min (popis)
b) lista [-1]
c) popis [0]
d) popis [1]

Pitanje 10. *

Što će biti rezultat izvršenja sljedećeg koda?

```
lista=[4, 1, 3, -5]
print( sum( lista ) / len( lista ) )
```

a) 0
b) 0.75
c) 3

Pitanje 11.

Kojom ugrađenom funkcijom u pythonu možemo sortirati list ?

a) sorted
b) min
c) max
d) sort

Pitanje 12. *

Što će biti rezultat izvršenja sljedećeg koda?

```
voce = [50, 40, 45, 80, 65]
cijena = sort( voce )
print( cijena )
```

a) [50, 40, 45, 80, 65]
b) [40, 45, 50, 65, 80]
c) [2, 0, 1, 4, 3]

Pitanje 13. **

Što će biti rezultat izvršenja sljedećeg koda?

```
povrce = [73, 162, 154, 142, 172, 75, 80]
cijena = sort(povrce)
print(sum( cijena[0 : 4] ) )
```

a) 531
b) 710
c) 370

Pitanje 14. **

Što će biti rezultat izvršenja sljedećeg koda?

```
sladoled = [77, 140, 174, 54, 59, 194, 105]
cijena = sort(sladoled)
print( cijena[-3 : ] )
```

a) Cijene tri najskuplje sladoleda.
b) Cijene četiri najskuplja sladoleda.
c) Cijena najjeftinijeg sladoleda.
d) Posljednja naredba nije ispravno napisana, pa će Python dati poruku o pogrešci

Pitanje 15. **

Što će biti rezultat izvršenja sljedećeg koda?

```
sladoled = [183, 105, 106, 136, 151, 166, 124]
print( sort(sladoled)[0 : 4] )
```

a) [105, 106, 124, 136]
b) [105, 106, 136, 183]
c) [105, 106, 124, 136, 151]
d) [105, 106, 136, 151, 183]
e) Posljednja naredba je pogrešno napisana, pa će Python dati poruku o pogrešci .

Pitanje 16. **

Što će biti rezultat izvršenja sljedećeg koda?

```
sladoled = [171, 148, 50, 50, 126, 142, 178]
print( sort( sladoled[0 : 4] ) )
```

a) [50, 50, 126, 142]
b) [50, 50, 148, 171]
c) [50, 50, 126, 142, 148]
d) [50, 50, 126, 148, 171]

Pitanje 17. *

lista = ['Marko Marković', 'Ivan Ivić', 'Petar Petić', 'Ivan Tot', 'Mario Car', 'Ivan Nađ']

Definirali smo listu s popisom učenika kako su navedeni u dnevniku, kojom naredbom biste dobili broj u dnevniku na kojem se nalazi Ivan Tot.

a) lista.index ("Ivan Tot") + 1
b) lista.find ("Ivan Tot")
c) lista.index ("Ivan Tot")

Pitanje 18. *

Što će biti rezultat izvođenja sljedećeg programa?

```
l= ['Marko Marković', 'Ivan Ivić', 'Petar Petić', 'Ivan Tot', 'Mario Car', 'Ivan Nađ']
k = l.index( "Mario Car" )
print( l[ k-1 ] )
```

a) Marko Marković
b) Ivan Nađ
c) Ivan Ivić

Pitanje 19. *

Što će biti rezultat izvođenja sljedećeg programa?

```
l= ['Marko Marković', 'Ivan Ivić', 'Petar Petić', 'Ivan Tot', 'Mario Car', 'Ivan Nađ']
k = len(l)
print( l[ k ] )
```

a) Ivan Nađ
b) Ivan Ivić
c) Pojavit će se pogreška jer smo pogrešno napisali programski kod.

Pitanje 20. *

Neka je mjerena najviša dnevna temperatura tokom 10 dana u mjesecu siječnju. Koliko puta je dnevna temperatura bila jednaka 0 ?

```
l = [14.2, 9.0, 10.0, 18.6, -2.9, 3.6, 17.7]
_____
print( k )
```

Koju od sljedećih naredbi bi koristili za točno rješenje zadatka ?

a) k = l.count (0)
b) k = l.count ("0")
c) k = broj (l, 0.0)
d) Nijedan od ponuđenih odgovora nije točan.

Pitanje 21. **

Što će biti rješenje ovog programskog koda?

```
l = [2, 5, 8, 9, 13, 1, 9]
k = 3
l[k] = l[0]
l[0] = k
print( l )
```

a) [3, 5, 8, 2, 13, 1, 9]
b) [3, 5, 2, 9, 13, 1, 9]
c) [2, 5, 8, 9, 13, 1, 9]

Pitanje 22. *

Što će biti rješenje ovog programskog koda?

```
l = [13, 11, 18]
k = [13, 16, 15]
n = l + k
print( n )
```

a) [13, 11, 18, 13, 16, 15]
b) [26, 27, 33]
c) 86
d) Treći redak koda nije ispravna Python naredba. Ispisat će se poruka o grešci

Python – Funkcije

Kod kompliciranijih zadataka imamo relativno puno redaka koda, često i dijelova koji se ponavljaju, ali su nam potrebni pri samoj izvedbi programa. Kako bismo izbjegli ponavljanje i ponovno pisanje istog ili sličnog koda, koristimo funkcije.

Funkcije su grupirani dio koda koji izvršava određenu radnju. Primjerice, imamo program gdje na različitim dijelovima zbrajamo i ispisujemo vrijednost. Ako to želimo raditi bez funkcija, nekoliko puta moramo pisati isti kod. S druge strane, uporabom funkcije, to nam neće biti potrebno.

Funkcije mogu biti ugrađene u programski jezik, a mogu ih pisati i korisnici. S ugrađenim funkcijama smo se sreli: **print(), len(), del()..,** dok funkcije koje piše korisnik tek trebamo objasniti.

Kod već ugrađenih funkcija vidimo da se samo imenom funkcije i korištenjem argumenata (tj. vrijednosti unutar zagrada) lako može doći do konačne vrijednosti koja nam je potrebna. Tako ne pišemo dodatne redove koda, nego uz pomoć funkcije jednostavno izvršimo određeni zadatak.

Na isti način djeluju i funkcije koje napiše korisnik. Ono što čini funkciju je ključna riječ **def**, nakon toga ime funkcije, pa zagrade unutar kojih možemo upisati argumente te dvotočka i uvučeni blok naredbi nakon toga:

```
def ime_funkcije():
// blok naredbi
```

Na ovaj način je definirana funkcija, dok se ona poziva na sljedeći način:

```
ime_funkcije()
```

Zapravo, sve je isto kod poziva funkcije kao i kod ugrađenih funkcija programskog jezika. Da sve to ne bi bila samo teorija, riješit ćemo jedan primjer uz pomoć funkcija. Zadatak koji ćemo riješiti je zbrajanje brojeva uz pomoć funkcija.

Prvo nam je potrebno definirati funkciju. Nazvat ćemo ju **zbrajanje**. Sljedeće što nam treba je unutar tijela funkcije napisati kod koji će tražiti unos dva broja te ispisati njihov zbroj. To će izgledati ovako:

```
def zbrajanje():
    a=int(input("Unesite prvi broj:=    "))
    b=int(input("Unesite drugi broj: =  "))
    print("Zbroj brojeva",a,"i",b,"je",a+b)
```

Nakon što smo definirali funkciju, potrebno ju je pozvati kako bi se ona mogla izvršiti. Pošto ćemo napraviti program koji zbraja brojeve **pet puta**, koristit ćemo **for** petlju:

```
for i in range(1,6):
    zbrajanje()
    print("")
```

Sada smo gotovi s izradom i pozivanjem funkcije te možemo pokrenuti program:

Kao što vidite, korištenje funkcija nam je skratilo vrijeme pisanja programa jer smo za zbrajanje i ispis samo jednom pisali kod i nije bilo potrebno kasnije to sve pisati ponovo.

Druga pozitivna stvar je povećanje čitkosti koda. Na vrhu nam se nalazi funkcija u kojoj piše što se radi, dok se kasnije ona samo poziva da izvrši određeni zadatak.

No, treba napomenuti jednu stvar. **Funkcija mora biti definirana prije pozivanja iste**, jer će program inače javiti pogrešku (u našem primjeru funkcija je napisana odmah na početku, a kasnije je tek pozivana).

Sljedeća stvar koju ćemo spomenuti su argumenti. Njih smo već sreli, i to kod gotovo svih ugrađenih funkcija koje smo koristili.

Argumenti su vrijednosti koje prosljeđujemo funkciji. Prosljeđujemo ih tako da unutar zagrade napišemo vrijednost ili ime varijable, liste i sl. koja sadržava određenu vrijednost.

print(“Pozdrav”)

U ovom primjeru vidimo da funkcija print() ima argument “Pozdrav”. Pošto je to funkcija ispisa, ona ispisuje upravo vrijednost tog argumenta.

Kod funkcija definiranih od strane korisnika, korisnik sam određuje koliko će argumenata biti potrebno za izvođenje određene funkcije. Za primjer ćemo uzeti funkciju koja množi dva broja i ispisuje vrijednost. Prvo ćemo definirati dvije varijable, gdje će se u svaku spremiti vrijednost koju korisnik unese. Nakon toga će se funkcija pozivati, a u zagradi će se nalaziti proslijeđeni argumenti (brojevi koje je unio korisnik).

```
def zbrajanje(a,b):
    print("Zbroj brojeva",a,"i",b,"je",a+b)


x=int(input("Unesite prvi broj =  "))
y=int(input("Unesite drugi broj =  "))
zbrajanje(x,y)
```

Ono što može biti zbunjujuće su imena argumenata pri definiranju i pozivu funkcije. Kod definiranja funkcije, imena argumenata možemo odabrati svojevoljno i ne moraju imati ista imena koja će kasnije u ostatku programa biti korištena. **Međutim, u samoj definiciji funkcije, imena argumenata unutar zagrada i u tijelu funkcije moraju biti ista.**

Ovo funkcionira tako da se pri pozivu funkcije stvore tipovi podataka (varijable, liste…) iste kao i proslijeđeni tip podataka, a vrijednosti se preslikaju u njih. Kada funkcija završi, ti se pomoćni tipovi podataka brišu iz memorije.

Upravo iz razloga da se pomoćni podaci korišteni u funkcijama brišu, odnosno, nakon izvršavanja funkcije da se oslobodi korištena memorija, postoji mogućnost da funkcija vrati vrijednost.

Zašto?

Čisto iz praktičnih razloga, odnosno, ako neku vrijednost iz funkcije trebamo koristiti u daljnjem dijelu programa bilo bi korisno da tu vrijednost sačuvamo.

Možemo vrijednost poslati u funkciju, i funkcija tu vrijednost može vrati nazad.

Primjer: Unesite dva boja i zbrojite ih.

```
def brojevi(x, y):
    return x + y


#glavni program

broj = brojevi(1,2)
print(broj)
```

```
Rezultat ovog primjera će biti 3.

U ovom primjeru mi smo željeli
zbrojiti brojeve (1 i 2). Unijeli
smo ih izravno u funkciju.

Kad bi mi htjeli unositi brojeve,
glavni program bi izgledao ovako:
```

```
def brojevi(x, y):
    return x + y


#glavni program

a=int(input(" Unesi prvi broj:  "))
b=int(input("Unesi drugi broj:  "))
broj = brojevi(a,b)
print(broj)
```

Kako bismo to postigli koristimo naredbu **return**. Uzet ćemo za primjer zadatak gdje izračunavamo aritmetičku vrijednost liste.

```
def aritmeticka_sredina(lista):
    suma=0
    for a in lista:
        suma=suma+a
        rezultat=suma/len(lista)
        return rezultat


lista=[3,5,7,5,8,2,1]
rezultat=aritmeticka_sredina(lista)
print("Aritmetička sredina iznosi",rezultat)
```

Na početku smo definirali funkciju koja prolazi kroz listu i računa aritmetičku vrijednost.

Nakon toga, definirali smo varijablu rezultat u koju ćemo spremiti vraćenu vrijednost.

Naravno, kako bi u varijablu spremili vraćenu vrijednost, koristimo znak pridruživanja. Ovo bi mogao biti primjer dijela programa gdje računamo prosjek nekog učenika koji kasnije koristimo u neke druge svrhe.

Primjeri funkcija:

Zadatak_1: Napišite funkciju koja od vas traži da 5 puta unesete neki broj

```
def unos():
    a=int(input("Unesite jedan broj"))

print("poziv funkcije")
for i in range(1,5):
    unos()
```

Ovaj dio programa je po funkcijom unos()

Ovo je dio glavnog programa , on nije pod funkcijom (vidite po uvlaci varijable a).

Zadatak_2:

Napišite funkciju za zbrajanje dva broja , program se poziva 3 puta.

```
def zbroj():
    a=int(input("Unesite jedan broj =  "))
    b=int(input("Unesite drugi broj =  "))
    c=a+b
    print("zbroj brojeva",a,"i",b,"je",a+b)
# glavni program
for i in range(1,4):
    zbroj()
```

Cijeli dio ovog programskog koda spada pod funkciju

Glavni dio programa u kojem pozivamo funkciju

Svi primjeri do sada su primjeri funkcije bez argumenata . Idealne su za programa koji imaju mogućnost višestrukog odabira tj. izbornici npr. Kalkulator s više operacija

Funkcije s argumentima

Primjer_1:

```
def mnozenje(a,b):
    print("Umnožak brojeva",a,"i",b,"je",a+b)

# glavni program

b1=int(input("Unesi prvi broj  "))
b2=int(input(unesi drugi broj  "))

#poziv funkcije s argumentima

mnozenje(b1,b2)
```

Argumenti u funkciji nam omogućuju unos vrijednosti iz glavnog programa .

Imena argumenata koja su definirana u funkciji ne moraju biti ista.

Bitno je da broj argumenata u funkciji bude jednak broju argomenata koji ulaze u funkciju.

Primjer_2:

Napišite program za unos nekog broja i provjeru da li je taj broj paran ili negativan.

Definirat ćemo dvije funkcije, jednu za provjeru da li je broj paran, a drugu da li je broj neparan.

```
def paran(n):
    if n%2==0:
        print("broj je paran",n)
    else:
        print("Broj nije paran",n)

def negativan(n):
     if n<0:
         print("broj je negativan",n)
     else:
         print("Broj je pozitivan",n)

#glavni program

unos=int(input("Unesite neki broj  "))
paran(unos)
negativan(unos)
```

Primjer korištenja više funkcija u okviru jednog programa. To predstavlja ozbiljnije programiranje.

Pogledajte kod:! – funkciju paran koristimo za provjeru parnosti broja.

Funkciju negativan koristimo za provjeru da li je broj pozitivan ili ne.

Glavni program omogućuje unos broja i poziva funkcije da provjere u koju funkciju taj broj pripada

Zadatak: nadopunite program tako da tri puta unesete brojeve i za svaki broj da se provjeri njegov status.

Možemo vrijednost poslati u funkciju, i funkcija tu vrijednost može vrati nazad.
RETURN

Primjer_1:

```
def zbroj(n):
    s=0
    for i in range(1,n+1):
        s=s+i
    return s

def unos():
    n=int(input("Unesi neki broj  "))
    print(zbroj(n))
unos()
```

Unesimo neki broj npr: 3, rezultat će nam biti 6 kako??

Broj 3 nam ulazi u funkciju zbroj i ulazi u petlju for.

Prolazi kroz petlju

1. s=0+1
2. s=1+2
3. s=3+3
4. return s pamti vrijednost 6 i vraća vrijednost funkcije nazad , a print(zbroj(n)) ispisuje vrijednost koju je primila od funkcije zbog naredbe RETURN.

Primjer_2:

Zbrajanje dva broja : u funkciji zbroj , zbrajanje je izvršeno u naredbi return a+b	def zbroj(a,b):
	  return a+b
	
	def unos():
	  a=int(input("Unesi prvi broj "))
	  b=int(input("Unesi drugi broj "))
	  print(zbroj(a,b))
	unos()

Primjer_2a:

Prethodni zadatak ali sada funkcija zbroj vraća pomoću return rezultat zbrajanja.
Obratite pažnju , u parametrima unosa koristili smo varijable f i g , a u funkciji zbroj koristili smo a i b.
Zadatak: *dopunite program taka da zbraja tri broja !!*
	def zbroj(a,b):
	  c=a+b
	  return c
	
	def unos():
	  f=int(input("Unesi prvi broj "))
	  g=int(input("Unesi drugi broj "))
	  print(zbroj(f,g))
	
	unos()

Primjer_3:

Napravi program koji pomoću funkcije ispisuje broj pojavljivanja slova A u nekom tekstu koji sami unosimo.
 	
Nadam se da ste razumjeli naredbu return.
Hvala!!
	def zbroj(s):
	  d=len(s)
	  br=0
	  for i in range(d):
	  if s[i]=="A" or s[i]=="a":
	  br=br+1
	  return br
	#glavni program
	tekst=input("Unesi tekst: ")
	print("Broj pojavljivanja slova A: ", zbroj(tekst))

Funkcija unosa i ispisa liste

```
def unos(niz):
    for i in range(0,5):
        niz[i]=int(input("Unesi elemnte niza  "))



def ispis(niz):
    for i in range(0,5):
        print(niz[i])


#glavni program

niz=[0]*5
unos(niz)
ispis(niz)
```

Primijenimo stečena znanja o funkcijama na listama za njihovo stvaranje, ispis i razne operacija nad elementima liste.

Složeni primjer sa izvršavanjem programa sa mogućnošću odabira operacije

```
def unos(niz):
    for i in range(0,5):
        niz[i]=int(input("Unesi elemnte niza  "))



def ispis(niz):
    for i in range(0,5):
        print(niz[i])



def obrnuto(niz):
    for i in range(4,-1,-1):
        print(niz[i])

def parni(niz):
    for i in range(1,5,2):
        print(niz[i])



#glavni dio programa i poziv funkcije
niz=[0]*5
suma=0

izbor=0
while 1:
    print("odaberite")
    print("1. Unos elemenata niza")
    print("2. Ispis elemenata niza")
    print("3. Obrnuti ispis elemenata niza")
    print("4. Ispis parnih elemenata niza")
    print("5. IZAZ IZ PROGRAMA")
    izbor=int(input(" Odaberite što želite"))
    if izbor==1:
        unos(niz)
    elif izbor==2:
        ispis(niz)
    elif izbor==3:
        obrnuto(niz)
    elif izbor==4:
        parni(niz)
    elif izbor==5:
        break
```

Pogledajte petlju while i način njene primjene pri stvaranju složenijih programa s mogućnošću izbora.

PRIMJERI ZA PROVJERU ZNANJA

Pitanje 1.

Što će biti rezultat izvršavanja sljedećeg koda?

```
def opseg(a):
    return 4 * a

print(opseg(6))
```

Odgovor:

Pitanje 2.

Koju od ponuđenih linija koda treba dodati na označeno mjesto kako bi se ispravno definirala funkcija koja izračunava kvadrat danog broja?

```
def kvadrat(a):
```

A. return a * a
B. return * 2
C. a * a
D. return kvadrat

Pitanje 3.

Što će biti rezultat izvršavanja sljedećeg koda?

```
def f(a):
    return - 3 * a

print(f(0) - f(-1))
```

Odgovor:

Pitanje 4. *

Što će biti rezultat izvršavanja sljedećeg koda?

```
def f(a):
    return 2 * a + 3

print(f(-2) - f(f(2)))
```

Odgovor:

Pitanje 5. *

Koju vrijednost treba imati varijabla m , da bi rezultat funkcije bio 15 ?

```
def f(a):
    if a % 5 == 0:
        return 2 * a
    else:
        return a + 1

m = int(input("unesi cijeli broj"))
print(f(m))
```

Pitanje 6. *

Editi je trebalo neko vrijeme da pročita knjigu, vrijeme je izraženo u minutama. Ispišite to vrijeme u satima i minutama. Koju od ponuđenih lija koda ćete uzeti da uspješno riješite zadatak

```
def vrijeme(a):
    s = a // 60
    m = a % 60
```

_____

```
x = int(input("Unesi koliko minuta je Edita čitala knjigu"))
(s,m) = f(x)
m = int(input("unesi cijeli broj"))
print(s, m)
```

A. return s, return m
B. return s, m
C. (s, m)
D. return (s, m)

Pitanje 7. **

Što će biti rezultat sljedećeg programa?

```
def f(l,n):
    return l + n

print(f(11,22)," ",f("11","22"))
```

A. 33 "33"
B. 33 "1122"
C. 1122 "1122"
D. 33 33
E. Python okruženje prijavit će pogrešku tijekom izvršavanja zadanog programa.

Pitanje 8. **

Što će biti rezultat sljedećeg programa?

```
def f(l,n):
  return l * n

print(f(2,"3"))
```

A. 6
B. "222"
C. 33
D. Python okruženje prijavit će pogrešku tijekom izvršavanja zadanog programa.
E. Nijedan od ponuđenih odgovora nije točan.

Pitanje 9. **

Dana je funkcija koja izračunava kvadrat zadanog broja.

```
def f(a):
    return a * a
```

Navedite ispravnu oznaku linije koda u kojoj se koristi funkcija koja ispisuje kvadrate svih brojeva od 4 do 10.

```
1. print([f(x) for x in range(4,10)])
2. print([f(a) for x in range(4,10)])
3. print([f(x) for x in range(4,11)])
4. print([f(a) for x in range(4,11)])
```

Odgovor:

Pitanje 10. *

Navedi oznaku funkcije koja će za dati dvoznamenkasti broj, vraća zbroj znamenaka jedinica i desetica.

```
1. def dvocifren(a):
2.     d = a // 10
3.     j = a % 10
4.     return sum(j, d)

5. def dvocifren(a):
6.     d = a // 10
7.     j = a % 10
8.     return (j, d)

9. def dvocifren(a):
10.    d = a // 10
11.    j = a % 10
12.    return j + d
```

Odgovor:

Pitanje 11. **

Zadana je funkcija kojom se izračunava opseg trokuta

```
def f(a, b, c):
  return a + b + c
```

Navedite programsku oznaku koju koristi zadanu funkciju za ispis opsega nekoliko trokuta čije su veličine stranica navedene u listi trokuti.

```
1. trokuti = [(3, 4, 5), (5, 12, 13), (7, 24, 25)]
2. for trokut in trokuti:
3.   print(opseg(*trokut))
4.
5. trokuti = [(3, 4, 5), (5, 12, 13), (7, 24, 25)]
6. for trokut in trokuti:
7.   print(opseg(trokut))
8.
9. trokuti = [(3, 4, 5), (5, 12, 13), (7, 24, 25)] for i in range(len(trokuti)):
10.    print(opseg(trokuti[i]))
11.
```

Odgovor:

Pitanje 12. **

Dat je jedan dio programskog koda.

```
pravokutnik = [(3, 9), (4, 9), (5, 10)]
for p in pravokutnik:
    print(površina(p))
```

Koja od sljedećih definicija funkcije može dati površina kako bi navedeni kôd radio ispravno?

```
1. def povrsina(a):
2.     return a * a
3.
4. def povrsina(a):
5.     return a[0]*a[1]
6.
7. def povrsina(a):
8.     return a(0)*a(1)
9.
10.  def povrsina(a,b):
11.      return a * b
12.
```

Pitanje 13. *

Prikazan je sljedeći kod.

```
a = formirajlistu(10)
print(a)
```

Odaberite među ponuđenim definicijama funkcija , funkciju koju bi ste dodali postojećem kodu koji generira i ispisuje listu parnih brojeva manjih od 10.

```
1.  def formirajlistu(n):
2.        return(range(2,n,2))
3.
4.  def formirajlistu(n):
5.        l = list(range(2,n,2))
6.        return l
7.
8.  def formirajlistu(n):
9.       for i in range(2,n,2):
10.       lista.append(i)
11.       return lista
12.
13.   def formirajlistu(n):
14.     lista = []
15.     for i in range(2,n,2):
16.           lista.append(i)
17.     return lista
18.
19.   def formirajlistu(n):
20.       l = list(range(2,n,2))
21.
```

Pitanje 14. **

Prikazan je sljedeći kod.

```
a = []
formirajlistu(a,10)
print(a)
```

Dovršiti program stvaranjem liste od 10 cijelih brojeva i ispišite listu . Koja od sljedećih definicija funkcije formlist () će biti ispravna i dati odgovarajući rezultat?

```
1. def formirajlistu(lista,n):
2.   x = int(input())
3.   for i in range(n):
4.     lista.append(x)
5.   return lista
6.
7. def formirajlistu(lista,n):
8.   for i in range(n):
9.     x = int(input())
10.      lista.append(x)
      return lista
11.
12.  def formirajlistu(lista,n):
13.    for i in range(n):
14.      x = int(input())
15.      lista[i] = x
16.    return lista
17.
18.  def formirajlistu(a,10):
19.    for i in range(10):
20.      x = int(input())
21.      a[i] = x
22.    return a
23.
```

Pitanje 15.

Što je definirano sljedećim kodom?

```
def ispisiVeci(broj):
  print(broj + 1)
```

A. Postupak koji ispisuje broj koji je za jedan veći od zadanog broja.
B. Definicija podataka je nepotpuna. .
C. Niti jedan od ponuđenih odgovora nije točan.