METODIKA NASTAVE RAČUNARSTVA A

Python – uvodni zadaci

Stefan Mišković

PythonjenaLinuxuautomatskiinstaliran.UbilokomtekstualnomeditorusePythonprogrammožesaču- vati sa ekstenzijom py, a zatim pokrenuti preko terminala (komandne linije) naredbom python naziv.py. Inače, i za Windows i za Linux se može instalirati bilo koji Python IDE, na primer *PyCharm Community*, kojisezaWindowsmožepreuzatisa[ovog](https://www.jetbrains.com/pycharm/download/#section%3Dwindows)linka,azaLinuxsa[ovog](https://www.jetbrains.com/pycharm/download/#section%3Dlinux)linka.Možesekoristitiinekiodonline editora, poput [ovog](https://www.onlinegdb.com/online_python_compiler), [ovog](https://www.tutorialspoint.com/execute_python_online.php)ili[ovog](https://repl.it/repls/ThornyBuzzingRedundancy).

Da biste na windows-u koristili python, morate ga prvo instalirati (pre PyCharm). Instalacije možete naći na ovom [linku](https://www.python.org/downloads/windows/). Preuzmete executable file najnovije verzije i pokrenete ga dvoklikom. Instalacija nakon toga ide pravolinijski.

# PROMENLJIVE I OPERATORI

1. Napisati program koji na standardni izlaz ispisuje*Zdravo svete!*.

*Rešenje.* Funkcija print se koristi za ispisivanje teksta na standardni izlaz.

**print**("Zdravo.svete!")

1. Napisatiprogramkojiispisujevrednostpromenljivenastandardniizlaz.

*Rešenje.* Promenljive u Pythonu nisu tipizirane, već se iz konkteksta zaključuje o kom tipu je reč. Tako možemoimatinekucelobrojnuilirealnupromenljivu,čijitipnijeeksplicitnonaveden.Urešenjuzadatkasu dodatiijednolinijskikomentari,kojipočinjuznakom#.

*# ova promenljiva j e celobrojna ( i n t t i p u C u)*

−

ceo Broj = 7

**print** ( ceo Broj )

*# ova promenljiva j e realna ( f l o a t t i p u C u)*

−

r e a l a n Br o j = 10 . 54

**print** ( r e a l a n Br o j )

1. Ispisaticelobrojnuirealnupromenljivuusklopustringanastandardniizlaz.

*Rešenje.* Za ispisivanje brojevne promenljive u okviru teksta na standardni izlaz, potrebno je kastovati njen tip u string preko funkcije str. Nadovezivanje se vrši pomoću operatora +.

a = b = 6 *# mogu se d o d e l i t i v rednosti i na ovaj nacin*

c = 12 . 3

**print**("Celobrojna.promenljiva.a.=."+**str**(a)) **print**("Celobrojna.promenljiva.b.=."+**str**(b)) **print**("Realna.promenljiva.c.=."+**str**(c))

1. Definisati promenljive celobrojnog, realnog tipa i tipa string. Ispisati tipove tih promenljivih na stan- dardni izlaz. Zatim izvršiti kastovanje, gde god je tomoguće.

*Rešenje.* Iako Python nije sktriktno tipiziran jezik, funkcijom type se može odrediti tip promenljive. Celo- brojna vrednost se može kastovati u realnu vrednost i u string. Realna se može kastovati u string, a pri kastovanju u celobrojnu vrednost, briše se razlomljeni deo broja. String se može kastovati u ceo ili realan broj, ukoliko je njegov zapis podesan.

ceo Broj = 5

realanBroj=10.9 string1="15"

s t r i n g 2 = " 15 . 5 "

s t r i n g 3 = " zdravo "

**print** ( **type** ( ceo Broj )) **print**(**type**(realanBroj)) **print** ( **type** ( s t r i n g 1 )) **print** ( **type** ( s t r i n g 2 )) **print**(**type**(string3))

**print** ( **f loat** ( ceo Broj ))

**print** ( **str** ( ceo Broj ))

*# ovde ce b i t i odsecen raz l oml j en i deo broja*

**print**(**int**(realanBroj)) **print**(**str**(realanBroj)) **print** ( **int** ( s t r i n g 1 )) **print**(**float**(string1)) **print**(**float**(string2))

*#sledece naredbe bi prijavljivale gresku, pa su pod komentarom #print(int(string2 ))*

*# print(int(string3))*

*# prin t ( f l o a t ( s t r i n g 3 ))*

1. Definisatibrojevnepromenljiveinadnjimaizvršitiračunskeoperacijesabiranja,oduzimanja,množenja, deljenja, stepenovanja imodula.

*Rešenje.* Osnovne računske operacije su u Pythonu definisane na sledeći način:

* + sabiranje:+,
  + oduzimanje:−,
  + množenje:∗,
  + deljenje:*/*,
  + celobrojno deljenje:*//*,
  + stepenovanje:∗∗,
  + modul:%.

a = 9

b = 9 . 2

c = 2

**print** ( a + b ) **print**(a c ) **print**(a b c)

∗ ∗

−

*# za dva cela broja u pythonu 2 ovo j e celobrojno d e l j en je , # a u pythonu 3 se t r e t i r a kao d e l j en j e real n i h brojeva* **print** ( a / c)

**print** ( b / c ) **print** ( a %c ) **print** ( b %c )

*# za celobrojno d e l j en j e se najcesce k o r i s t i // umesto /*

**print** ( a // c)

**print** ( b // c)

1. Iskombinovatioperatoredodelesaaritmetičkimoperatorima,radikompaktnijegzapisa.Ispisatidobijene vrednosti na standardniizlaz.

*Rešenje.* Umesto naredbe broj = broj + 2, može se koristiti broj += 2. Slično, može se pisati i -=, \*=,

/=, //=, = i \*\*=.

b r o j = 10 **print**(broj) b r o j +=6

broj = 2 **print**(broj) broj =2

−

∗

b r o j %= 10 **print**(broj) broj = 2 **print**(broj)

∗∗

1. Koristećioperatorepoređenja,uporeditivrednostidvabroja.Istinitosnevrednostiispisatinastandardni izlaz.

*Rešenje.* Operatori poređenja u Pythonu su ==, !=, <, >, <= i >=. Vrednosti izraza koji koriste ove operatore su True ili False.

b roj 2= 2

b roj 3= 3

je d n a k i = broj 2 == broj 3

r a z l i c i t i = broj 2 != broj 3

**print** ( je d n a k i ) **print** ( r a z l i c i t i ) **print** ( b roj 2 > b roj 3 )

*# i z r az se moze n ap i s a t i i u ovom ob l i k u*

**print** ( 1 < broj 2 < broj 3 < 4 )

**print** ( 4 >= 10 >= 3 )

**print** ( b roj 2 <= b roj 3 )

# STRINGOVI

1. Iskombinovati vrednosti dva stringa koristeći operatorkonkatenacije.

*Rešenje.* Operator konkatenacije stringova je +. zdravo = " zdravo "

s v e te=". s v e te "

zdravo Svete = zdravo + s v e te

**print** ( zdravo Svete )

**print** ( " zdravo ." + " svima " )

1. Ispisatistringodgovarajućibrojputamnožećigasaprirodnimbrojem.

*Rešenje.* U Pythonu se stringovi mogu množiti brojevima, a rezultat predstavlja nadovezani string odgo- varajući broj puta. Na primer, za string *zdravo*, množenjem brojem 3, dobija se *zdravozdravozdravo*.

zdravo = " zdravo" b r o j = 3

**print** ( zdravo ∗b r o j )

1. Ispisati vrednosti nekih karaktera stringa, koristećiindeksiranje.

*Rešenje.*Karakterustringasemožepristupitipreko[].Indeksiranjeslevapočinjesa0,aindeksiranjezdesna sa 1. Tako, na primer, za treći karakter sleva indeks je 2, a za pretposlednji karakter indeks je 2. Može se ispisati više karaktera istovremeno, i to na sledećinačin:

− −

* + [i:j] – svi karakteri od indeksa i do indeksa j -1,
  + [i:] – svi karakteri od indeksai,
  + [:j] – svi karakteri do indeksa j -1,
  + [:] – svi karakteristringa.

rec="Republika.Srbija" prvoSlovo=rec[0]

**print** ( prvo Slovo )

*# i s p i s u j e p r e t p os l ed n j e s l ovo*

**print**(rec[ 2])

−

*# i s p i s u j e sva s l ova sem prvog i poslednjeg*

**print**(rec[1: 1 ] ) prvaRec = r e c [ : 9 ] drugaRec = r e c [ 1 0 :] **print** ( prvaRec ) **print**(drugaRec)

−

*# dva nacina i s p i s a celog s t r i n g a*

**print** ( r e c )

**print** ( r e c [ : ] )

1. Proveritidaliseodređenipodstringnalaziunekomstringu.

*Rešenje.* Za proveru da li se podstring nalazi u nekom stringu, koristi se operator in. podstring1="Srbija"

podstring2 ="Srbije"

r e c = " Republika . S r b i ja " **print** ( p od s tr in g 1 **in** r e c ) **print** ( p od s tr in g 2 **in** r e c )

1. Ispisati dužinu stringa. Razmotriti i slučaj kada je string napisan u višeredova.

*Rešenje.* Funkcija koja vraća dužinu stringa je len. Ukoliko string počinje i završava se trostrukim navod- nicima, može se napisati u više redova.

manjiString="ovo.je.manji.string" veciString=*"""ovojeveci string. onsemozenapisatiuviseredova.*

*pocinje i zavrsava se t rostrukim navodnicima"""*

**print**("Manji.string:."+manjiString)

**print** ( " Veci . s t r i n g : ." + v e c i S t r i n g )

**print**("Duzina.manjeg.stringa:."+**str**(**len**(manjiString)))

**print** ( **len** ( v e c i S t r i n g ) )

1. Dati string, koristeći odgovarajuće funkcije, ispisati svim velikim, odnosno svim malimslovima.

*Rešenje.* Funkcija koja uvećava sva slova stringa je upper, a koja umanjuje lower.

drzava="Republika.Srbija"

**print** ( drzava . lower ( ) ) drzavaVelikimSlovima=drzava.upper() **print**(drzavaVelikimSlovima)

1. Razmotritislučajevekadasejednostrukiilidvostrukinavodnicimogunaćiunutarstringa.

*Rešenje.* Jednostruki navodnici se mogu koristiti u stringu koji počinje i završava se dvostrukim, i obratno. (String se ravnopravno može koristiti i ako je naveden i jednostrukim i dvostrukim navodnicima.) Inače, karaktere je potrebno naznačiti sa \’, odnosno \".

string1=’Program.ispisuje:."Zdravo.svete!"’ string2="Program.ispisuje:.’Zdravo.svete!’"

string3=’Program.ispisuje:.\’Zdravo.svete!\’’ string4="Program.ispisuje:.\"Zdravo.svete!\""

string5="Oba.navodnika.’.i.\".u.jednom.stringu"

**print**(string1) **print**(string2) **print**(string3) **print**(string4) **print**(string5)

1. Ispisatibrojevnitippodatkailistring,koristećiformatiranjeuokvirustringa.

*Rešenje.*Ceobrojseformatirapomoćud,realanpomoćuf,astringpomoćus. ceo Broj =6

realanBroj=12.3 rec="zdravo"

**print**("Ceo.broj.je.%d"%ceoBroj)  **print**("Realan.broj.je.%f"%realanBroj) **print**("String.je.%s"%rec)

*# i s p i s i se mogu i skombinovati*

**print**("Brojevi.su.%d.i.%f,.a.rec.%s"%(ceoBroj,realanBroj,rec))

# STRUKTURE PODATAKA

1. Definisati listu, a zatim ispisati neke od njenih elemenata, koristeći indeksiranje. Ispisati dužinuliste.

*Rešenje.*Listapredstavljakolekcijupodataka,kojimogubitiproizvoljnogtipa.ZarazlikuodnizovauC-u, u Pythonu se lista može sastojati i od različitih vrsta podataka. Sintaksa za definisanje liste je lista = [element1, element2, element3]. Indeksiranje se vrši na sličan način kao kod stringa. Dužina liste se određuje pomoću funkcijelen.

*#u l i s t i mogu b i t i t i p o v i r a z l i c i t o g t i p a*

l i s t a = [ " jedan " , " dva" , 3 , 4 . 5 ]

**print** ( l i s t a ) **print** ( **len** ( l i s t a ) ) **print** ( l i s t a [ 0 ] ) **print** ( l i s t a [ −1 ] )

1. Izvršiti nadovezivanjedve liste, a zatim i dodavanje, izmenu i brisanje nekih njenih elemenata.

*Rešenje.* Konkatenacija dve liste se vrši pomoću operatora +. Dodavanje elementa listi se vrši pomoću funkcije append. Izmena ili brisanje elemenata se može izvršiti pristupanjem odgovarajućim indeksima elemenata koji treba da budu izmenjeni ili izbrisani.

l i s t a = [ " jedan " , " dva" , " t r i " , " c e t i r i " ]

**print** ( l i s t a )

*# dodavanje novih elemenata*

l i s t a . append ( " pet " )

l i s t a += [ " s e s t " , "sedam" ]

**print** ( l i s t a )

*# menjanje v rednosti i l i b r i s an j e jednog elementa*

l i s t a [ 1 ] =" s to "

lista[ 1]="hiljadu"

−

**print** ( l i s t a ) **del** l i s t a [ 0 ] **print** ( l i s t a )

*# menjanje v rednosti i l i b r i s an j e v i s e elemenata*

l i s t a [ 2 : 5 ] = [ " m i l i on " ]

**print** ( l i s t a )

l i s t a [ 2 : 4 ] = [ ]

**print** ( l i s t a )

l i s t a [ 0 : ] = [ ]

**print** ( l i s t a )

1. Definisatitorku,azatimjeispisatinastandardniizlaz,uključujućinjenudužinu.

*Rešenje.*Zarazlikuodliste,torkasedefinišepreko().Indeksiranjeiodređivanjenjenedužinesevršislično kao kod liste. Elementi torke se ne mogu brisati niti menjati, a ne mogu se ni nakon definisanja dodavati novi.

torka = ( 4 . 5 , 4 , "sedam" , 2 )

**print** ( torka )

**print**("Duzina.je:.%d."%**len**(torka))

**print** ( torka [ 0 ] )

**print** ( torka [ −1 ] )

1. Definisati rečnik, a zatim i vršiti izmenu, brisanje i dodavanje novih elemenata. Ispisati sve vrednosti ključeva, kao i vrednostirečnika.

*Rešenje.* Rečnici se u Pythonu definišu u obliku recnik = {kljuc1:vrednost1, kljuc2:vrednost2, kljuc2:vrednost2}. Ključevi se mogu ispisati funkcijom keys, a vrednosti funkcijom values. Rečnici predstavljaju uopštenje listi, u smislu da njihovi indeksi (ključevi) ne moraju biti celi brojevi, već bilo koji tip podatka.

imenik= {" Petar " : 2 1 3 , " Jovan" : 7 9 8 , "Ana" : 427 }

**print** ( imenik . keys ( ) ) **print**(imenik.values()) **print**(imenik)

imenik [ " Petar " ] = 214 **print**(imenik["Petar"]) imenik["Milica"]=908 **del** imenik [ " Jovan" ] **print**(imenik)

1. Proveritidalisenekielementnalaziulisti.

*Rešenje.* Slično kao provere da li se podstring nalazi u nekom stringu, preko operatora in se proverava da li se neki element nalazi u datom nizu.

l i s t a = [ " ponedeljak " , " utorak " , " sreda " ]

**print** ( " c e tv r ta k " **in** l i s t a )

pripada = " sreda " **in** l i s t a

**print** ( pripada )

imenik= {" Petar " : 2 1 3 , " Jovan" : 7 9 8 , "Ana" : 427 }

**print** ( "Ana" **in** imenik . keys ( ) )

# NAREDBE GRANANJA

1. Napisatiprogramkojiproveravadalijedužinastringavećaoddatevrednosti.

*Rešenje.* U Pythonu naredba if ima sledeći oblik:

**i f** uslov :

naredbe

Naredbe će se izvršavati, ukoliko je uslov ispunjen. Sve naredbe treba uvući i navesti jednu ispod druge. KodPythonatrebastrogovoditiračunaouvlačenjunaredbi.Ovajprincipsekoristiikodnaredbigrananja i kod ciklusa, kao i kod definisanja funkcija ili klasa. U C-u se, na primer, ne mora strogo voditi računa o uvlačenju,jersenaredbeodvajajuzagradama{}.

vrednost = 12

string1="Zdravo.svete" string2="Zdravo.svete!"

**i f len** ( s t r i n g 1 ) > vrednost :

**print**("Prvi.string.je.duzi.od.%d"%vrednost)

# else :

**print**("Prvi.string.nije.duzi.od.%d"%vrednost)

drugi Duzi= **len** ( s t r i n g 2 ) >vrednost

**i f** drugiDuzi :

**print**("Drugi.string.je.duzi.od.%d"%vrednost)

# else :

**print**("Drugi.string.nije.duzi.od.%d"%vrednost)

1. Napisatiprogramkojiodređujedalijedatibrojpozitivan,negativanilijednaknuli.

*Rešenje.* Naredba grananja if se može napisati i u kombinaciji sa else na sledeći način:

**i f** uslov :

naredbe 1

# else :

naredbe 2

Ukoliko je uslov ispunjen, izvršiće se naredbe iz prve grupe, a ako nije, izvršiće se naredbe iz druge grupe. Može se razmatrati i više uslova istovremeno na sledeći način:

**i f** uslov 1 :

naredbe1

**elif** uslov2:

naredbe2

**elif** uslov3:

naredbe3

# else :

naredbe 4

Možepostojatiproizvoljanbrojelifnaredbi.U ovom primeru,naredbeizprvegrupećeseizvršitiukoliko je ispunjen prvi uslov. Inače, ukoliko je ispunjen drugi uslov, izvršiće se naredbe iz druge grupe. Inače, ukolikojeispunjentrećiuslov,izvršićesenaredbeiztrećegrupe.Inače,izvršićesenaredbeizčetvrtegrupe.

broj1 = 3

−

b roj 2= 19

**i f** b roj 1 >= 0 :

**print** ( " Prvi .b r o j . j e .nenegativan " )

# else :

**print** ( " Prvi .b r o j . j e .negativan " )

**i f** b roj 2 == 0 :

**print** ( " Drugi .b r o j . j e .nula " )

**e l i f** b roj 2 > 0 :

**print** ( " Drugi .b r o j . j e . p o z i t i v a n " )

# else :

**print** ( " Drugi .b r o j . j e .negativan " )

1. Napisatiprogramkojimseunaredbigrananjaproveravajuuslovikojiusebisadrželogičkefunkcijeand, or ilinot.

*Rešenje.* Logičke funkcije and, or i not su definisane u Pythonu i služe za konjukciju, disjunkciju, odnosno negaciju logičkih izraza. Vrednosti logičkih izraza koje ih koriste su definisane na sledeći način:

* + izraz1 and izraz2: vrednost izraza je True ako su vrednosti oba izraza True, a inače jeFalse,
  + izraz1 or izraz2: vrednost izraza je True ako je vrednost bar jednog izraza True, a inače jeFalse,
  + not izraz: vrednost izraza je suprotna vrednosti izraza koji je zadatkao argiment.

Logičkafunkcijanotimanajvećiprioritet,zatimfunkcijaand,anajmanjiprioritetimafunkcijaor. ime = " Petar"

godine = 20

**i f** ime == " Petar " **and** godine == 2 1 :

**print**("Petar.ima.21.godinu")

# else :

**print**("Petar.nema.21.godinu")

**i f** ime == " Jovan" **or** godine == 2 0 :

**print**("Ime.je.Jovan.ili.su.godine.20")

# else :

**print**("Ime.nije.Jovan,.a.godine.nisu.20")

*# redosled operatora j e not , and , or #rezultatuslovajeFalse*

**i f** ime == " Jovan" **or** ime == " Petar " **and not** godine == 2 0 :

**print** ( " P o s l e d n j i .uslov . j e .i s p u n je n " )

# else :

**print** ( " P o s l e d n j i .uslov . n i j e .i s p u n je n " )

# CIKLUSI

1. Koristeći for ciklus, ispisati elemente niza, odnosnokaraktere stringa.

*Rešenje.* U Pythonu je for ciklus oblika:

**for** b r o ja c **in** i n t e r v a l : naredbe

Intervalmožebitinavedenkorišćenjemfunkcijerange,pričemurange(i)označavadasuuintervalusvi brojeviod0do*i*1,arange(i,j)dasuuintervalusvibrojeviizmeđu*i*i*j*1.Elemetimanizaistringa semožepristupatiuforciklusuprekoindeksa(tadasekoristifunkcijarange)ilidirektnouoblikufor element in niz, odnosno for element instring.

− −

*# i s p i s u j e b rojeve od 0 do 4*

**for** i **in range** ( 5 ) :

**print** ( i )

*# i s p i s u j e b rojeve od 3 do 10*

**for** i **in range** ( 3 , 1 1 ) :

**print** ( i )

n i z = [ 4 . 5 , 2 . 3 , 1 1 . 5 , 1 6 . 8 ]

*# elementi niza preko indeksa*

**for** i **in range** ( **len** ( n i z ) ) :

**print** ( n i z [ i ] )

*# elementi niza bez indeksa*

**for** element **in** n i z :

**print** ( element )

r e c = " zdravo "

*# elementi s t r i n g prekoindeksa*

**for** i **in range** ( **len** ( r e c ) ) :

**print** ( r e c [ i ] )

*# elementi s t r i n g a bez indeksa*

**for** element **in** r e c :

**print** ( element )

1. Preko while ciklusa, ispisati kvadrate svih jednocifrenih prirodnihbrojeva.

*Rešenje.* U Pythonu je while ciklusoblika:

**while** uslov :

naredbe

Naredbe će se izvršavati sve dok je ispunjen uslov koji je naveden. b r o j = 1

**while** b r o j < 1 0 : **print**(broj 2) b r o j +=1

∗∗

1. Unutar for ciklusa koji prolazi kroz sve brojeve od 1 do 100, ispisati sve parne brojeve. Ukoliko je broj ispisanihparnihbrojevavećiod40,zaustavitiizvršavanje.

*Rešenje.* Izvršavanje for ciklusa može biti prekinuto korišćenjem naredbe break. Naredbom continue se direktno prelazi na sledeću iteraciju.

broj Parnih = 0

**for** i **in range** ( 1 , 1 0 1 ) :

**i f** i % 2 == 1 :

# continue

**i f** i % 2 == 0 : **print** ( i ) broj Parnih +=1

**i f**broj Parnih > 4 0 :

# break

1. Koristeći while ciklus, ispisati niz Fibonačijevih brojeva. Ukoliko je trenutni ispisani Fibonačijev broj veći od unapred zadatog broja, prekinutiizvršavanje ciklusa.

*Rešenje.* Naredbe break i continue se mogu koristiti i u kombinaciji sa while ciklusom. U ovom primeru se usvakojiteracijidefinišenoviFibonačijevbrojkaozbirprethodnadva,azatimseredom,naosnovunjegove vrednosti,ažurirajuvrednostiprethodnogitrenutnogbroja.

g r a n i c a = 10000 prethodni = 1 **print** ( prethodni ) tr e n u tn i = 1 **while** True :

**print** ( tr e n u tn i )

**i f** tr e n u tn i > g r a n i c a :

# break

novi = tr e n u tn i + prethodni prethodni = tr e n u tn i

tr e n u tn i = novi

# FUNKCIJE

1. Definisatifunkcijukojanastandardniizlazispisuje*Zdravosvete!*,azatimjepozvati.

*Rešenje.* Funkcije se definišu korišćenjem ključne reči def, a zatim navođenjem imena funkcije i zagrada (), unutar kojih se eventualno mogu navesti argumenti. Naredbe unutar funkcije se zatim moraju uvući i navesti jedna ispod druge. U ovom primeru je navedena funkcija koja nema povratnu vrednost (odgovara tipu void u C-u) i koja nema argumenata. Funkcija se zatim može pozvati u glavnom deluprograma.

**def** i s p i s ( ) :

**print**("Zdravo.svete!")

*# pozivanje funkcije ispis*

ispis()

1. Definisati funkciju koja sabira dva broja i pozvati je, pri čemu zbir treba ispisati na standardniizlaz.

*Rešenje.* Funkcija zbir koja je ovde implementirana ima dva argumenta i povratnu vrednost. Povratna vrednost funkcije se dodeljuje preko ključne reči return (na sličan način kao i u C-u).

**def** z b i r ( x , y ) : z = x + y **return** z

a = 10

b =5

c = z b i r ( a , b )

**print** ( " Zbir . j e ." + **str** ( c ) )

1. Definisati funkciju čiji su argumenti string i broj. Ukoliko pri pozivu funkcije broj nije naveden, po- drazumevanavrednostje3.Funkcijatrebadavraćastringnadovezanonolikoputakolikajevrednostbroja.

*Rešenje.* Ukoliko neki argument ima podrazumevanu vrednost, ona se može navesti pri navođenju argu- menata funkcije. Ako se takav argument eksplicitno ne navede, pri izračunavanju će se koristiti njegova podrazumevanavrednost.Argumentikojiimajupodrazumevanuvrednost,morajubitinavedenikaoposled- nji argumentifunkcije.

**def** nadovezi ( rec , b r o j = 3 ) :

**return** r e c ∗b r o j

rec= "zdravo" b r o j = 4

**print** ( nadovezi ( rec , b r o j ) )

**print** ( nadovezi ( r e c ) )