

INFORMATIČKI KLUB
FUTURA

LIGA PROGRAMIRANJA



python

#2

**LIGA PROGRAMIRANJA U PYTHONU ZA
OSNOVNE ŠKOLE – 5. RADIONICA**

Mario Miličević, Informatički klub FUTURA
Dubrovnik, 12. ožujka 2016.



Zaštićeno licencom <http://creativecommons.org/licenses/by-nc-sa/3.0/hr/>



Creative Commons



slobodno možete:

- Dijelite dalje** — možete umnažati i redistribuirati materijal u bilo kojem mediju ili formatu
- Stvarajte prerade** — možete remiksirati, mijenjati i prerađivati djelo



pod slijedećim uvjetima:



- Imenovanje** — Morate adekvatno navesti autora, uvrstiti link na licencu i naznačiti eventualne izmjene. Možete to učiniti na bilo koji razuman način, ali ne smijete sugerirati da davatelj licence izravno podupire Vas ili Vaše korištenje djela.



- Nekomercijalno** — Ne smijete koristiti materijal u komercijalne svrhe.



- Djeli pod istim uvjetima** — Ako remiksirate, mijenjate ili prerađujete materijal, Vaše prerade morate distribuirati pod istom licencom pod kojom je bio izvornik.

U slučaju daljnog korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela. Najbolji način da to učinite je linkom na ovu internetsku stranicu.

Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.

Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.

Tekst licence preuzet je s <http://creativecommons.org/>

Raspored Lige programiranja

- 12.03.2016. – **5. radionica**
- 02.04.2016. – **Finale Lige programiranja**



- Web stranica Lige programiranja:
www.futura.com.hr/liga-programiranja-u-pythonu-2015-2016/



Ponavljanje: Moduli – zbirke funkcija

- Modul se prije korištenja mora uvesti s naredbom **import naziv_modula**
- Funkcija **sqrt** (korijen) je iz **math** modula

```
>>> import math
>>> sqrt(9)
Traceback (most recent call last):
  File "<pyshell#6>", line 1, in <module>
    sqrt(9)
NameError: name 'sqrt' is not defined
>>> math.sqrt(9)                      >>> math.sqrt(2)
3.0                                     1.4142135623730951
```



Ponavljanje: Moduli – zbirke funkcija

- Drugi način uvoza funkcija iz modula s naredbom **from naziv_modula import funkcija1, funkcija2, ...**

```
>>> from math import sqrt, fabs
>>> sqrt(9)
3.0
>>> fabs(-3)
3.0
```

$|3| = 3, \ |-3| = 3$

funkcija **fabs** vraća absolutnu vrijednost broja

- Ako se žele uvesti sve funkcije iz nekog modula koristi se naredba:

```
>>> from math import *
>>> sqrt(9)
3.0
```



Ponavljanje: Modul math

- Najčešće korištene funkcije modula math:

<code>sqrt(x)</code>	korijen broja x
<code>fabs(x)</code>	apsolutna vrijednost broja x
<code>ceil(x)</code>	zaokruživanje na najmanji cijeli broj veći ili jednak broju x
<code>floor(x)</code>	zaokruživanje na najveći cijeli broj manji ili jednak broju x

- <https://docs.python.org/3/library/math.html>

```
>>> from math import *
>>> ceil(5.2)                      >>> floor(5.2)
6                                     5
>>> ceil(6.9)                      >>> floor(6.9)
7                                     6
```



Ponavljanje: Modul time

- Modul **time** sadrži dosta funkcija vezanih za vrijeme, ali nama je trenutno potrebna samo funkcija **time()**:

```
>>> from time import time  
>>> print(time())  
1455056710.04582  
>>> print(time())  
1455056717.3822396
```

koliko sekundi je proteklo od početka "epohe" (tj. od 1.1.1970.)

- Funkcija **time()** najviše se koristi za mjerjenje proteklog vremena.
- <https://docs.python.org/3/library/time.html>

Zadatak: Loto



- Lora igra Loto **4 od 20**. Zanima je koliko je izvlačenja potrebno kako bi bila izvučena njezina kombinacija. Izvlačenja se simuliraju pomoću generatora slučajnih brojeva.
- Ulazni podatci:
 - Lorina kombinacija - **4 broja između 1 i 20**, u istom redu odvojeni prazninama
- Izlazni podatci:
 - Izvučene kombinacije
 - Koliko je izvlačenja bilo potrebno kako bi se izvukla Lorina kombinacija

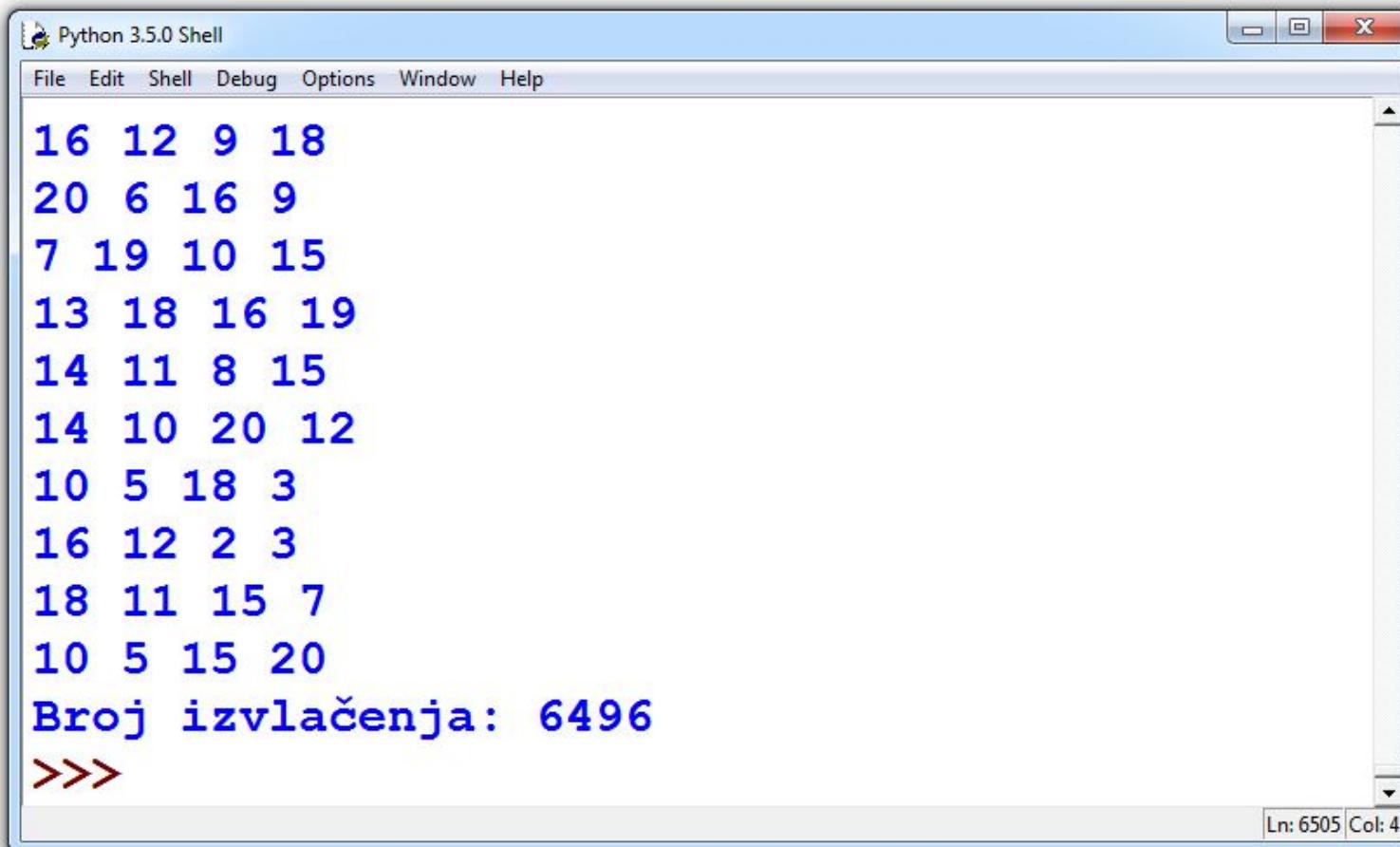
Vrijeme





Zadatak: Loto

- Primjer pokretanja programa, za Lorine brojeve
5 10 15 20:



The screenshot shows a Windows-style application window titled "Python 3.5.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays a list of lottery numbers and a result message. The numbers are listed in groups of four, separated by spaces. The last group contains the number 20. Below the numbers, the message "Broj izvlačenja: 6496" is displayed, followed by a prompt "">>>>". In the bottom right corner of the window, there is a status bar with "Ln: 6505 Col: 4".

```
16 12 9 18
20 6 16 9
7 19 10 15
13 18 16 19
14 11 8 15
14 10 20 12
10 5 18 3
16 12 2 3
18 11 15 7
10 5 15 20
Broj izvlačenja: 6496
>>>
```



Loto - rješenje

```
from random import sample
brojevi = [i for i in range(1, 21)]
br = 0
ul = input('Lorina kombinacija - 4 broja (1-20): ')
lora = ul.split()
for i in range(len(lora)):
    lora[i]=int(lora[i])
while True:
    komb = sample(brojevi, 4)
    print()
    for i in komb:
        print(i, end=' ')
    br += 1
    lora.sort()
    komb.sort()
    if lora == komb:
        print('\nBroj izvlačenja:', br)
        break
```



loto.py

Zadatak: Loto - trajanje progr.



- Za prethodni zadatak još ispisati i koliko je vremena bilo potrebno za ispis svih generiranih izvlačenja:

A screenshot of the Python 3.5.0 Shell window. The window title is "Python 3.5.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:
17 14 19 20
16 14 18 17
20 12 17 5
8 15 4 7
6 20 13 14
20 19 1 18
15 10 5 20
Broj izvlačenja: 3867
Proteklo vrijeme (s) : 42.81
>>>
In the bottom right corner of the shell window, there is a status bar with "Ln: 15365 Col: 4".



Loto - trajanje progr. - rješenje

```
from random import sample
from time import time
brojevi = [i for i in range(1, 21)]
br = 0
ul = input('Lorina kombinacija - 4 broja (1-20): ')
lora = ul.split()
for i in range(len(lora)):
    lora[i]=int(lora[i])
start = time()
while True:
    komb = sample(brojevi, 4)
    print()
    for i in komb:
        print(i, end=' ')
    br += 1
    lora.sort()
    komb.sort()
    if lora == komb:
        print('\nBroj izvlačenja:', br)
        break
stop = time()
print('Proteklo vrijeme (s):', round(stop-start, 2))
```

 Loto-time.py

početak i kraj mjerjenja vremena

Zadatak: Križić - kružić



- Marica i Ivica igraju **križić-kružić** na ploči dimenzija 4x4. Marica je kružić (O) i uvijek igra posljednja. Međutim, nikad nije sigurna hoće li tim zadnjim potezom pobijediti.
- Treba napraviti program koji će pomoći Marici odrediti da li je odigravanjem zadnjeg kružića pobijedila!

Napomena: ne
treba
provjeravati
dijagonale

O	X	O	X
O	O		O
X	O	X	X
X	X	O	X

hoće li
Marica
pobijediti?

Zadatak: Križić - kružić



Ulazni podatci:

- Raspored na ploči - 4 znaka u 4 reda. Koriste se velika slova 'O' i 'X', kao i znak '_' za prazno mjesto.

Izlazni podatak:

- Ispisati DA ili NE (pobjeđuje li Marica)

<u>Ulaz</u>	<u>Izlaz</u>
_xoo	DA
oxxo	
oxxx	
oxxx	
<u>Ulaz</u>	<u>Izlaz</u>
oxox	NE
xx_x	
000x	
oxxo	

<u>Ulaz</u>	<u>Izlaz</u>
00xx	
xx00	
0_0x	
0xxx	
<u>Ulaz</u>	<u>Izlaz</u>
xxox	DA
0xxx	
00_0	
oxox	



Križić - kružić - rješenje



```
odg = 'NE'
ul = []
for i in range(4):
    ul.append(input('Unesi 4 znaka: '))
    if '_' in ul[i]:
        r = i
        s = ul[i].index('_')
if ul[r].count('0') == 3:
    odg = 'DA'
if (ul[0][s]+ul[1][s]+ul[2][s]+ul[3][s]).count('0')==3:
    odg = 'DA'

print(odg)
```



krizic-kruzic.py

Zadatak: Prosti brojevi



- Učitati prirodni broj N
- Provjeriti i ispisati da li su prosti sljedeći brojevi:

- $N - 1$

- N

- $N + 1$

<u>Ulag</u>	<u>Izlag</u>
<u>6</u>	5 je prost 6 nije prost 7 je prost
<u>150</u>	149 je prost 150 nije prost 151 je prost
<u>50</u>	49 je prost 50 nije prost 51 nije prost

Vrijeme



Prosti brojevi - rješenje



```
from math import sqrt, ceil
N = int(input('Unesi N: '))

prost = 'je'
for i in range(2, ceil(sqrt(N-1))+1):
    if (N-1) % i == 0:
        prost = 'nije'
        break
print(N-1, prost, 'prost')

... nastavak
```



prosti.py

Prosti brojevi - rješenje



... *nastavak*

```
prost = 'je'  
for i in range(2, ceil(sqrt(N))+1):  
    if N % i == 0:  
        prost = 'nije'  
        break  
print(N, prost, 'prost')
```

```
prost = 'je'  
for i in range(2, ceil(sqrt(N+1))+1):  
    if (N+1) % i == 0:  
        prost = 'nije'  
        break  
print(N+1, prost, 'prost')
```

Problem: ponavljanje koda



```
from math import sqrt, ceil
N = int(input('Unesi N: '))

prost = 'je'
for i in range(2, ceil(sqrt(N-1))+1):
    if (N-1) % i == 0:
        prost = 'nije'
        break
print(N-1, prost, 'prost')

prost = 'je'
for i in range(2, ceil(sqrt(N))+1):
    if N % i == 0:
        prost = 'nije'
        break
print(N, prost, 'prost')

prost = 'je'
for i in range(2, ceil(sqrt(N+1))+1):
    if (N+1) % i == 0:
        prost = 'nije'
        break
print(N+1, prost, 'prost')
```



Definiranje vlastitih funkcija

- Programski kod koji se ponavlja najbolje je napisati kao funkciju.
- U Pythonu, kao i u drugim programskim jezicima, moguće je definirati vlastite funkcije:

```
def ime_funkcije(popis parametara)
    naredba1_1
    ...
    naredba1_n
    return vrijednost
```



Definiranje vlastitih funkcija

```
def ime_funkcije(popis parametara)
    naredba1_1
    ...
    naredba1_n
    return vrijednost
```

- Parametri su vrijednosti koje se predaju kod poziva funkcije. Funkcija ne mora imati ulazne parametre - u tom slučaju se nakon imena funkcije pišu prazne zagrade () .
- Naredbom return funkcija vraća jednu ili više vrijednosti. Ako funkcija ne vraća vrijednost, onda se iza naredbe return ništa ne piše.



Definiranje vlastitih funkcija

- Primjer: kako se često ponavlja kod za unos cijelih brojeva, onda ćemo napraviti odgovarajuću funkciju.
- Funkciju je moguće definirati i u *Idle* sučelju:

```
>>> def unos_cijeli():
    n = int(input('Unesi cijeli broj: '))
    return n
```

ključna riječ

vrijednost koju funkcija vraća



Definiranje vlastitih funkcija

The screenshot shows the Python 3.5.0 Shell window. The code in the shell is:

```
>>> def unos_cijeli():
    n = int(input('Unesi cijeli broj: '))
    return n
```

Annotations in orange:

- An arrow points from the word "def" to the text "ključna riječ".
- An arrow points from the word "return" to the text "vrijednost koju funkcija vraća".

- Funkcija se zove **unos_cijeli** i nema ulaznih parametara.
- Funkcija **unos_cijeli** vraća cijeli broj koji je unešen pomoću tipkovnice.



Definiranje vlastitih funkcija

□ Poziv funkcije **unos_cijeli**:

The screenshot shows a Python 3.5.0 Shell window. The code entered is:

```
>>> a = unos_cijeli()
Unesi cijeli broj: 5
>>> b = unos_cijeli()
Unesi cijeli broj: -17
>>> print(a, b)
5 -17
>>>
```

An orange arrow points from the text "poziv funkcije" to the opening parenthesis of the first function call "unos_cijeli()".

- Varijabla **a** pokazuje na prvu vrijednost koju vraća funkcija, a varijabla **b** na drugu vrijednost.
- Uočiti da se u definiciji funkcije koristila varijabla **n**.



Definiranje vlastitih funkcija

- Sljedeća funkcija provjeravat će je li broj **paran**:

```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
>>> def paran(x):
    if x%2 == 0:
        print('Broj {0} je paran'.format(x))
    else:
        print('Broj {0} je neparan'.format(x))
return

Ln: 40 Col: 4
```

- Funkcija se zove **paran** i ima jedan ulazni parametar **x** - broj za kojeg će se provjeriti da li paran.
- Funkcija **paran** ne vraća vrijednost u program iz kojeg je pozvana, već samo ispiše poruku.



Definiranje vlastitih funkcija

□ Poziv funkcije **paran**:

The screenshot shows a Python 3.5.0 Shell window. The user has defined a function named 'paran' which prints whether a number is even or odd. The user then calls 'paran' with various arguments, including a single integer and a range of integers, demonstrating how the function handles different inputs. A red error message is shown when the user tries to call 'paran' with two arguments, illustrating that the function only expects one argument.

```
>>> paran(8)
Broj 8 je paran
>>> a = 11
>>> paran(a * 2 - 1)
Broj 21 je neparan
>>> paran(15, 20)
Traceback (most recent call last):
  File "<pyshell#35>", line 1, in <module>
    paran(15, 20)
TypeError: paran() takes 1 positional argument but 2 were given
Ln: 69 Col: 4
```

- U slučaju neispravnog broja parametara dolazi do **pogreške**.



Definiranje vlastitih funkcija

- Jedna funkcija može biti parametar u pozivu druge funkcije (naravno ako odgovara tip podataka koji funkcija vraća):

The screenshot shows a Python 3.5.0 Shell window. The code input is:

```
>>> for br in range(1, 4):
    paran(unos_cijeli())
```

The output shows the execution of the code, which prints two numbers and their classification as even or odd:

```
Unesi cijeli broj: 20
Broj 20 je paran
Unesi cijeli broj: 123456
Broj 123456 je paran
Unesi cijeli broj: 571
Broj 571 je neparan
>>>
```

In the bottom right corner of the shell window, there is a status bar with "Ln: 117 Col: 4".

Zadatak: Prosti brojevi



- Učitati prirodni broj N
- Provjeriti i ispisati da li su prosti sljedeći brojevi:
 - $N - 1$
 - N
 - $N + 1$
- Za provjeru da li je broj prost napisati funkciju **prost(x)!**

<u>Ulaz</u> 6	<u>Izlaz</u> 5 je prost 6 nije prost 7 je prost
<u>Ulaz</u> 150	<u>Izlaz</u> 149 je prost 150 nije prost 151 je prost
<u>Ulaz</u> 50	<u>Izlaz</u> 49 je prost 50 nije prost 51 nije prost

Prosti brojevi - rješenje



```
from math import sqrt, ceil
```

```
def prost(x):
    prost = 'je'
    for i in range(2, ceil(sqrt(x))+1):
        if x % i == 0:
            prost = 'nije'
            break
    return prost
```

```
N = int(input('Unesi N: '))
print(N-1, prost(N-1), 'prost')
print(N, prost(N), 'prost')
print(N+1, prost(N+1), 'prost')
```

} funkcija
} glavni program

Prosti-fun.py



Glavni program kao funkcija



```
from math import sqrt, ceil  
  
def prost(x):  
    prost = 'je'  
    for i in range(2, ceil(sqrt(x))+1):  
        if x % i == 0:  
            prost = 'nije'  
            break  
    return prost  
  
def main():  
    N = int(input('Unesi N: '))  
    print(N-1, prost(N-1), 'prost')  
    print(N, prost(N), 'prost')  
    print(N+1, prost(N+1), 'prost')  
  
main() ← pokretanje glavnog programa
```

Prosti-main.py

funkcija

glavni
program
kao
funkcija



Glavni program kao funkcija

- Dodatna prednost ovakvog pristupa je što se sada u ***Idle*** sučelju program može više puta pokretati pozivom funkcije **main()**:

The screenshot shows a window titled "Python 3.4.2 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Windows, and Help. The main area displays the following text:

```
>>> main()
Unesi N: 60
59 je prost
60 nije prost
61 je prost
>>> main()
Unesi N: 1000
999 nije prost
1000 nije prost
1001 nije prost
>>>
```

In the bottom right corner of the shell window, there is a status bar with "Ln: 144 Col: 4".

Zadatak: Inicijali



- Napisati **funkciju** **inic(a)** koja će za bilo koji broj imena/prezimena vraćati **inicijale**.
- U glavnom programu ispisati inicijale za tri osobe s različitim brojem imena/prezimena – u skladu s testnim podatcima:

<u>Ulaz</u>	<u>Izlaz</u>	Vrijeme
Marko Karatomislavić Ana Iva Lovrić Tea Ita Anić Perić	M.K. A.I.L. T.I.A.P.	
Luka Modrić Pave Župan Rusković Anna Maria Smith Wesson	L.M. P.Ž.R. A.M.S.W.	

Inicijali - rješenje



```
def inic(a):
    rez = ''
    lst = a.split()
    for i in lst:
        rez = rez + i[0] + '.'
    return rez

def main():
    ul = input('Ime i prezime: ')
    print(inic(ul))
    ul = input('Ime i dva prezimena: ')
    print(inic(ul))
    ul = input('Dva imena i dva prezimena: ')
    print(inic(ul))

main() ← pokretanje glavnog programa
```

The code is contained within a yellow box. A red bracket on the right side groups the first two code blocks under the label "funkcija". Another red bracket on the right side groups the last three code blocks under the label "glavni program kao funkcija". A red arrow points from the word "main()" to the text "pokretanje glavnog programa" at the bottom.

iniciali.py

funkcija

glavni program kao funkcija

pokretanje glavnog programa

Zadatak: Dužina riječi



- Napisati funkciju **len_hr(s)** koja ispravno broji dužinu riječi na hrvatskom jeziku!
- U glavnom programu u petlji unositi riječi i ispisivati dužinu unešene riječi - dok se ne unese prazni string.

<u>Ulas</u>	<u>Izlaz</u>
Dživo	4
Programiranje	12
Ljubav	5
Njegovanje	8
Polje	4
Ljuljanje	6



Dužina riječi - rješenje



```
def len_hr(s):
    lj = s.upper().count('LJ')
    nj = s.upper().count('NJ')
    dz = s.upper().count('DŽ')
    return len(s) - lj - nj - dz } funkcija

def main():
    ul = '?'
    while ul != '':
        ul = input('Unesi riječ: ')
        print('Dužina riječi:', len_hr(ul)) } glavni
                                                program
                                                kao
                                                funkcija

main() ← pokretanje glavnog programa
```

len_hr.py 

Ne zaboravite!

- Za 21 dan – u subotu **02.04.2016.**
Finale Lige programiranja
- 5./6. razredi** - početak **10:00**
- 7./8. razredi** - početak **10:00**
- 3 zadatka rješavate 60 minuta
- nemojte kasniti!

