

INFORMATIČKI KLUB

FUTURA

LIGA PROGRAMIRANJA



LIGA PROGRAMIRANJA U PYTHONU ZA OSNOVNE ŠKOLE – 6. RADIONICA

Tomo Sjekavica, Informatički klub FUTURA
Dubrovnik, 14. ožujka 2015.



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



Creative Commons



- ☐ **slobodno smijete:**
 - **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
 - **remiksirati** — prerađivati djelo
- ☐ **pod slijedećim uvjetima:**
 - **imenovanje.** Morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
 - **nekomercijalno.** Ovo djelo ne smijete koristiti u komercijalne svrhe.
 - **dijeli pod istim uvjetima.** Ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.

U slučaju daljnjeg 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

- 14.03.2015 – **6. radionica**
- 28.03.2015 – **5. kolo Lige programiranja**
- 28.03.2015 **9:00** – **5./6. razredi**
- 28.03.2015 **10:00** – **7./8. razredi**
- 28.03.2015 **11:00** – predavanje u sklopu svjetskog dana Arduina **#ArduinoD15**
- 28.03.2015 **11:30** – proglašenje najboljih u pojedinačnom i ekipnom poretku
- Web stranica Lige programiranja:

www.futura.com.hr/liga-programiranja-u-pythonu-2014-2015/



Ponavljjanje: String – niz znakova

□ Spajanje nizova znakova

```
>>> ime = 'Pero'
>>> prez = 'Perić'
>>> ucenik = ime + ' ' + prez
>>> print(ucenik)
Pero Perić
```

Koristi se standardni operator za zbrajanje: **+**

Svi operandi su stringovi!

□ Uvišestručenje niza znakova

```
>>> fut = 'Futura'
>>> fut3 = fut * 3
>>> print(fut3)
FuturaFuturaFutura
>>> print(fut3 * 2)
FuturaFuturaFuturaFuturaFuturaFutura
```

Koristi se standardni operator za množenje: *****
-> Drugi operand je cijeli broj!



Ponavljjanje: String – niz znakova

□ Duljina niza znakova

```
>>> fut = 'Informatički klub FUTURA'
>>> len(fut)
24
>>> prazno = ''
>>> len(prazno)
0
>>> prazno = ' '
>>> len(prazno)
1
```

□ Dohvaćanje pojedinih znakova indeksiranjem

```
>>> fut = 'Informatički klub FUTURA'
>>> print(fut[2])
f
>>> print(fut[2:6])
form
>>> print(fut[0], fut[2:6])
I form
>>> print(fut[0] + fut[2:6])
Iform
```



Ponavljjanje: String – niz znakova

□ Neke od raspoloživih metoda za stringove

```
>>> fut = 'Informatički klub FUTURA'
```

```
>>>
```

```
>>> print(fut.upper())  
INFORMATIČKI KLUB FUTURA
```

`ime_stringa.upper()`
vraća kopiju stringa sa
svim velikim slovima

```
>>>
```

```
>>> print(fut.lower())  
informatički klub futura
```

`ime_stringa.lower()`
vraća kopiju stringa sa
svim malim slovima

```
>>>
```

```
>>> print(fut.count('i'))  
2
```

`ime_stringa.count()`
broji koliko se puta
javlja zadani podniz

```
>>> print(fut.upper().count('I'))  
3
```



Ponavljjanje: String – niz znakova

□ Neke od raspoloživih metoda za stringove

```
>>> fut = 'Informatički klub FUTURA'
```

```
>>>
```

```
>>> print(fut.find('mat'))
```

```
5
```

```
>>> print(fut.find('MAT'))
```

```
-1
```

```
>>>
```

```
>>> print(fut.replace('UTURA', 'utura'))
```

```
Informatički klub Futura
```

```
>>>
```

`ime_stringa.find()`
vraća poziciju
prvog pojavljivanja
zadanog podniza,
ili -1 ako podniz
nije pronađen

`ime_stringa.replace()`
vraća kopiju stringa
sa zamijenjenim
podnizom

Zadatak: Inicijali



- Napiši program u kojem se unosi ime i prezime u jednoj liniji, a ispisuju se inicijali unesenog imena i prezimena.

Vrijeme

- **NAPOMENA:** I ime i prezime su jedna riječ. Riješiti bez korištenja funkcije `split`.

- Ulazni podaci:

- ime i prezime

- Izlazni podaci:

- inicijali imena i prezimena

- Testni podaci:

<u>Ulaz</u>
Pero Perić

<u>Izlaz</u>
P. P.

<u>Ulaz</u>
Tomo Sjekavica

<u>Izlaz</u>
T. S.





Inicijali - rješenje

pronalazak pozicije na
kojem se prvi
pojavljuje praznina

```
imeprezime = input('Unesite ime i prezime: ')
```

```
praznina = imeprezime.find(' ')
```

```
inicijali = imeprezime[0] + ' ' +  
            imeprezime[praznina+1] + ' '
```

```
print(inicijali)
```

spajanje
stringova



inicijali.py

```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
>>> ===== RESTART =====
>>>
Unesite ime i prezime: Tomo Sjekavica
T. S.
>>> ===== RESTART =====
>>>
Unesite ime i prezime: Pero Perić
P. P.
>>> |
```

Moduli – zbirke funkcija



- ❑ Modul se prije korištenja mora uvesti s naredbom **import naziv_modula**
- ❑ Funkcija **sqrt** (korijen) iz **math** modula
 $\sqrt{9} = 3 \rightarrow 3 * 3 = 3^2 = 9$

```
>>> 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
```

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
```

funkcija **fabs**
vraća apsolutnu
vrijednost broja

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

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

```
>>> from math import *
```

```
>>> sqrt(9)
```

```
3.0
```

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)
6
>>> ceil(6.9)
7
>>> floor(5.2)
5
>>> floor(6.9)
6
```

Modul math



- Izračunati korijen sljedećeg izraza:

$$\sqrt{\frac{2 + 1}{2} + 2(3 + 1) + \frac{7 + 6}{2}}$$

```
>>> from math import *  
>>> sqrt((2+1)/2 + 2*(3+1) + (7+6)/2)  
4.0
```

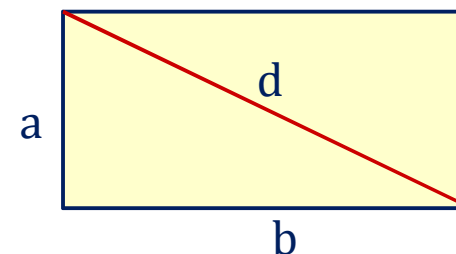
- Zaokružiti na veći i na manji cijeli broj sljedeći izraz: $\frac{2 + 1}{2} + 2(3 + 1)$

```
>>> ceil((2+1)/2 + 2*(3+1))  
10  
>>> floor((2+1)/2 + 2*(3+1))  
9
```

Zadatak: Dijagonala



- Napiši program u kojem se unose duljine stranica a i b pravokutnika, te se računa duljina dijagonale pravokutnika.



$$d = \sqrt{a^2 + b^2}$$

- Ulazni podaci:
 - duljina stranica a i b u jednom retku
- Izlazni podaci:
 - duljina dijagonale

- Testni podaci:

Ulaz
3 4

Izlaz
5.0

Ulaz
5 9

Izlaz
10.295630140987

Vrijeme





Dijagonala - rješenje

uvoz funkcije **sqrt** iz
modula **math**

```
from math import sqrt
stranice = input('Unesi stranice pravokutnika: ')
a, b = stranice.split()
a = int(a)
b = int(b)
d = sqrt(a*a + b*b)
print('Dijagonala pravokutnika je:', d)
```

```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
>>> ===== RESTART =====
>>>
Unesi stranice pravokutnika: 3 4
Dijagonala pravokutnika je: 5.0
>>> ===== RESTART =====
>>>
Unesi stranice pravokutnika: 5 9
Dijagonala pravokutnika je: 10.295630140987
>>>
```

dijagonala.py

Modul random



□ Funkcije za generiranje slučajnih brojeva:

<code>randint(a, b)</code>	vraća slučajni cijeli broj n koji je $a \leq n \leq b$
<code>random()</code>	vraća slučajni realni broj n koji je $0.0 \leq n < 1.0$
<code>uniform(a, b)</code>	vraća slučajni realni broj n koji je $a \leq n \leq b$ ako je $a \leq b$ ili je $b \leq n \leq a$ ako je $b < a$
<code>sample(N, k)</code>	vraća listu od k jedinstvenih elemenata iz seta N

□ <https://docs.python.org/3/library/random.html>

```
>>> from random import *
>>> randint(0, 10)
5
>>> randint(0, 10)
9
```

slučajni cijeli broj
u intervalu $[0, 10]$

Modul random



```
>>> from random import *
```

```
>>> random()
```

slučajni realni broj
u intervalu [0, 1)

```
0.8460300294602602
```

```
>>> random()
```

```
0.9592937131735048
```

```
>>>
```

```
>>> uniform(0, 10)
```

slučajni realni broj
u intervalu [0, 10]

```
1.594305867774457
```

```
>>> uniform(0, 10)
```

definicija liste koja sadrži
20 brojeva od 0 do 19

```
3.394179944212329
```

```
>>>
```

```
>>> brojevi = [i for i in range(20)]
```

```
>>> sample(brojevi, 5)
```

lista od 5 jedinstvenih
brojeva iz liste brojevi

```
[14, 7, 16, 11, 17]
```

Zadatak: Pogodi broj



- Napiši program u kojem se pogađa cijeli broj koje računalo „zamisli” (slučajni broj iz intervala 1 do 20). Da bi se broj prije pogodio ispisivati poruku „Uneseni broj je manji.” ako je unesen manji broj od zamišljenog ili „Uneseni broj je veći.” ako je uneseni veći od zamišljenog broja. Na kraju programa ispisati poruku da je broj pogođen i broj pokušaja.

Vrijeme



```
Python 3.4.1 Shell
File Edit Shell Debug Opti
Python 3.4.1 (v3.4.1:cd
10:38:22) [MSC v.1600 :
Type "copyright", "crea
more information.
>>> =====
>>>
Pogodite broj: 5
Uneseni broj je manji.
Pogodite broj: 10
Uneseni broj je manji.
Pogodite broj: 15
Uneseni broj je veći.
Pogodite broj: 13
Uneseni broj je manji.
Pogodite broj: 14
Pogodili ste broj.
Broj pokušaja: 5
>>> |
```



Pogodi broj - rješenje

```
from random import randint
brojac = 0
broj = 0
zamisljeni = randint(1, 20)
while broj != zamisljeni:
    broj = int(input('Pogodite broj: '))
    brojac = brojac + 1
    if broj < zamisljeni:
        print('Uneseni broj je manji.')
    elif broj > zamisljeni:
        print('Uneseni broj je veći.')
    else:
        print('Pogodili ste broj.')
print('Broj pokušaja:', brojac)
```

uvoz funkcije **randint** iz
modula **random**

generiranje slučajnog broja
iz intervala [1, 20]



pogodiBroj.py

Zadatak: Kockica



- ☐ Ani je dosadilo bacati kockicu, pa joj treba program koji bi simulirao bacanje kockice. Kockica ima standardne vrijednosti od 1 do 6. Ana bi unijela broj bacanja, a program bi simulirao toliko bacanja i ispisao koliko je puta izašao koji broj.

Vrijeme



- ☐ Ulazni podaci:
 - broj bacanja kockice
- ☐ Izlazni podaci:
 - koliko je puta izašao neki broj

```
Python 3.4.1 Shell
File Edit Shell Debug Options
Windows Help
>>> =====
===== RESTART =====
>>>
Unesite broj bacanja: 500
Broj 1: 90
Broj 2: 81
Broj 3: 82
Broj 4: 61
Broj 5: 92
Broj 6: 94
Ln: 253 Col: 0
```



Kockica - rješenje

```
from random import randint
kockica = [0] * (6+1)

n = int(input('Unesite broj bacanja: '))

for i in range(n):
    broj = randint(1, 6)
    kockica[broj] = kockica[broj] + 1

for i in range(1, 6+1):
    print('Broj', str(i)+':', kockica[i])
```



kockica.py

Zadatak: Kockica 2



- ☐ Nadograditi prethodni zadatak tako da se ispisuje i poruka je li program pošten ili nije. Prema Ani program je pošten ako broj koji je izašao najviše puta ne odstupa više od 10% od idealne vrijednosti bacanja za neki broj.
- ☐ Ulazni podaci: **Vrijeme**
 - broj bacanja kockice
- ☐ Izlazni podaci:
 - koliko je puta izašao neki broj
 - poruka je li program pošten



```
Python 3.4.1 Shell
File Edit Shell Debug Options
Windows Help
=====
>>>
Unesite broj bacanja: 500
Broj 1: 78
Broj 2: 62
Broj 3: 105
Broj 4: 90
Broj 5: 90
Broj 6: 75
Program nije pošten!
>>>
```



Kockica 2 - rješenje

```
from random import randint
kockica = [0] * (6+1)
n = int(input('Unesite broj bacanja: '))
for i in range(n):
    broj = randint(1, 6)
    kockica[broj] = kockica[broj] + 1
for i in range(1, 6+1):
    print('Broj', str(i)+':', kockica[i])
prosijek = n / 6
najveci = max(kockica)
if najveci <= (prosijek * 1.1):
    print('Program je pošten!')
else:
    print('Program nije pošten!')
```



kockica2.py

Ne zaboravite!

- ☐ za 15 dana – **u subotu 28.03.2015** –
5. kolo Lige programiranja
- ☐ **5./6. razredi** ekipe od 3 učenika (ili barem 2)
početak **9:00**
- ☐ **7./8. razredi** ekipe od 3 učenika (ili barem 2)
početak **10:00**
- ☐ 3 zadatka rješavate **60 minuta**
- ☐ nemojte kasniti!
- ☐ **11:00** predavanje **Arduino**
- ☐ **11:30** dodjela nagrada i priznanja najboljima

