

INFORMATIČKI KLUB

FUTURA

LIGA PROGRAMIRANJA



python

#2

**LIGA PROGRAMIRANJA U PYTHONU ZA
OSNOVNE ŠKOLE – 1. RADIONICA**

Tomo Sjekavica, Informatički klub FUTURA
Dubrovnik, 7. studenog 2015.



Dubrovnik

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

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/>

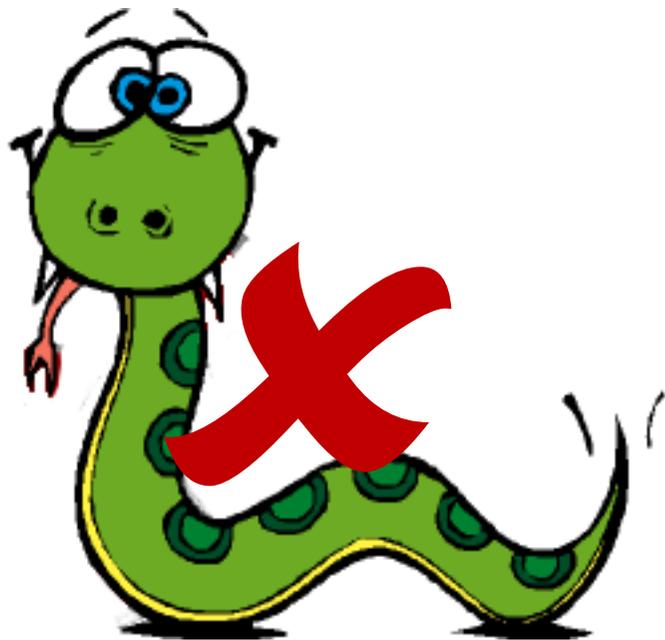
Raspored Lige programiranja

- 07.11.2015. – **1. radionica**
- 21.11.2015. – **1. kolo Lige programiranja**
- 05.12.2015. – 2. radionica
- 19.12.2015. – 2. kolo Lige programiranja
- termini u 2016. godini će biti naknadno određeni
- Web stranica Lige programiranja:
www.futura.com.hr/liga-programiranja-u-pythonu-2015-2016/

Programski jezik Python



□ <https://www.python.org/>



Preuzimanje Pythona 3.5.0



□ <https://www.python.org/downloads/>

Python PSF Docs PyPI Jobs Community

python™ Search GO Socialize Sign In

About Downloads Documentation Community Success Stories News Events

Download the latest version for Windows

[Download Python 3.5.0](#) [Download Python 2.7.10](#)

Wondering which version to use? [Here's more about the difference between Python 2 and 3.](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)

Want to help test development versions of Python? [Pre-releases](#)

Looking for a specific release?
Python releases by version number:

Release version	Release date		Click for more
Python 3.5.0	2015-09-13	Download	Release Notes
Python 2.7.10	2015-05-23	Download	Release Notes

Instalacija Pythona 3.5.0



Python 3.5.0 (32-bit) Setup

Install Python 3.5.0 (32-bit)

Select **Install Now** to install Python with default settings, or choose **Customize** to enable or disable features.

Install Now
C:\Users\Korisnik\AppData\Local\Programs\Python\Python35-32

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

Install launcher for all users (recommended)
 Add Python 3.5 to PATH

python for windows

Python 3.5.0 (32-bit) Setup

Setup Progress

Installing:

C Runtime Update (KB2999226)

python for windows

Python 3.5.0 (32-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

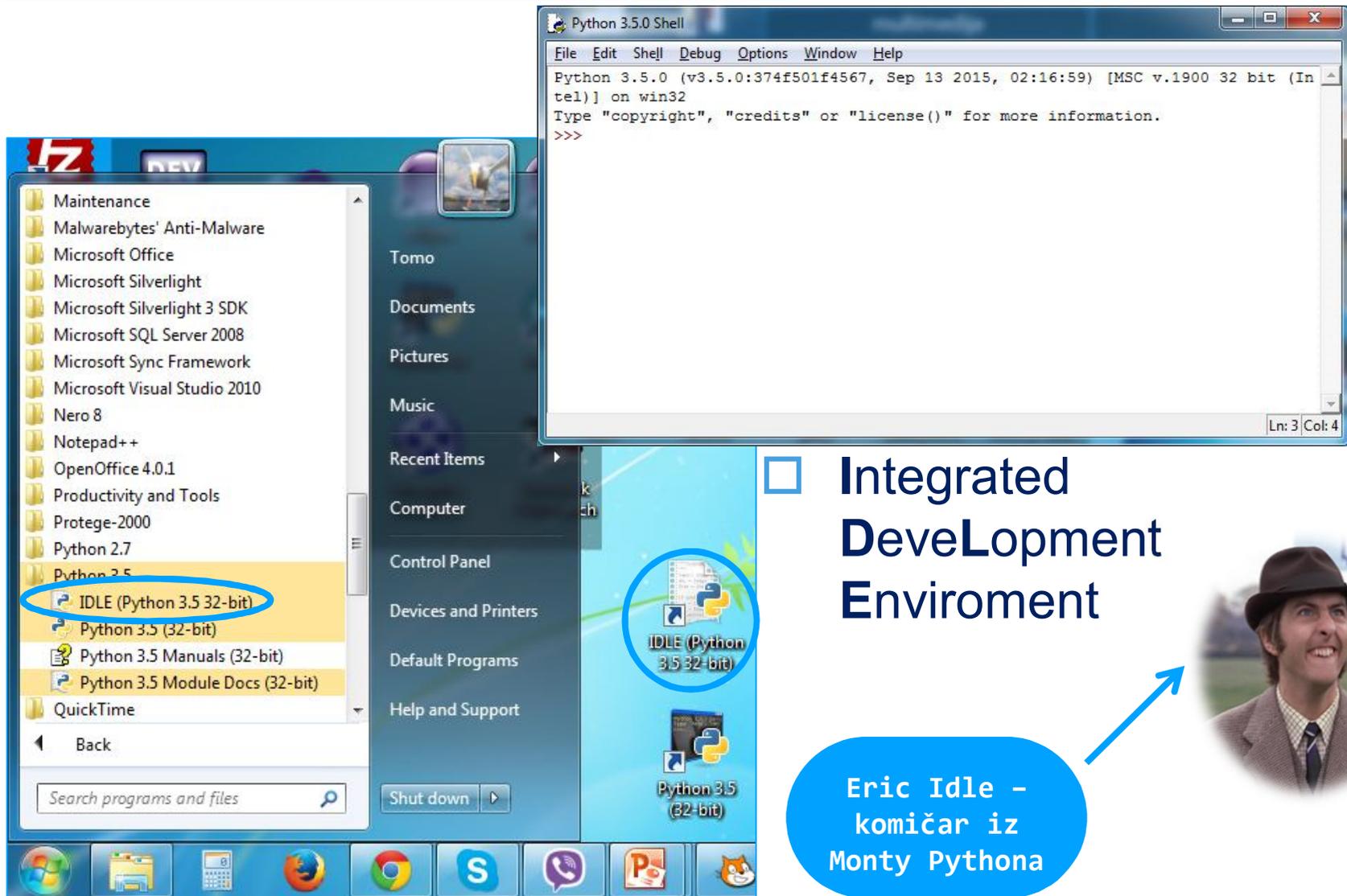
New to Python? Start with the [online tutorial](#) and [documentation](#).

See [what's new](#) in this release.

python for windows

Close

Pokretanje Python 3.5 IDLE



The image shows a Windows Start menu search for 'Python 3.5'. The search results list several items, with 'IDLE (Python 3.5 32-bit)' highlighted in blue. An inset window titled 'Python 3.5 Shell' is shown, displaying the Python 3.5.0 version information and the prompt '>>>'.

Integrated
DeveLopment
Enviroment



Eric Idle -
komičar iz
Monty Pythona

Osnovni tipovi podataka u Pythonu



- **int** – cijeli broj
- **float** – broj s pomičnom točkom
- **str** – niz znakova (string)
- **bool** – logički tip podatka

Cijeli i brojevi s pomičnom točkom

□ Primjeri cijelih brojeva

```
>>> 13
13
>>> 10001
10001
```

```
>>> -13
-13
```

```
>>> 
SyntaxError: invalid token
```

GREŠKA: kod Pythona ne možemo unositi vodeće nule, crvenom bojom je ispisana greška, a crvenom bojom pozadine je označen dio koda gdje se dogodila greška

Python unos ili rezultat neke naredbe ispisuje fontom plave boje, pa se lako može prepoznati što smo mi unijeli, a što je Python ispisao

□ Primjeri brojeva s pomičnom točkom

```
>>> 2.3
2.3
>>> 0.00032
0.00032
>>> 1e15
10000000000000000.0
>>> -2.
-2.0
>>> .000023
2.3e-05
>>> 1e16
1e+16
>>> 0.23
0.23
>>> 1e2
100.0
```

Aritmetički operatori



zbrajanje	+
oduzimanje	-
množenje	*
dijeljenje	/
cjelobrojno dijeljenje	//
modulo (ostatak od dijeljenja)	%
potenciranje	**

- Prvenstvo pri izvođenju ima potenciranje, pa nakon toga množenje, dijeljenje, cjelobrojno dijeljenje i modulo, te na kraju zbrajanje i oduzimanje

Nizovi znakova



□ Jednostruki ili dvostruki navodnici

```
>>> 'Python'  
'Python'
```

```
>>> "Python"  
'Python'
```

nizovi znakova su
označeni fontom
zelene boje

□ Ispis dvostrukih navodnika u nizu znakova

```
>>> 'Radionica "Python" za \"osnovne škole\"'  
'Radionica "Python" za "osnovne škole"'
```

□ Ispis jednostrukih navodnika u nizu znakova

```
>>> "Radionica 'Python' za \'osnovne škole\'"  
"Radionica 'Python' za 'osnovne škole'"
```

Nizovi znakova – funkcija print



- Funkcija je definirani skup naredbi
- Opći oblik funkcije u Pythonu

```
naziv_funkcije(parametar1, parametar2, ... , parametarN)
```

- Funkcija može primiti 0, 1 ili više parametara
- Funkcija **print**

```
>>> print()
```

```
>>> print('Python')
```

```
>>> print('Radionica', 'Python', 2014)
```

```
Radionica Python 2014
```

funkcije print kao
parametre može primiti
različite tipove podataka

standardne Python
funkcije su označene
fontom ljubičaste boje

Nizovi znakova – funkcija print



□ Aritmetički izrazi u ispisu

```
>>> print('Zbroj brojeva', 2, 'i', 3, 'je:', 2 + 3)
Zbroj brojeva 2 i 3 je: 5
```

□ Ispis lijevo nakošene crte \

```
>>> print('Nakošena crta - \\.')
Nakošena crta - \.
```

□ Tabulator - \t

```
>>> print('Korištenje\ttabulatora\tu\tPythonu.')
Korištenje      tabulatora      u      Pythonu.
```

□ Prelazak u novi red pri ispisu - \n

```
>>> print('Prelazak\nu novi red u Pythonu.')
Prelazak
u novi red u Pythonu.
```

Varijable

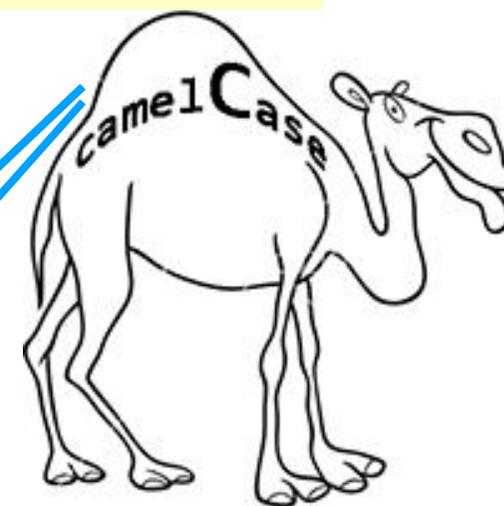


- Varijabla je memorijska lokacija kojoj pristupamo preko njenog naziva, a na njoj je zapisana vrijednost koja se može mijenjati
- Varijabla ima naziv i vrijednost

```
>>> varijabla = 10  
>>>
```

Diagram explaining the code: `varijabla` is the **naziv varijable** (variable name) and `10` is the **vrijednost varijable** (value of the variable).

loši nazivi	dobri nazivi
aaaaa	brojac
abcdefgh	ime_prezime
ahauifhasfuhsaiu	godinaRodjenja
hfjhds3u4444	imeNajPrijatelja



Varijable



- Pravila za imenovanje varijabli:
 - Naziv varijable može sadržavati slova, brojeve i podvlake
 - Naziv varijable ne smije počinjati s brojem
 - Naziv varijable ne smiju biti ključne riječi za koje su rezervirani nazivi, kao što **bool**, **True**, **False**, ...
 - Naziv varijable smije sadržavati naše znakove (čćžšđČĆŽŠĐ), ali se to **nikako ne preporuča**
 - Python razlikuje velika i mala slova, pa su **x** i **X** dvije različite varijable

Pridruživanje vrijednosti varijablama

□ Znak pridruživanja =

```
>>> x = 10
>>> print(x)
10
>>> x = x + 20
>>> print('x =', x)
x = 30
>>> y = -2.3
>>> print(y)
-2.3
>>> y = y * 2
>>> print('y =', y)
y = -4.6
```

□ U varijable se mogu spremiti i nizovi znakova

```
>>> godina = 2015
>>> radionica = 'Liga programiranja u Pythonu'
>>> print('Radionice', radionica, godina, 'oš')
Radionice Liga programiranja u Pythonu 2015 oš
```

Unos s tipkovnice



- ❑ Funkcija **input**
- ❑ Pomoću funkcije **input** unesite vaše ime s tipkovnice, spremite ga u varijablu **ime**, te nakon toga ispišite vrijednost varijable **ime**.

```
>>> ime = input('Unesi svoje ime: ')
Unesite vaše ime: Tomo
>>> print('Uneseno ime je:', ime)
Uneseno ime je: Tomo
```

- ❑ Funkcija **input** sve što se unese s tipkovnice sprema kao niz znakova

Unos s tipkovnice



□ Primjer funkcije `input` s cijelim brojem

```
>>> broj = input('Unesi cijeli broj: ')\nUnesi cijeli broj: 10
```

```
>>> broj + 10
```

```
Traceback (most recent call last):  
  File "<pyshell#6>", line 1, in <module>  
    broj+10
```

```
TypeError: Can't convert 'int' object to str implicitly
```

GREŠKA: broj 10 unesen s tipkovnice je spremljen kao niz znakova

□ Funkcija `int` – pretvara u cijeli broj

```
>>> broj = input('Unesi cijeli broj: ')\nUnesi cijeli broj: 10
```

```
>>> broj = int(broj)
```

```
>>> broj + 10
```

```
20
```

skraćeno se može pisati:
`broj = int(input('Unesi cijeli broj: '))`

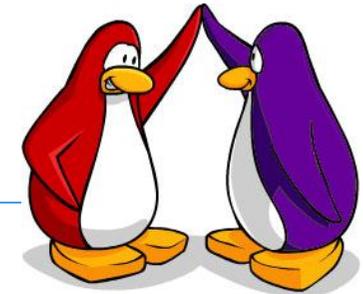
za pretvorbu u broj s pomičnom točkom koristi se funkcija `float`

Program



- Naredbe smo dosad unosili i odmah pokretali u Python IDLE-u
- Što će se dogoditi ako zatvorimo Python IDLE?
- Izgubili smo sve naredbe koje smo unosili
- Program je skup naredbi čijim se izvršenjem obavlja neki posao
- Naredbe možemo spremiti kao poseban program, pa taj program možemo naknadno ažurirati i pokretati

Prvi Python program



The image shows a screenshot of the Python 3.5.0 Shell and the Python IDLE editor. The Python 3.5.0 Shell window is open, showing the File menu with options like New File, Open, Save, and Exit. The Python IDLE editor window is open, showing a code editor with the following code:

```
#Jednolinijski komentari
#Ovo je moj prvi Python program.

"""
Komentar u više linija.
Mogu se koristiti i jednostruki navodnici.
Ovo je moj prvi Python progr.
"""

'Ovo je također jednolinijski komentar.'
'Ovo je moj prvi Python program.'

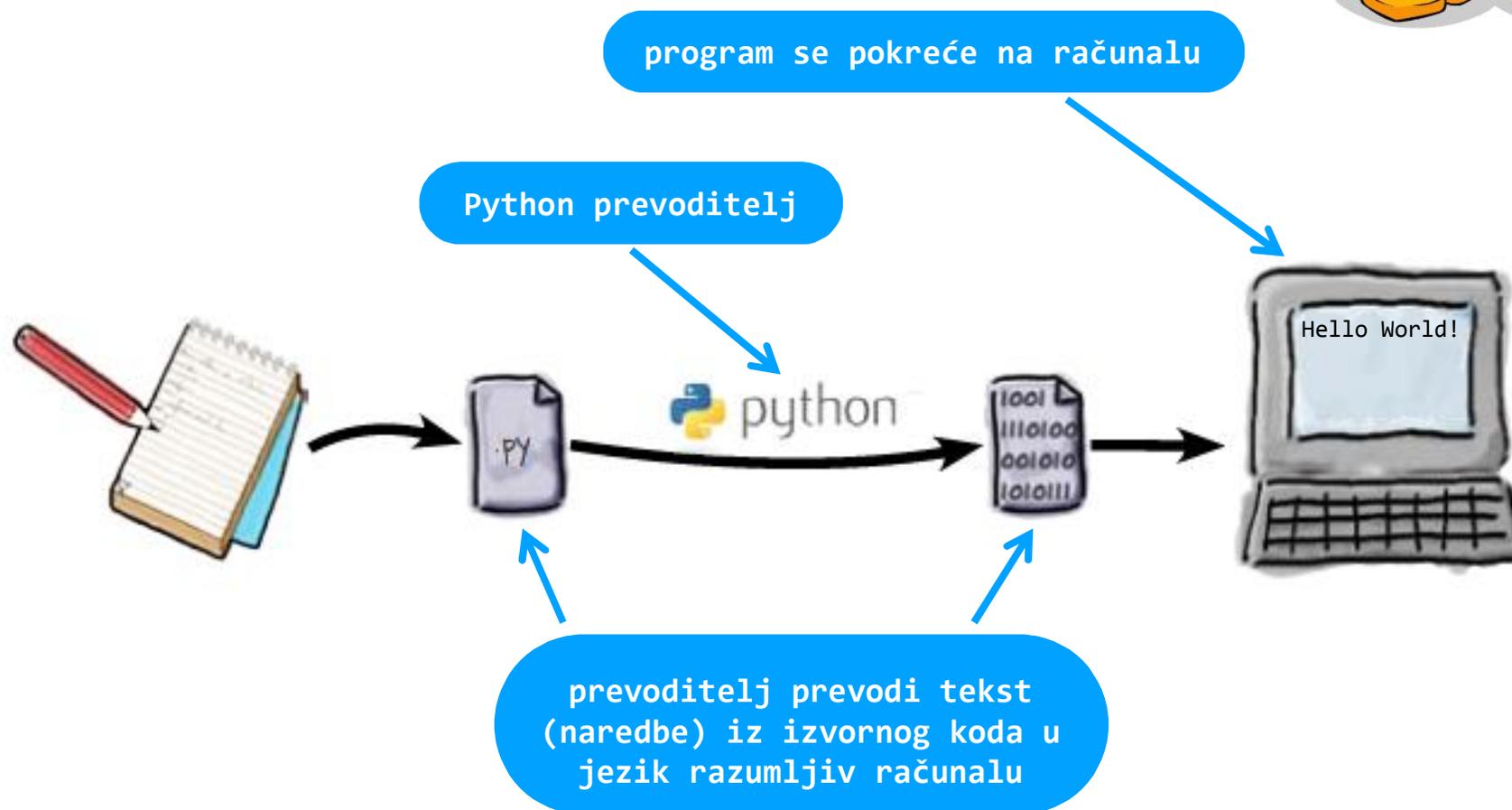
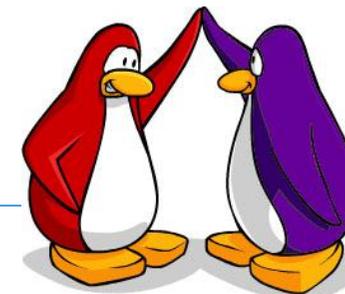
print('Hello World!')
```

The code editor window has a menu bar with File, Edit, Format, Run, Options, Windows, and Help. The code is annotated with blue callouts:

- Python IDLE**: Points to the Python 3.5.0 Shell window.
- Python IDLE editor**: Points to the code editor window.
- komentari**: Points to the multi-line comment block in the code.
- ekstenzija .py**: Points to the file extension in the Save As dialog.
- unos naziva programa**: Points to the file name input field in the Save As dialog.
- spremanje programa**: Points to the Save button in the Save As dialog.
- kod programa**: Points to the code in the editor.

The Save As dialog is open, showing the file name `helloworld.py` and the save as type `Python files (*.py;*.pyw)`. The dialog also shows the file name input field and the Save button.

Prevođenje programa



Prilagođeno iz: P. Barry & D. Griffiths, Head First Programming, O'Reilly, 2009

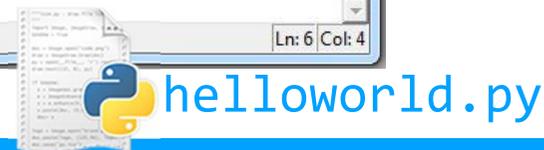
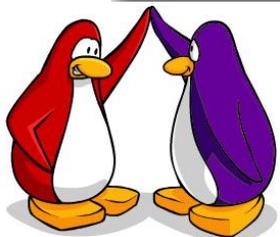
Pokretanje programa u Python IDLE-u

The screenshot shows the Python IDLE environment. The main window displays a Python script named `helloworld.py` with the following content:

```
#Jednolinijski  
#Ovo je moj pr  
  
"""  
Komentar u vi  
Mogu se koristiti i jednorazni navodnici.  
Ovo je moj prvi Python progr.  
"""  
  
'Ovo je također jednolinijski komentar.'  
"Ovo je moj prvi Python program."  
  
print('Hello World!')
```

The `Run` menu is open, showing the `Run Module F5` option. A blue callout bubble points to this option with the text "pokretanje programa". Another blue callout bubble points to the `F5` key with the text "prečac na tipkovnici: funkcijska tipka F5". The `Python 3.4.1 Shell` window shows the output of the script:

```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (In  
tel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> ===== RESTART =====>>>  
>>>  
Hello World!  
>>> |
```



Program: Unos imena



- Naredbe za unos imena i ispis imena na ekran iz primjera spremite u program `ime.py`, te pokrenite program u Python IDLE-u.

```
ime = input('Unesite vaše ime: ')  
print('Uneseno ime je: ', ime)
```

unosi se ime
s tipkovnice

```
File Edit Shell Debug Options Windows Help  
Python 3.4.1 (vs.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> ===== RESTART =====  
>>>  
Unesite vaše ime: Tomo  
Uneseno ime je: Tomo  
>>>
```

ispis unesenog imena



Zadatak: Vratar

775 minuta = 8 utakmica,
1 poluvrijeme i 10 minuta



- Vratar Lovre Kalinić nedavno je postao novi rekorder Hajduka u HNL-u po broju minuta bez primljenog gola. Njegov rekord iznosi 775 minuta. Novinare zanima koliko bi to bilo utakmica, poluvremena i minuta.
- Nogometna utakmica traje 90 minuta, a podijeljena je na 2 poluvremena po 45 minuta.
- Napisati program u kojem se unosi ime i prezime vratara, te njegov rekord u minutama.
- Izračunati i ispisati broj utakmica, poluvremena i minuta bez primljenog gola.

Vrijeme





Zadatak: Vratari - rješenje

```
vratar = input('Unesite ime i prezime vratara: ')\nminute = int(input('Unesite broj minuta: '))\npoluvrijeme = minute // 45\nminute = minute % 45\nutakmica = poluvrijeme // 2\npoluvrijeme = poluvrijeme % 2\nprint(vratar, 'nije primio gol', utakmica,\n      'utakmica,', poluvrijeme, 'poluvrijeme i', minute,\n      'minuta.')
```



```
Python 3.5.0 Shell\nFile Edit Shell Debug Options Window Help\nRESTART: D:/Google disk/futura/radionice/Liga_programiranja_2015/01-\nprimjeri_zadaci/vratar.py\nUnesite ime i prezime vratara: Lovre Kalinić\nUnesite broj minuta: 775\nLovre Kalinić nije primio gol 8 utakmica, 1 poluvrijeme i 10 minuta.\n>>> |
```

Relacijski operatori



veće od	>
manje od	<
veće od ili jednako	>=
manje od ili jednako	<=
jednako	==
nije jednako	!=

- Relacijski operatori uspoređuju dva operanda. Rezultat usporedbe ima vrijednosti **True** ili **False** (Istina ili Laž).

Relacijski operatori



□ Provjeriti kako djeluju operatori:

```
>>> 3 > 2
True
>>> 3 < 2
False
>>> 3 >= 2
True
>>> 3 <= 2
False
>>> 3 == 2
False
>>> 3 != 2
True
```

```
>>> a = 2
>>> b = 7
>>> b > a
True
>>> b + 1 == a * 4
True
>>> b / a != b // a
True
>>> (a + b) ** 2 <= 10 * a
False
```

prvo se izračunaju aritmetički izrazi s lijeve i desne strane, pa se izvršava usporedba

Donošenje odluka u programima

- Izvođenje jedne ili više naredbi na temelju ispitivanja nekog uvjeta:

```
...  
ako je uvjet onda  
    naredba1_1  
    ...  
    naredba1_n  
...
```

- Odabir jedne od dvije mogućnosti (izvršava se samo jedan blok naredbi):

```
...  
ako je uvjet onda  
    naredba1_1  
    ...  
    naredba1_n  
inače  
    naredba2_1  
    ...  
    naredba2_m  
...
```

- Višestruki izbor izvršava se samo jedan od blokova naredbi):

```
...  
ako je uvjet_1 onda  
    blok_naredbi_1  
inače ako je  
uvjet_2 onda  
    blok_naredbi_2  
...  
inače ako je  
uvjet_n onda  
    blok_naredbi_n  
inače  
    blok_naredbi  
...
```

Donošenje odluka u Pythonu

- Izvođenje jedne ili više naredbi na temelju ispitivanja nekog uvjeta:

```
...  
if uvjet:  
    naredba1_1  
    ...  
    naredba1_n  
...
```

- Odabir jedne od dvije mogućnosti (izvršava se samo jedan blok naredbi):

```
...  
if uvjet:  
    naredba1_1  
    ...  
    naredba1_n  
else:  
    naredba2_1  
    ...  
    naredba2_m  
...
```

- Višestruki izbor izvršava se samo jedan od blokova naredbi):

```
...  
if uvjet_1:  
    blok_naredbi_1  
elif uvjet_2:  
    blok_naredbi_2  
...  
elif uvjet_n:  
    blok_naredbi_n  
else:  
    blok_naredbi  
...
```

Primjer: Veći broj v1



...

if uvjet:

→ naredba1_1

→ ...

→ naredba1_n

...

izvršit će se ako je *uvjet* zadovoljen (logički izraz je istinit)

svaki redak koji će se izvesti ako je *uvjet* zadovoljen mora biti uvučen. Najbolje je koristiti tipku TAB

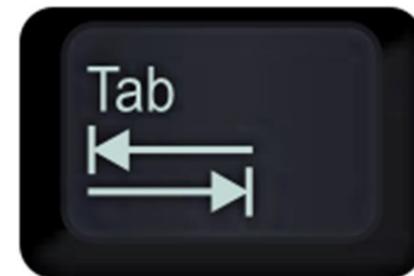
```
A = int(input('Unesi broj A: '))
```

```
B = int(input('Unesi broj B: '))
```

```
if A > B:
```

```
    print('A je veće od B')
```

```
print('Kraj programa!')
```



veci_broj_v1.py

Primjer: Veći broj v2

izvršit će se samo
JEDAN od ova dva
bloka naredbi!



...

if *uvjet*:

→ naredba1_1

→ ...

→ naredba1_n

else:

→ naredba2_1

→ ...

→ naredba2_m

...

izvršit će se ako je *uvjet* zadovoljen
(logički izraz je istinit)

izvršit će se ako *uvjet* nije zadovoljen
(logički izraz je lažan)

```
A = int(input('Unesi broj A: '))  
B = int(input('Unesi broj B: '))
```

```
if A > B:  
    print('A je veće od B')  
else:  
    print('A nije veće od B')  
print('Kraj programa!')
```

Najbolje je
koristiti
tipku TAB



veci_broj_v2.py

Primjer: Veći broj v3



```
...
if uvjet_1:
    blok naredbi_1
elif uvjet_2:
    blok naredbi_2
...
elif uvjet_n:
    blok naredbi_n
else:
    blok naredbi
...
```

izvršit će se
samo JEDAN od
blokova
naredbi!

```
A = int(input('Unesi broj A: '))
B = int(input('Unesi broj B: '))

if A > B:
    print('A je veće od B')
elif A == B:
    print('A i B su jednaki')
else:
    print('A je manje od B')
print('Kraj programa!')
```



veci_broj_v3.py

Logički operatori i logički izrazi



- Što ako je uvjet na temelju kojeg treba donijeti odluku složen?
- Logički operatori:

logička I operacija	<code>and</code>
logička ILI operacija	<code>or</code>
NE operacija (negacija)	<code>not</code>

- Redoslijed izvođenja logičkih operacija:

1.	<code>not</code>
2.	<code>and</code>
3.	<code>or</code>

Logički operatori i logički izrazi



□ Primjeri logičkih operacija:

```
>>> a = 2
>>> b = 3
>>> c = 10
>>> a > b
False
>>>
>>> c > b
True
>>>
>>> a > b and c > b
False
>>>
>>> a > b or c > b
True
```

za logičku **AND** operaciju rezultat će biti **True** (istina) samo ako su oba izraza True (istinita)

za logičku **OR** operaciju rezultat će biti **True** (istina) već ako je jedan od izraza True (istinit)

Redoslijed izvođenja operacija



1.	aritmetički
2.	relacijski
3.	logički

Ako imamo kombinirane aritmetičke, relacijske i logičke operatore, onda je ovo redoslijed izvođenja operacija.

```
>>> a=2
>>> b=3
>>> c=10
>>> a+2*3 >= c or not(a > b) and a*b-2 == c%6
```

True

```
>>>
```

```
>>> (a+2*3 >= c) or (not(a > b) and (a*b-2 == c%6))
```

True

```
>>>
```

Ako ipak nismo posve sigurni u redoslijed operacija onda je najbolje koristiti zagrade!

Zadatak: Skok u dalj



- Na natjecanju u skokovima u dalj organizatori su odlučili podijeliti više zlatnih, srebrnih i brončanih medalja prema sljedećim kriterijima:
 - Brončana medalja za sve koji preskoče između 5,5 i 6,5 metara, uključujući skokove od 5,5 i 6,5 metara
 - Srebrna medalja za sve koji preskoče preko 6,5 metara i manje od 7 metara
 - Zlatna medalja za sve koji preskoče 7 metara i preko
- Ispisati koju je medalju osvojio natjecatelj, te poruku ako nije osvojio medalju.

Vrijeme





Zadatak: Skok u dalj - rješenje

```
skok = float(input('Unesi duljinu skoka: '))
```

```
if skok >= 5.5 and skok <= 6.5:  
    print('Brončana medalja')  
elif skok > 6.5 and skok < 7:  
    print('Srebrna medalja')  
elif skok >= 7:  
    print('Zlatna medalja')  
else:  
    print('Nije osvojena medalja')
```

```
Python 3.5.0 Shell  
File Edit Shell Debug Options Window Help  
mjeri_zadaci/skok_u_dalj.py  
Unesi duljinu skoka: 4  
Nije osvojena medalja  
>>>  
RESTART: D:/Google disk/futura/radi  
onice/Liga_programiranja_2015/01-pri  
mjeri_zadaci/skok_u_dalj.py  
Unesi duljinu skoka: 5.6  
Brončana medalja  
>>>  
RESTART: D:/Google disk/futura/radi  
onice/Liga_programiranja_2015/01-pri  
mjeri_zadaci/skok_u_dalj.py  
Unesi duljinu skoka: 6.7  
Srebrna medalja  
>>>  
RESTART: D:/Google disk/futura/radi  
onice/Liga_programiranja_2015/01-pri  
mjeri_zadaci/skok_u_dalj.py  
Unesi duljinu skoka: 8  
Zlatna medalja  
>>> |  
Ln: 60 Col: 4
```



skok_u_dalj.py

Ponavljanje bloka naredbi



- Ponavljanje bloka naredbi određeni broj puta:

```
...  
za i := 1 do n činiti  
  naredba_1  
  ...  
  naredba_z
```

- **for** petlja

- Uvjetno ponavljanje bloka naredbi:

```
...  
dok je uvjet činiti  
  naredba_1  
  ...  
  naredba_z
```

- **while** petlja

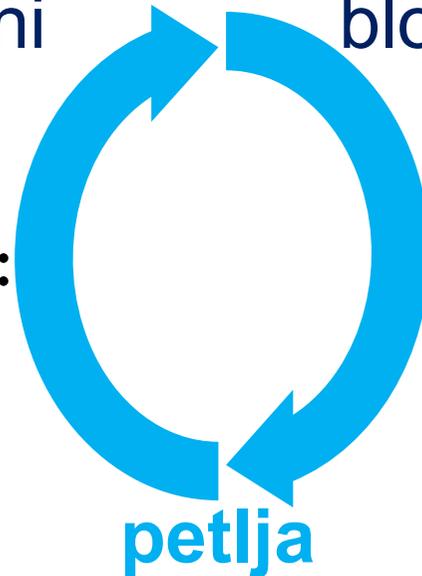
programska petlja

Ponavljanje bloka naredbi Python



- Ponavljanje bloka naredbi određeni broj puta:

```
...  
for i in range(n):  
    naredba_1  
    ...  
    naredba_z  
...
```



- Blok naredbi će se izvesti **n puta**, za vrijednosti varijable **i** od **0** do **n-1**.

- Uvjetno ponavljanje bloka naredbi:

```
...  
while uvjet:  
    naredba_1  
    ...  
    naredba_z  
...
```

- Blok naredbi će se izvoditi dok je **uvjet** ispunjen (daje vrijednost **True**)

Vrste for petlje



□ range(n)

```
Python 3.5.0 ...
File Edit Shell Debug
Options Window Help
>>> for i in range(6):
>>>     print(i)
0
1
2
3
4
5
>>> |
Ln: 82 Col: 4
```

□ Blok naredbi se izvodi **6 puta**, za vrijednosti **i** od **0** do **5**.

□ range(n,m)

```
Python 3.5.0 Shell
File Edit Shell Debug Options
Window Help
>>> for i in range(2,9):
>>>     print(i)
2
3
4
5
6
7
8
>>>
Ln: 82 Col: 0
```

□ Blok naredbi se izvodi **7 puta**, za vrijednosti **i** od **2** do **8**.

□ range(n,m,k)

```
Python 3.5.0 Shell
File Edit Shell Debug Options
Window Help
>>> for i in range(3,23,3):
>>>     print(i)
3
6
9
12
15
18
21
>>>
Ln: 124 Col: 4
```

□ Varijabla **i** mijenja vrijednost od **3** do **22** (tj. $23-1$), s korakom **3**.

Primjer s for petljom: djelitelji



- Unijeti prirodni broj N i ispisati sve njegove djelitelje. Ako broj N nema djelitelja osim 1 i N, ispisati: "N je prosti broj". Inače ispisati: "N je složeni broj".

```
N = int(input('Unesi prirodni broj: '))
brDjel = 0
for i in range(1, N+1):
    if N%i == 0:
        print(i)
        brDjel = brDjel + 1
if brDjel == 2:
    print(N, 'je prosti broj!')
else:
    print(N, 'je složeni broj!')
```

```
Python 3.5.0 Shell
File Edit Shell Debug
RESTART: D:/Google
ce/Liga_programira
zadaci/djelitelji.py
Unesi prirodni broj: 29
1
29 je prosti broj!
>>> |

Python 3.5.0 Shell
File Edit Shell Debug Options
Window Help
RESTART: D:/Google disk/fut
ura/radionice/Liga_programir
anja_2015/01-primjeri_zadaci
/djelitelji.py
Unesi prirodni broj: 35
1
5
7
35
35 je složeni broj!
>>> |
Ln: 138 Col: 4
```



Uvjetno ponavljanje bloka naredbi

□ **while** petlja

```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
>>> i = 10
>>> while i < 20:
>>>     print(i)
>>>     i = i + 1
10
11
12
13
14
15
16
17
18
19
>>>
```

početna vrijednost varijable *i*

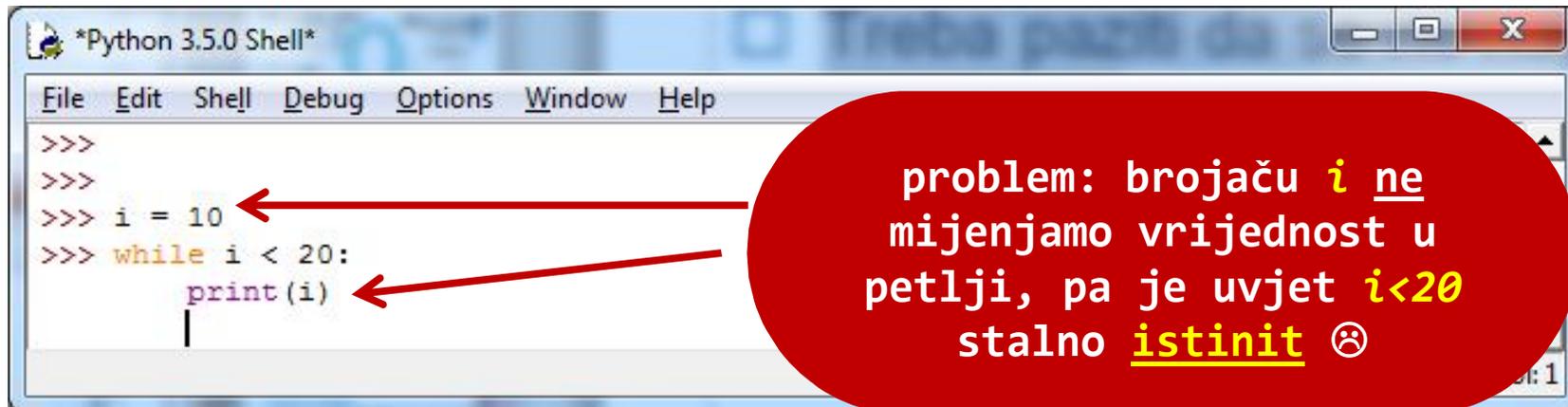
dvije naredbe u bloku će se izvršavati više puta - sve dok je $i < 20$

brojač (ovdje varijabla *i*) mora se prije petlje inicijalizirati, a u petlji povećavati (smanjivati)!

Ln: 19 Col: 4

Uvjetno ponavljanje bloka naredbi

- ❑ Treba paziti da se ne napiše "beskonačna" petlja 😞 :



```
*Python 3.5.0 Shell*
File Edit Shell Debug Options Window Help
>>>
>>>
>>> i = 10
>>> while i < 20:
>>>     print(i)
>>>
```

problem: brojaču *i* ne mijenjamo vrijednost u petlji, pa je uvjet *i < 20* stalno istinit 😞

- ❑ Što će se dogoditi?
- ❑ Napisali smo "beskonačnu" petlju, pa je moramo prekinuti istovremenim pritiskom tipki **Ctrl i C**

Primjer s while petljom: djelitelji



- Unijeti prirodni broj N i ispisati sve njegove djelitelje. Ako broj N nema djelitelja osim 1 i N, ispisati: "N je prosti broj". Inače ispisati: "N je složeni broj".

```
N = int(input('Unesi pr. broj: '))
brDjel = 0
i = 1
while i <= N:
    if N%i == 0:
        print(i)
        brDjel = brDjel + 1
    i = i + 1
if brDjel == 2:
    print(N, 'je prosti broj!')
else:
    print(N, 'je složeni broj!')
```

```
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
RESTART: D:/Google disk/futura/ra
dionice/Liga_programiranja_2015/01
-primjeri_zadaci/djelitelji_while.
py
Unesi prirodni broj: 29
1
29
29 je prosti broj!
>>>
RESTART: D:/Google disk/futura/ra
dionice/Liga_programiranja_2015/01
-primjeri_zadaci/djelitelji_while.
py
Unesi prirodni broj: 35
1
5
7
35
35 je složeni broj!
>>>
```

 [djelitelji_while.py](#)

Zadatak: Bodovi



- Unosi se broj zadataka, pa broj bodova za svaki zadatak. Izračunati ukupan broj bodova i prosjek bodova po zadatku.
- **7./8. razredi:** Svaki zadatak može imati najviše 50 bodova. Iz ukupnog broja bodova i prosjeka bodova po zadatku izbaciti bodove najboljeg i najlošijeg zadatka.

Primjeri testnih podataka

5./6. razredi			7./8. razredi		
ULAZ	ULAZ	ULAZ	ULAZ	ULAZ	ULAZ
5	4	6	5	4	6
10	16	10	10	16	10
15	18	30	15	18	30
20	2	50	20	2	50
10	25	40	10	25	40
15		20	15		20
		16			16
IZLAZ	IZLAZ	IZLAZ	IZLAZ	IZLAZ	IZLAZ
70	61	166	40	34	106
14.0	15.25	27.666	13.333	17.0	26.5

Vrijeme





Zadatak: Bodovi 5./6. – rješenje

```
N = int(input('Unesi broj zadataka: '))
ukupno = 0

for i in range(N):
    bod = int(input('Unesi broj bodova: '))
    ukupno = ukupno + bod

prosjek = ukupno / N

print('Ukupan broj bodova je:', ukupno)
print('Prosjek bodova po zadatku je:', prosjek)
```



bodovi_5-6.py



Zadatak: Bodovi 7./8. – rješenje

```
N = int(input('Unesi broj zadataka: '))
ukupno = 0
najmanji = 50
najveci = 0
for i in range(N):
    bod = int(input('Unesi broj bodova: '))
    ukupno = ukupno + bod
    if bod < najmanji:
        najmanji = bod
    if bod > najveci:
        najveci = bod
ukupno = ukupno - najmanji - najveci
prosjek = ukupno / (N - 2)
print('Ukupan broj bodova je:', ukupno)
print('Prosjek bodova po zadatku je:', prosjek)
```

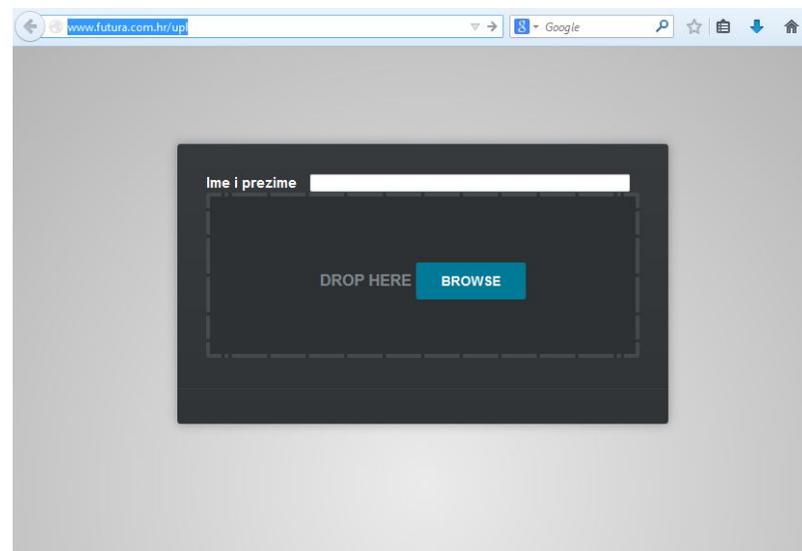
 [bodovi_7-8.py](#)

Slanje programa na natjecanju?



- ❑ Kad idući put bude kolo **Lige programiranja**, bit će potrebno poslati (**upload**) programski kod riješenih zadataka.
- ❑ Link za slanje programa:

www.futura.com.hr/upl



Slanje programa na natjecanju?



1. Upisati ime i prezime

2. Za svaki program:
"drag & drop"
ili
koristiti "browse"

www.futura.com.hr/upl

Ime i prezime Dživo Programić

DROP HERE BROWSE

prosjOcjena.py
0.36 KB ✓

Spremanje datoteke prosjOcjena.py [prosjOcjena.py] u arhivu dzivo programic.zip uspješno !

Ne zaboravite!

- Za 15 dana – u subotu 21.11.2015. –
1. kolo Lige programiranja
- 5./6. razredi** - početak **9:00**
- 7./8. razredi** - početak **10:30**
- 3** zadatka rješavate **75** minuta
- nemojte kasniti!

