INFORMATIČKI KLUB

FUTURA

LIGA PROGRAMIRANJA #2

python

LIGA PROGRAMIRANJA U PYTHONU ZA OSNOVNE ŠKOLE – 1. RADIONICA

Tomo Sjekavica, Informatički klub FUTURA

Dubrovnik, 7. studenog 2015.

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/`

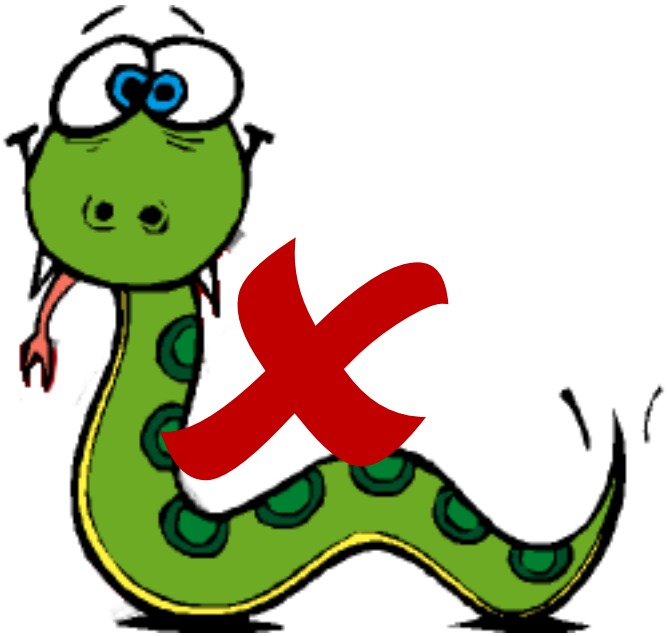
Raspored Lige programiranja

- 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/

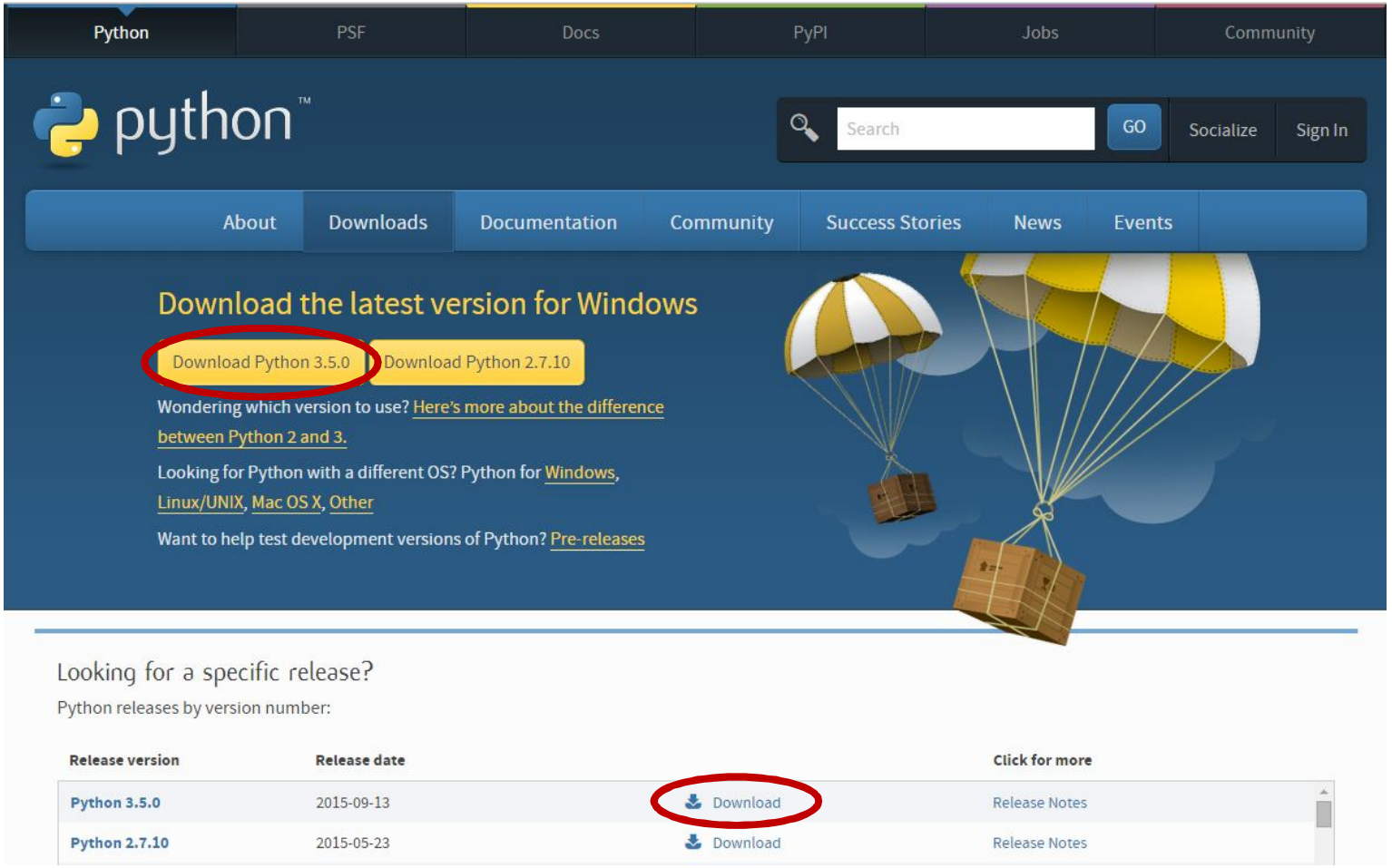
Programski jezik Python

- https://www.python.org/

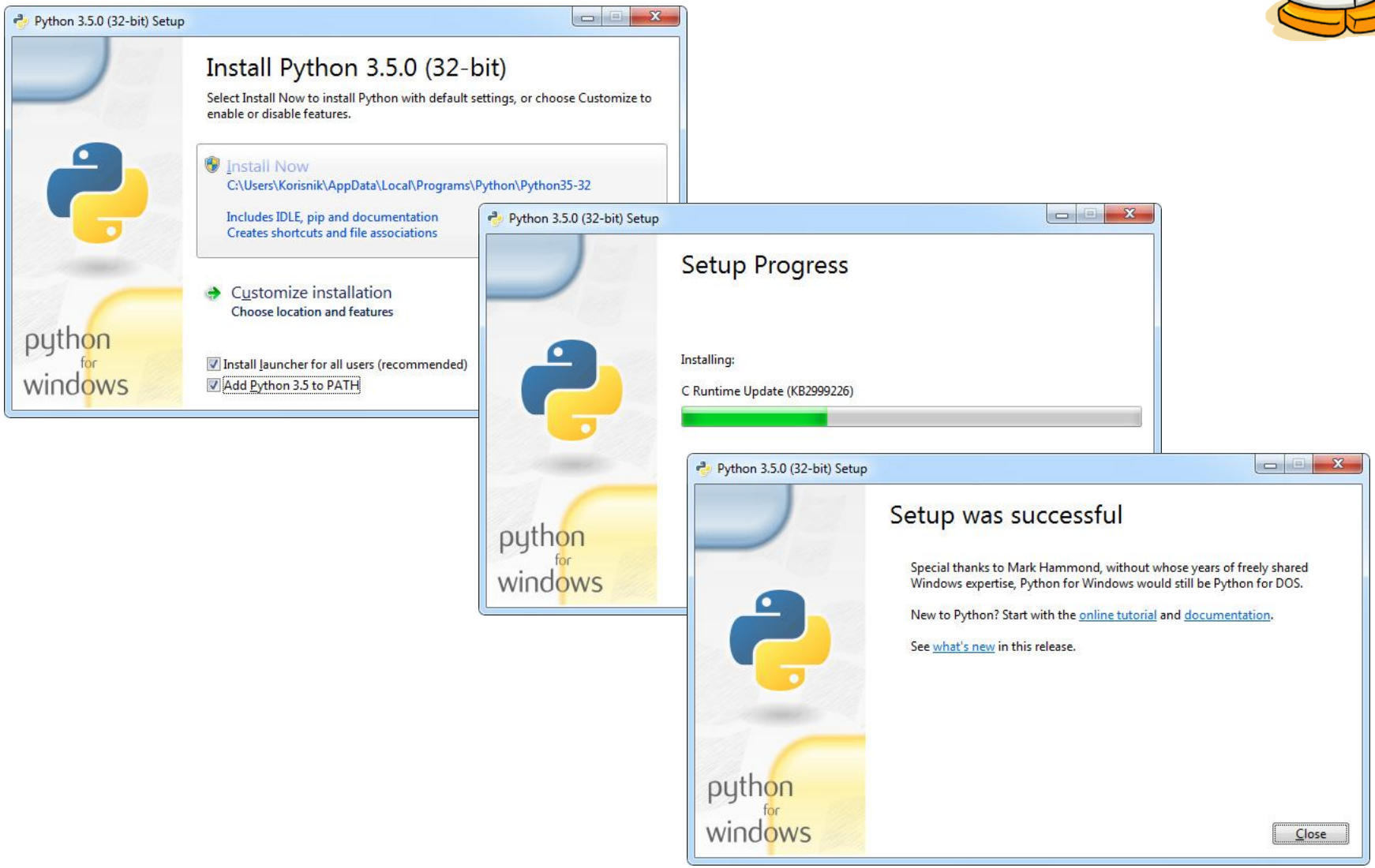


Preuzimanje Pythona 3.5.0

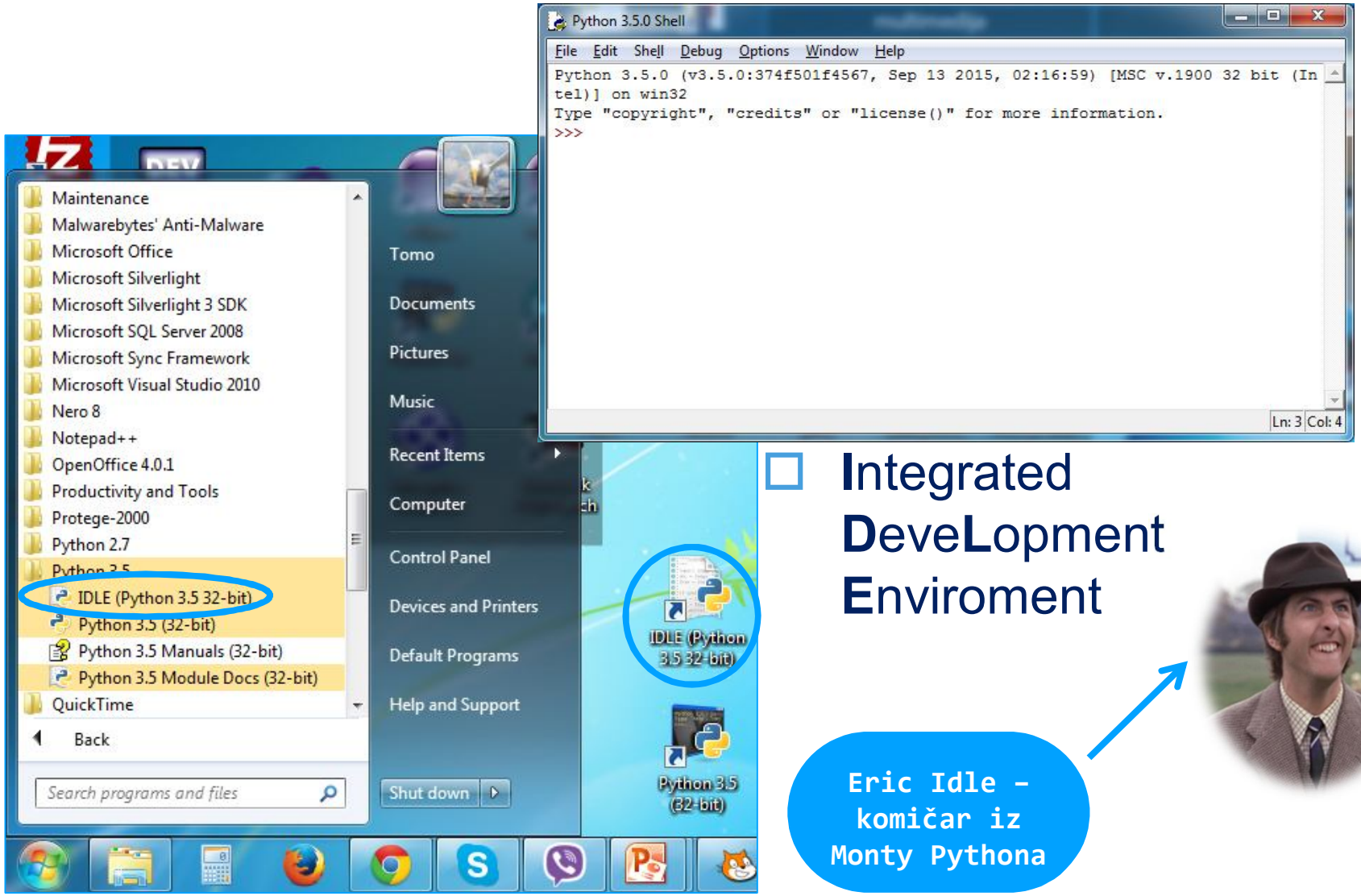
- https://www.python.org/downloads/



Instalacija Pythona 3.5.0



Pokretanje Python 3.5 IDLE



- **I**ntegrated
  Deve**L**opment
  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
>>> 0123
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
1000000000000000.0
```

```
>>> -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

Aritmetički izrazi

Jednostavni aritmetički izrazi

```
>>> 3 + 2
5
>>> 2 / 3
0.6666666666666666
>>> 2 // 3
0
>>> 3 - 4
-1
>>> 2 % 3
2
>>> 2 +
SyntaxError: invalid syntax
>>> 3 * 2
6
>>> 2 ** 3
8
```

Složeni aritmetički izrazi

```
>>> 2 + 2 * 3
8
>>> (2 + 2) * 3
12
>>> (2 + 2) * 3 + 2 / 3
12.666666666666666
```

GREŠKA: ako ne napišemo dobro aritmetički izraz prikazati će nam se greška

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

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

funkcije print kao parametre može primati 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

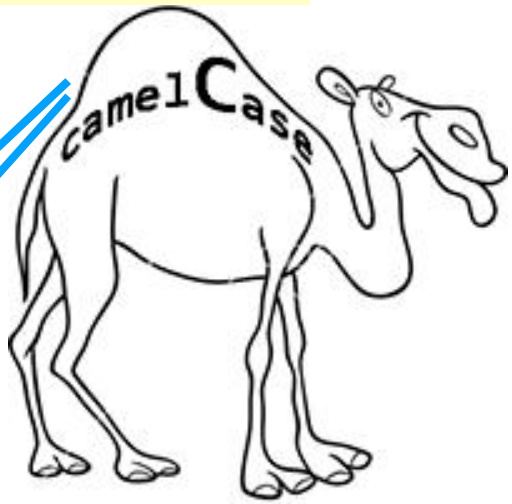
- Varijabla je memorijska lokacija kojoj pristupamo preko njenog naziva, a na njoj je zapisana vrijednost koja se može mijenjati
- Varijabla ima naziv i vrijednost

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

vrijednost varijable

naziv varijable

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 progamiranja 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: ')
Unesi 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: ')
Unesi cijeli broj: 10
>>> broj = int(broj)
>>> broj + 10
20
```

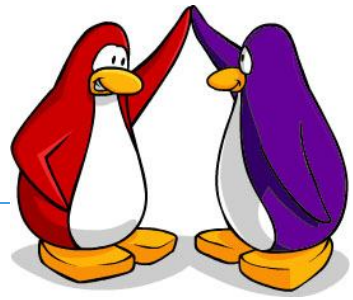
skraćeno se može pisati:
`broj = int(input('Unesi cijeli broj: ')`

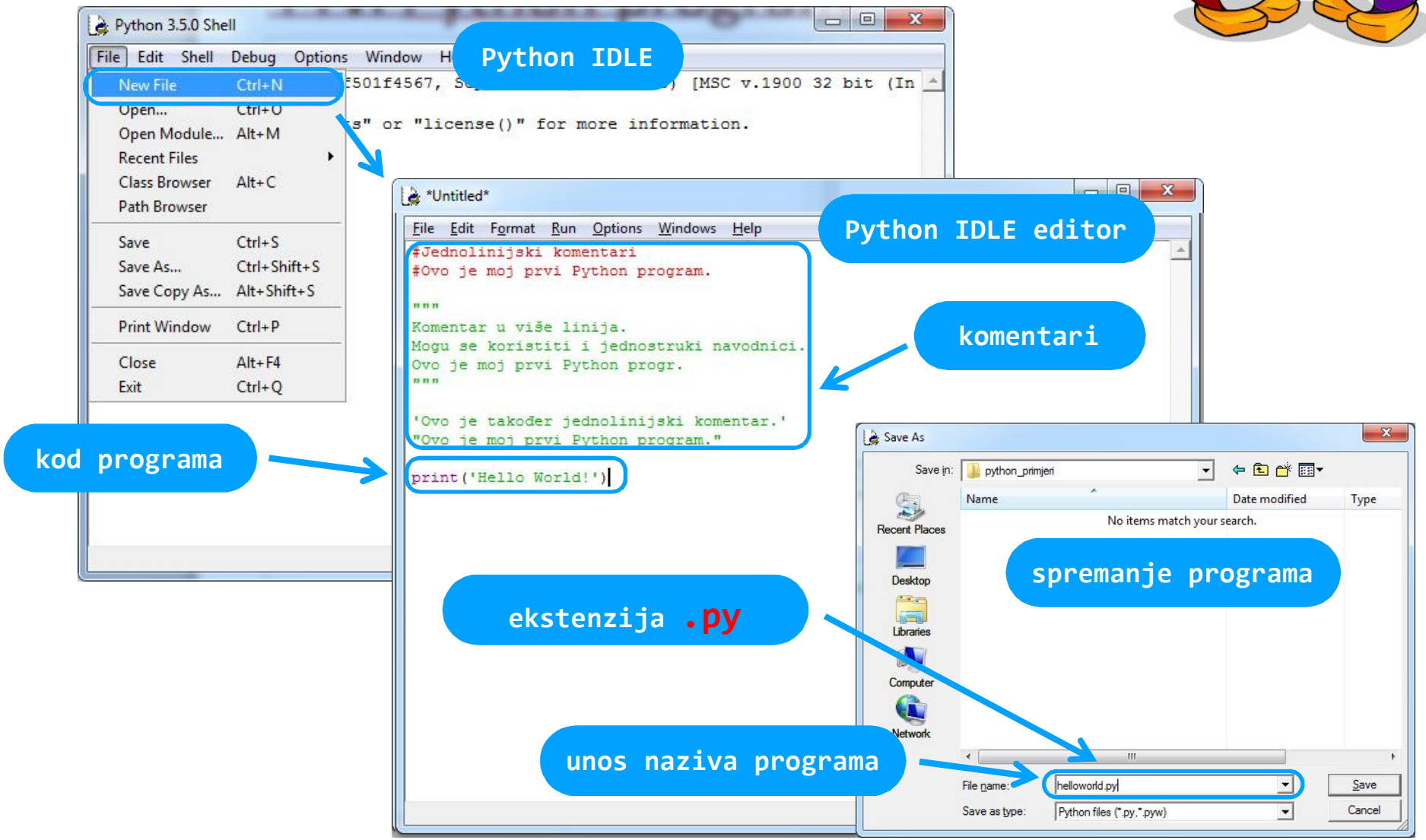
za pretvorbu u broj s pomičnom točkom koristi se funkcija `float`

Program

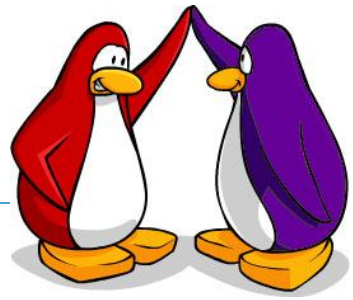
- Naredbe smo dosad unosili i odmah pokretali u Python IDLE-u
- Što će se dogoditi ako zatvorimo Python IDLE?
- Izgubili smo sve naredbe koje smo unosili
- Program je skup naredbi čijim se izvršenjem obavlja neki posao
- Naredbe možemo spremiti kao poseban program, pa taj program možemo naknadno ažurirati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

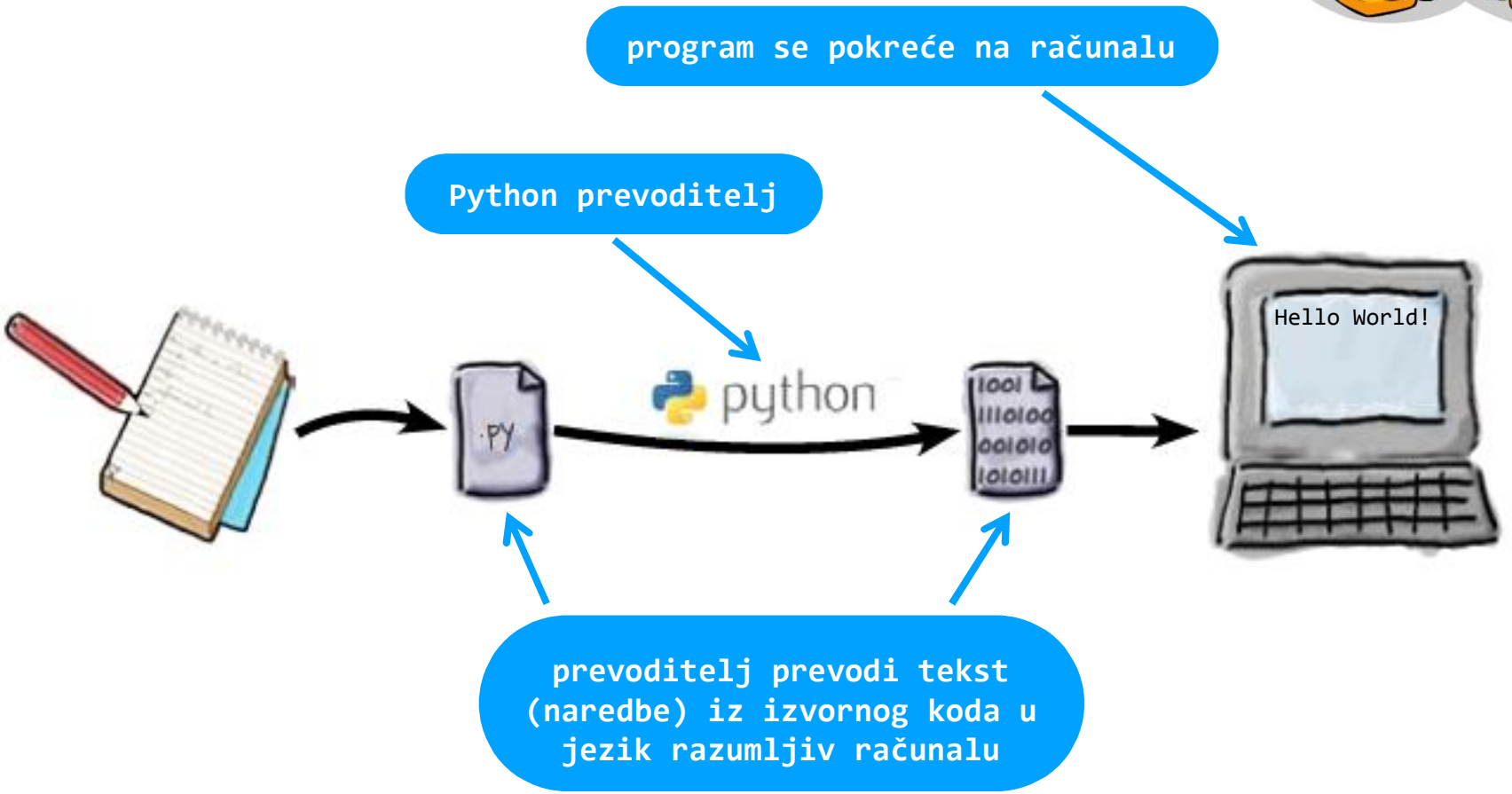
Prvi Python program





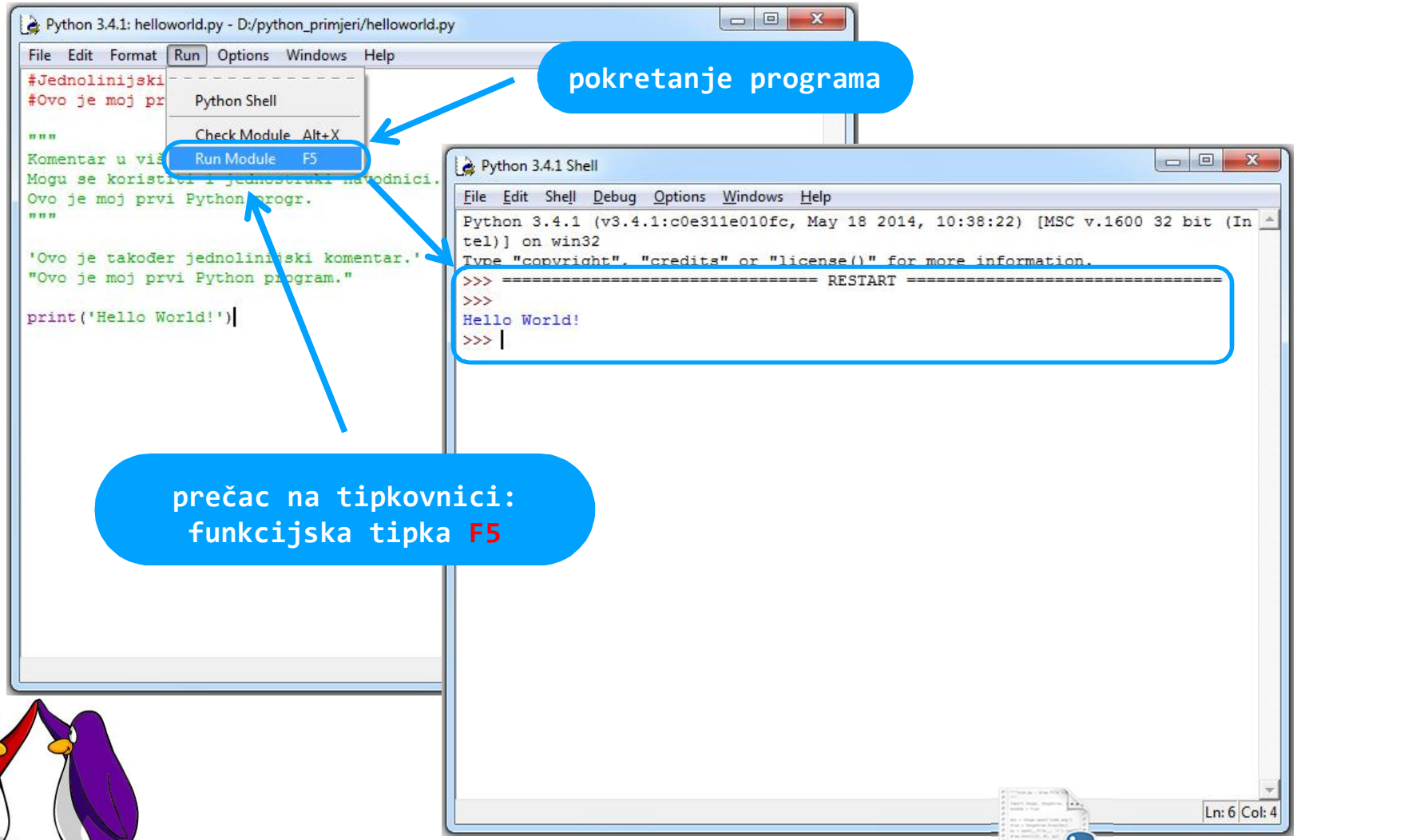
Prevođenje programa





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

Pokretanje programa u Python IDLE-u

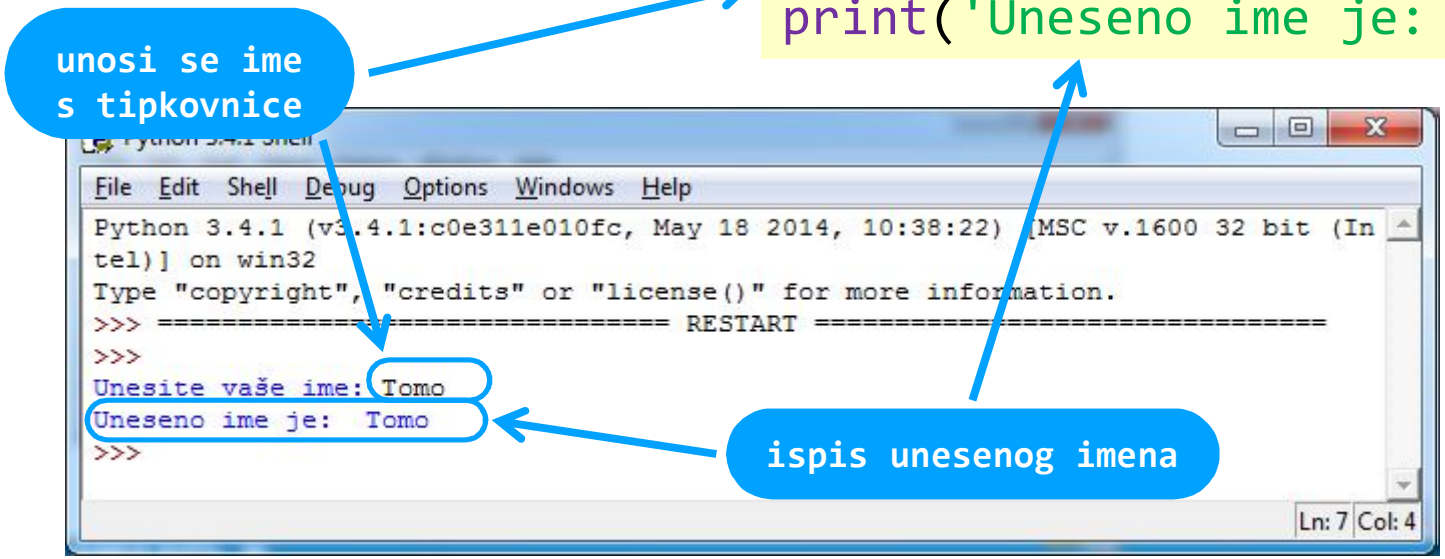


helloworld.py

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)
```



Zadatak: Vratar

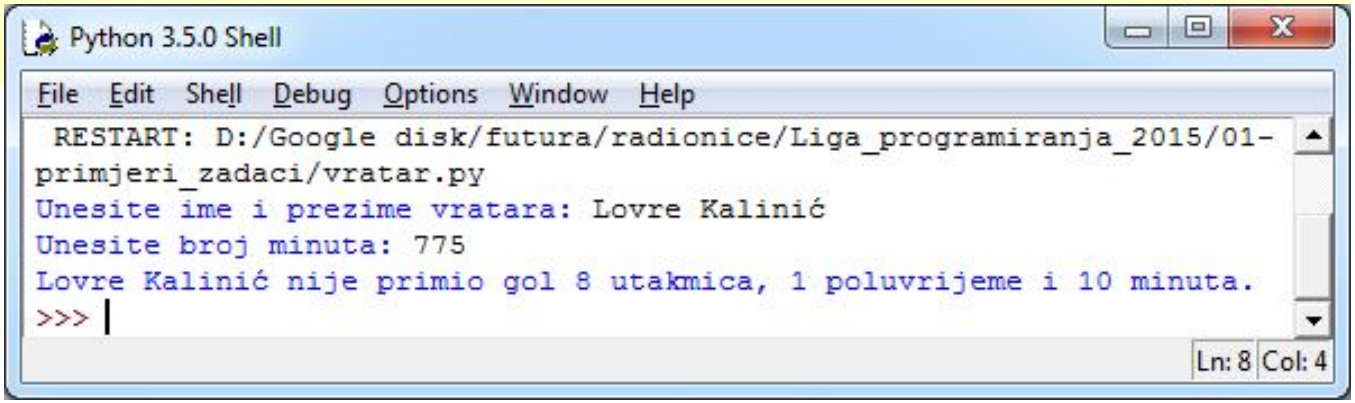
```
775 minuta = 8 utakmica,
1 poluvrijeme i 10 minuta
```

- 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.

Zadatak: Vratar - rješenje

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

vratar.py



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
```

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

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

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)

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

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	`and`
logička ILI operacija	`or`
NE operacija (negacija)	`not`

- Redoslijed izvođenja logičkih operacija:

1.	`not`
2.	`and`
3.	`or`

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**

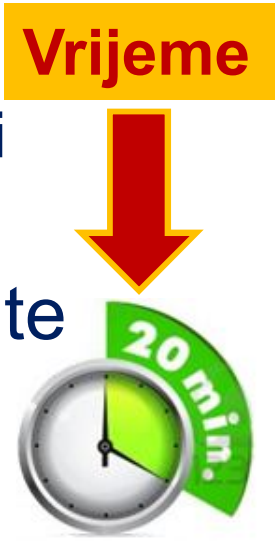
Ako imamo kombinirane aritmetičke, relacijske i logičke operatore, onda je ovo redoslijed izvođenja operacija.

```
>>> 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

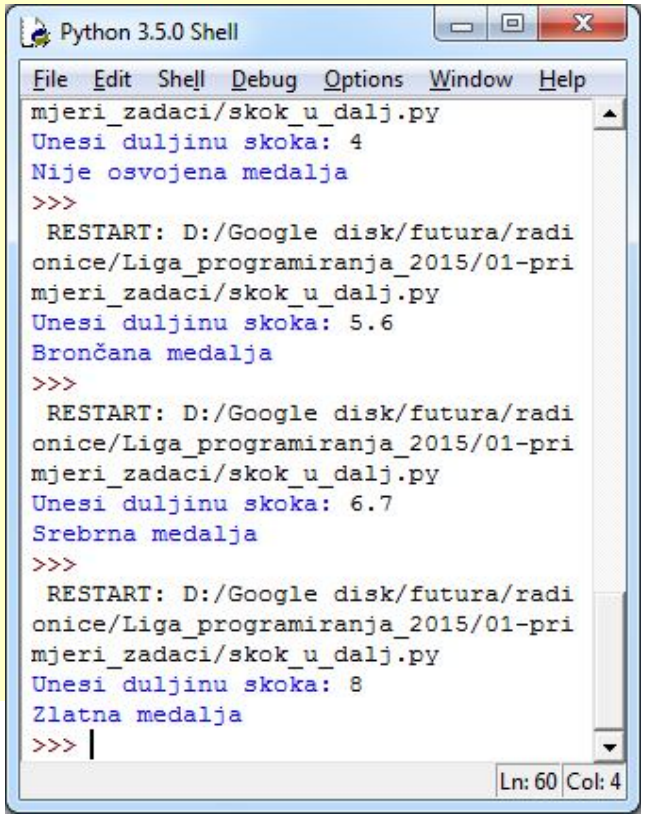
- Na natjecanju u skokovima u dalj organizatori su odlučili podijeliti više zlatnih, srebrnih i brončanih medalja prema sljedećim kriterijima:
  - Brončana medalja za sve koji preskoče između 5,5 i 6,5 metara, uključujući skokove od 5,5 i 6,5 metara
  - Srebrna medalja za sve koji preskoče preko 6,5 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.



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')
```



skok_u_dalj.py

Ponavljanje bloka naredbi

- Ponavljanje bloka naredbi određeni broj puta:

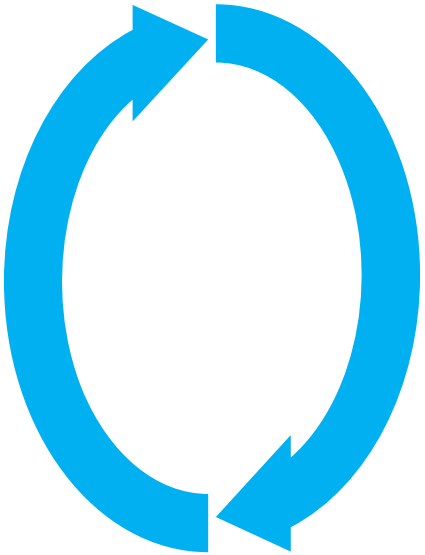
```
...
za i := 1 do n činiti
      naredba_1
      ...
      naredba_z
...
```

- `for` petlja

- Uvjetno ponavljanje bloka naredbi:

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

- `while` petlja



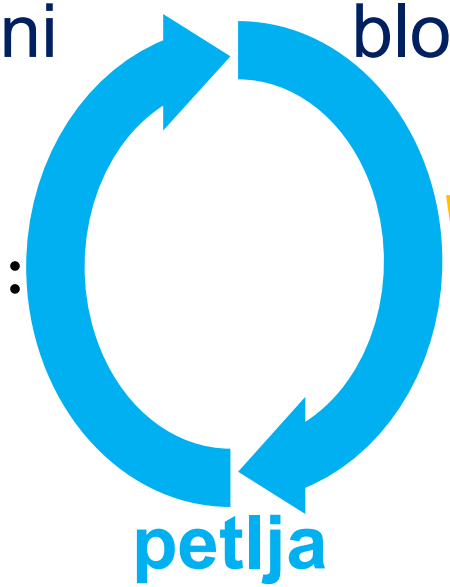
programska petlja

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**.



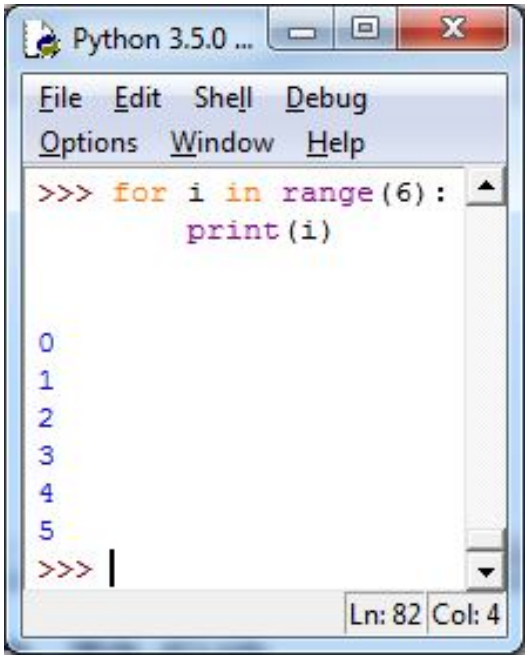
- Uvjetno ponavljanje bloka naredbi:

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

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

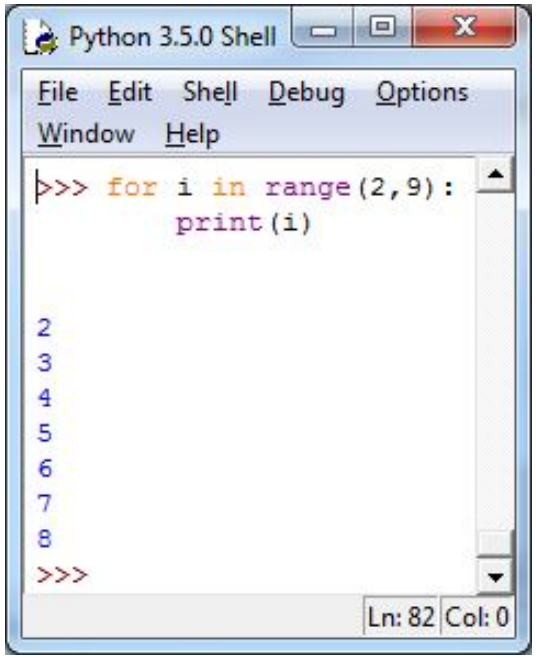
Vrste for petlje

- `range(n)`



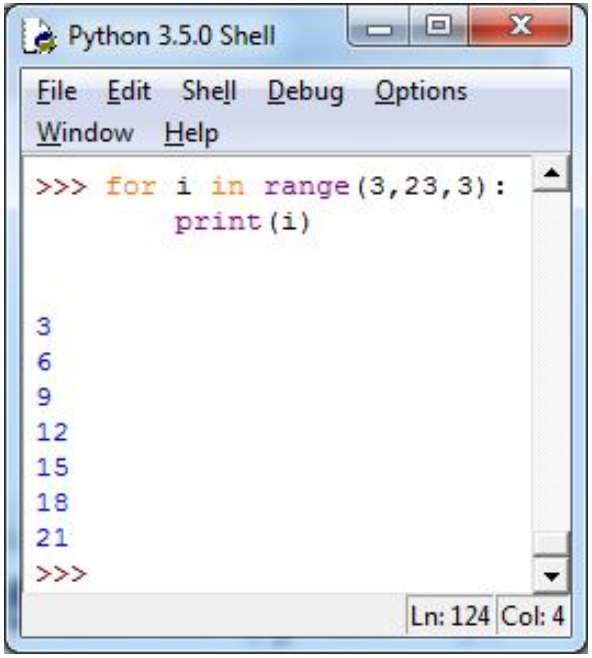
- Blok naredbi se izvodi **6 puta**, za vrijednosti **i** od **0** do **5**.

- `range(n,m)`



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

- `range(n,m,k)`

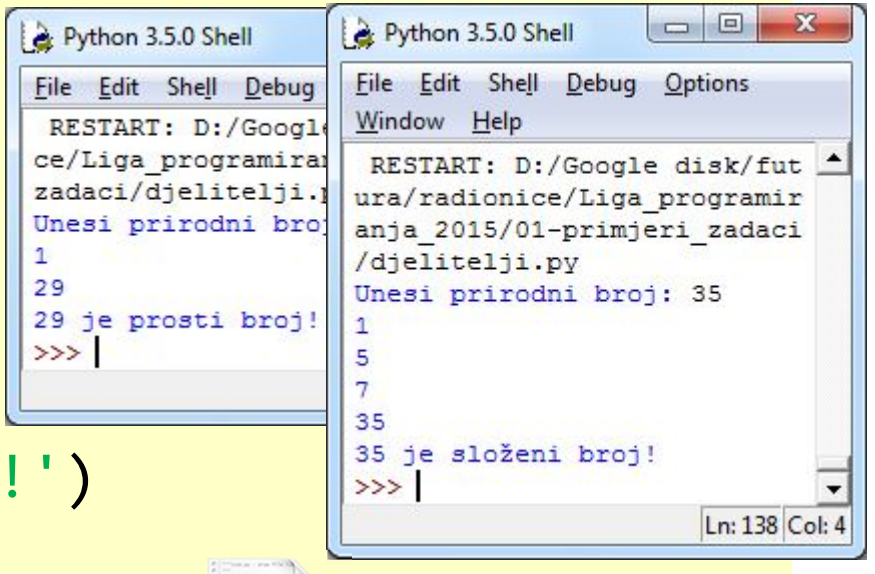


- Varijabla **i** mijenja vrijednost od **3** do **22** (tj. 23-1), s korakom **3**.

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".

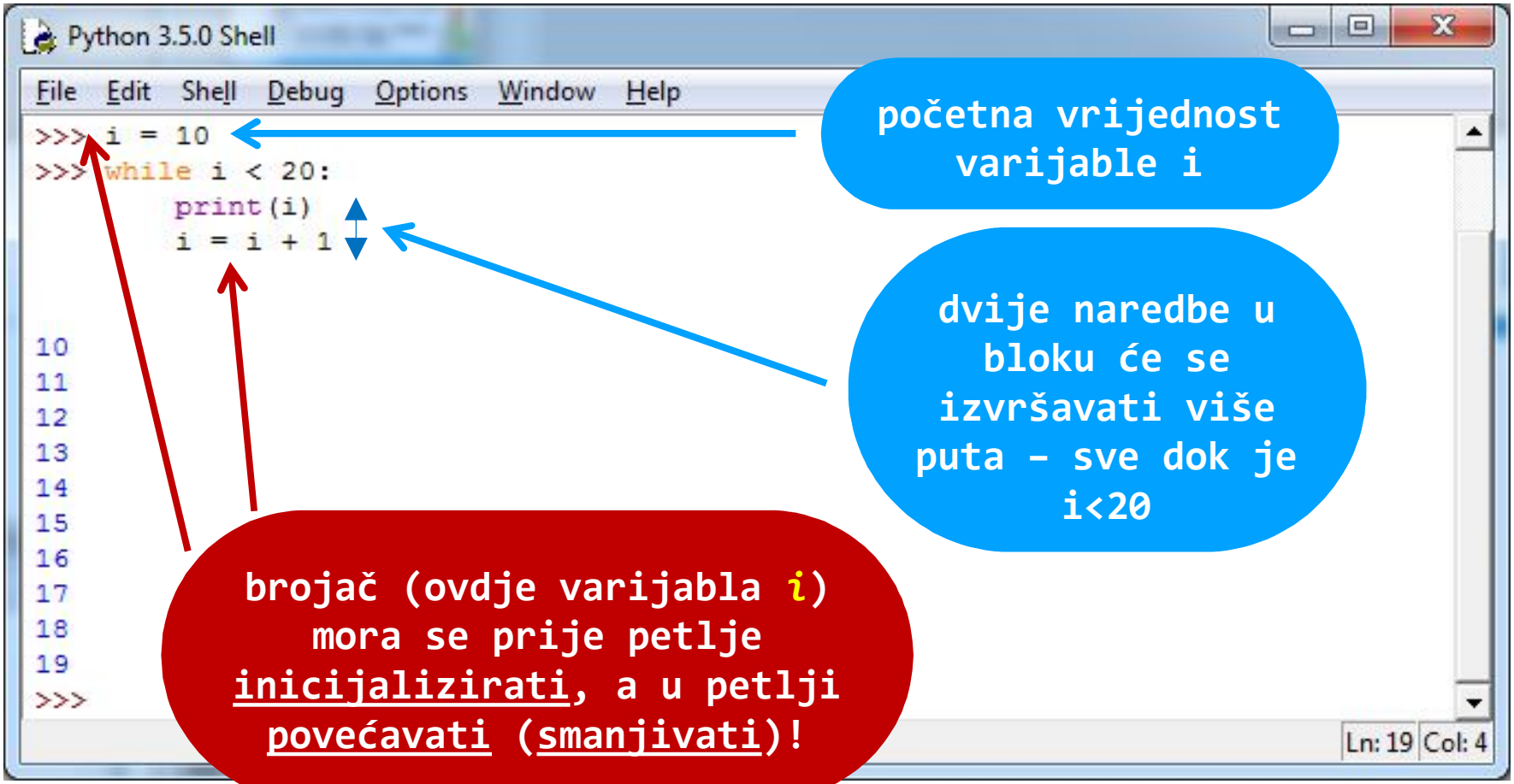
```
N = int(input('Unesi prirodni broj: '))
brDjel = 0
for i in range(1, N+1):
    if N%i == 0:
        print(i)
        brDjel = brDjel + 1
if brDjel == 2:
    print(N,'je prosti broj!')
else:
    print(N,'je složeni broj!')
```



djelitelji_for.py

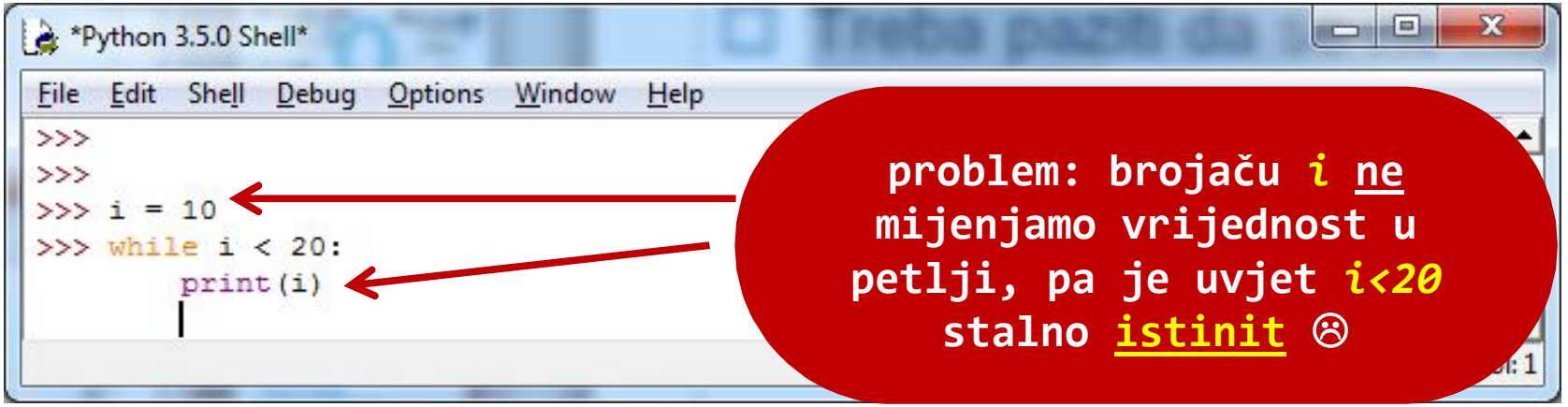
Uvjetno ponavljanje bloka naredbi

- **`while`** petlja



Uvjetno ponavljanje bloka naredbi

- Treba paziti da se ne napiše "beskonačna" petlja ☹ :

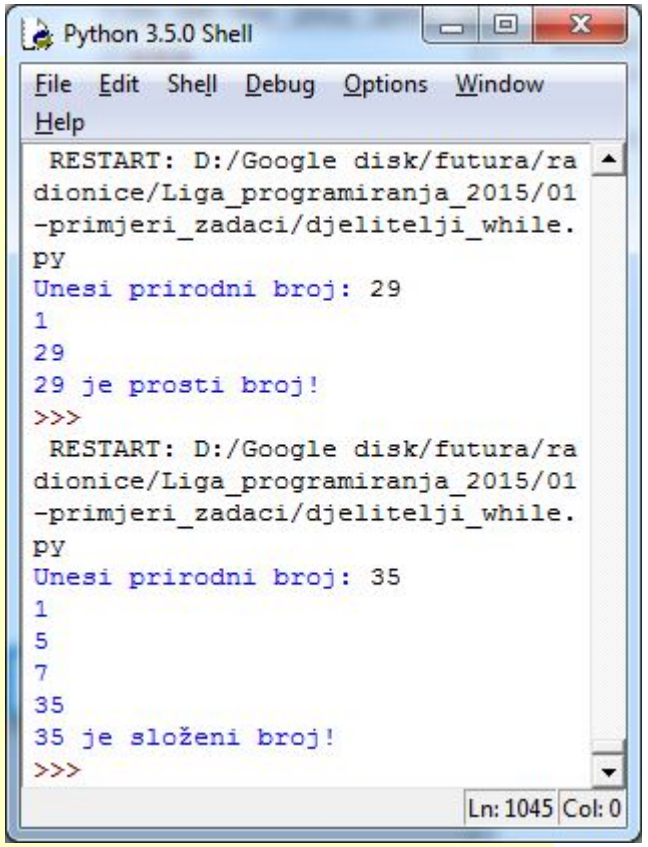


- Što će se dogoditi?
- Napisali smo "beskonačnu" petlju, pa je moramo prekinuti istovremenim pritiskom tipki **Ctrl** i **C**

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!')
```



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

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

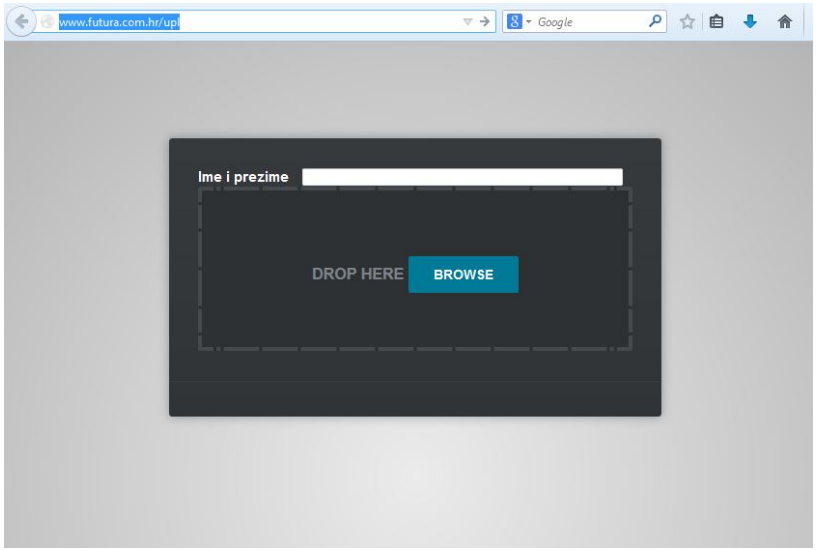
```
N = int(input('Unesi broj zadataka: '))
ukupno = 0
najmanji = 50
najveci = 0
for i in range(N):
    bod = int(input('Unesi broj bodova: '))
    ukupno = ukupno + bod
    if bod < najmanji:
        najmanji = bod
    if bod > najveci:
        najveci = bod
ukupno = ukupno - najmanji - najveci
prosjek = 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?

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!