INFORMATIČKI KLUB

FUTURA

python

PYTHON - PREDAVANJE I RADIONICA ZA NASTAVNIKE OSNOVNIH ŠKOLA

Tomo Sjekavica, Informatički klub FUTURA

Dubrovnik, 26. rujna 2014.

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/.

Sadržaj

- Programski jezik Python
- Instalacija Pythona
- Python IDLE
- Osnovni tipovi podataka
- Aritmetički operatori i izrazi
- Varijable
- Funkcija print
- Grananje i programske petlje
- Moduli

Programski jezik Python

- www.python.org
- Open source program
- Besplatni program
- Python Software Foundation (PSF)
- Podržan na Windows, Linux i Mac OS operacijskim sustavima
- Jednostavna sintaksa
- Podržava proceduralno programiranje i objektno orijentirano programiranje

Povijest Pythona

- Autor: Guido van Rossum - kraj 1989. godine
- Python 1.0 – siječanj 1994. godine
- Python 2.0 – listopad 2000. godine
- Python 3.0 – prosinac 2008. godine
- Posljednje verzije Pythona:
  - Python 2.7.8
  - Python 3.4.1
- Dokumentacija:
  - Python 2.7.8 – docs.python.org/2.7/
  - Python 3.4.1 – docs.python.org/3/

Primjena Pythona

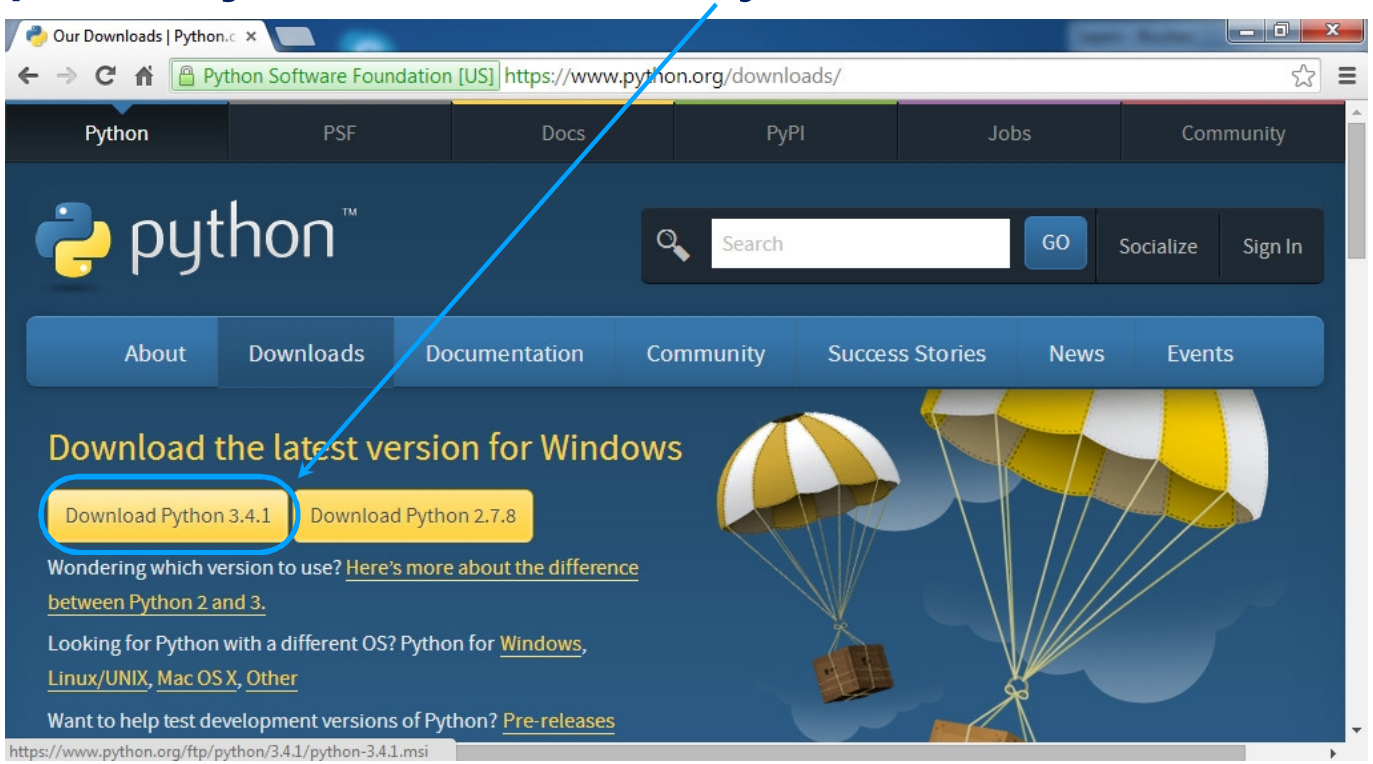
- Web programiranje:
  - Django, Pyramid, Botle, Tornado, Flask, web2py
- Razvoj samostojeće programske potpore:
  - wxPython, tkInter, PyGtk, PyQt
- Znanost i numeričke simulacije:
  - SciPy, Pandas, Ipython
- Razvoj softvera:
  - Buildbot, Trac, Roundup, Scons, Apache Gump
- Administracija sustava:
  - Ansible, Salt, OpenStack

Primjena Pythona

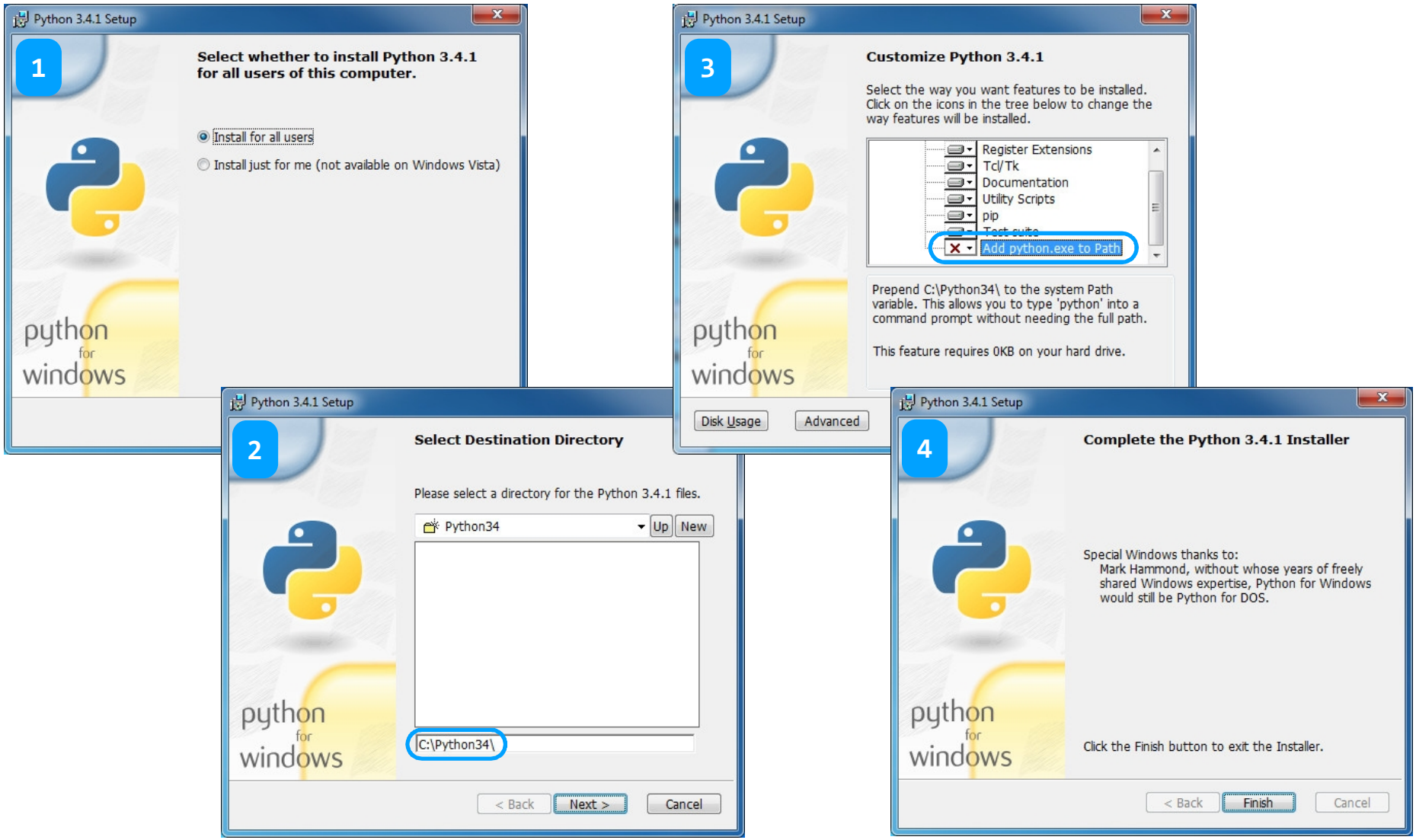
- Disqus – www.disqus.com
- Lanyrd – www.lanyrd.com
- Pinterest – ww.pinterest.com
- Instagram – www.instagram.com
- Google Inc. – www.google.com
- YouTube – www.youtube.com
- Mozilla Support – support.mozilla.org
- NASA – www.nasa.gov
- New York Times - www.nytimes.com
- The Guardian - www.theguardian.com/uk
- …

Instalacija Pythona

- www.python.org/downloads
- Koristiti ćemo zadnju verziju za Windows operacijski sustav – Python 3.4.1

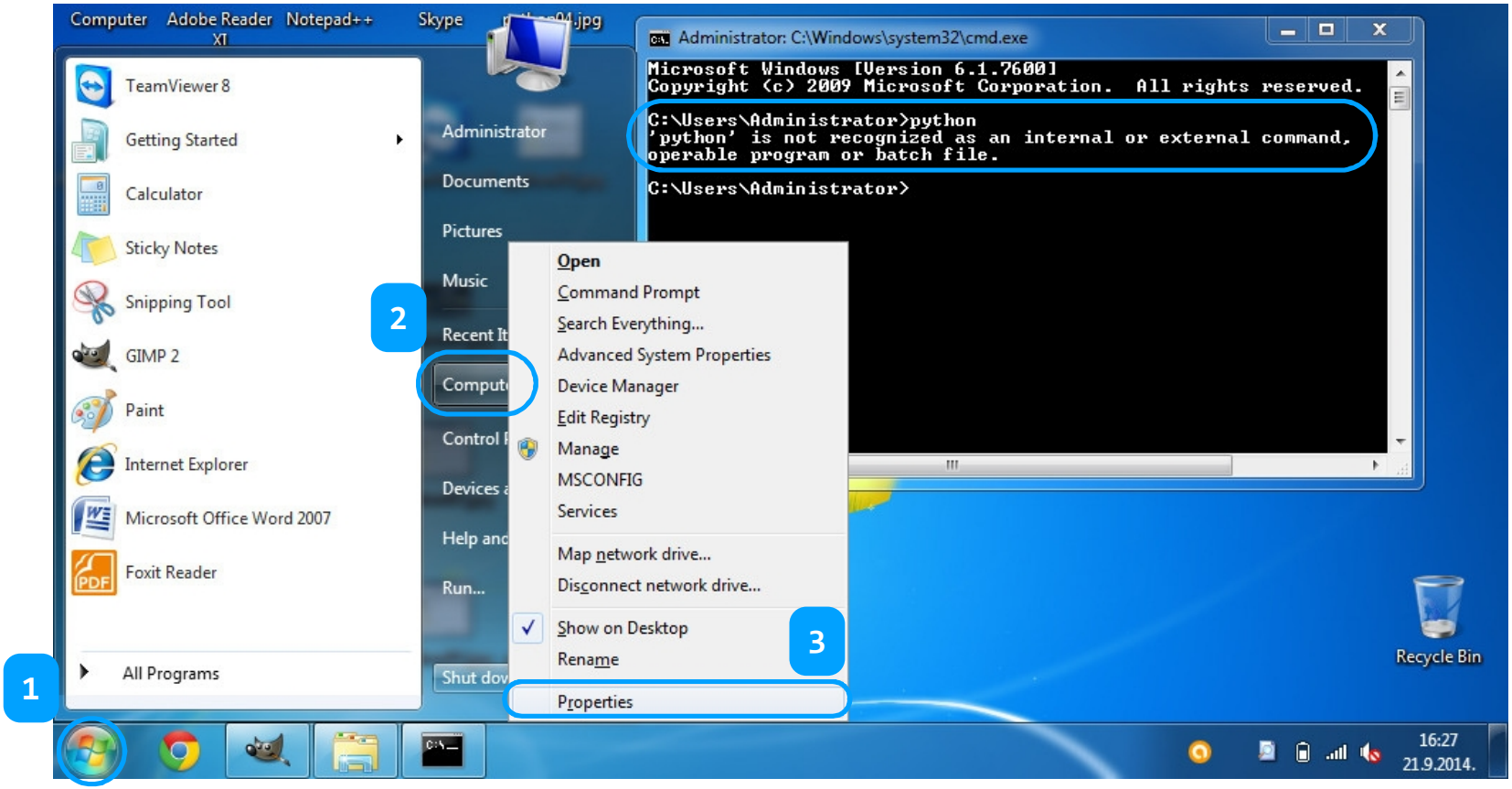


Instalacija Pythona

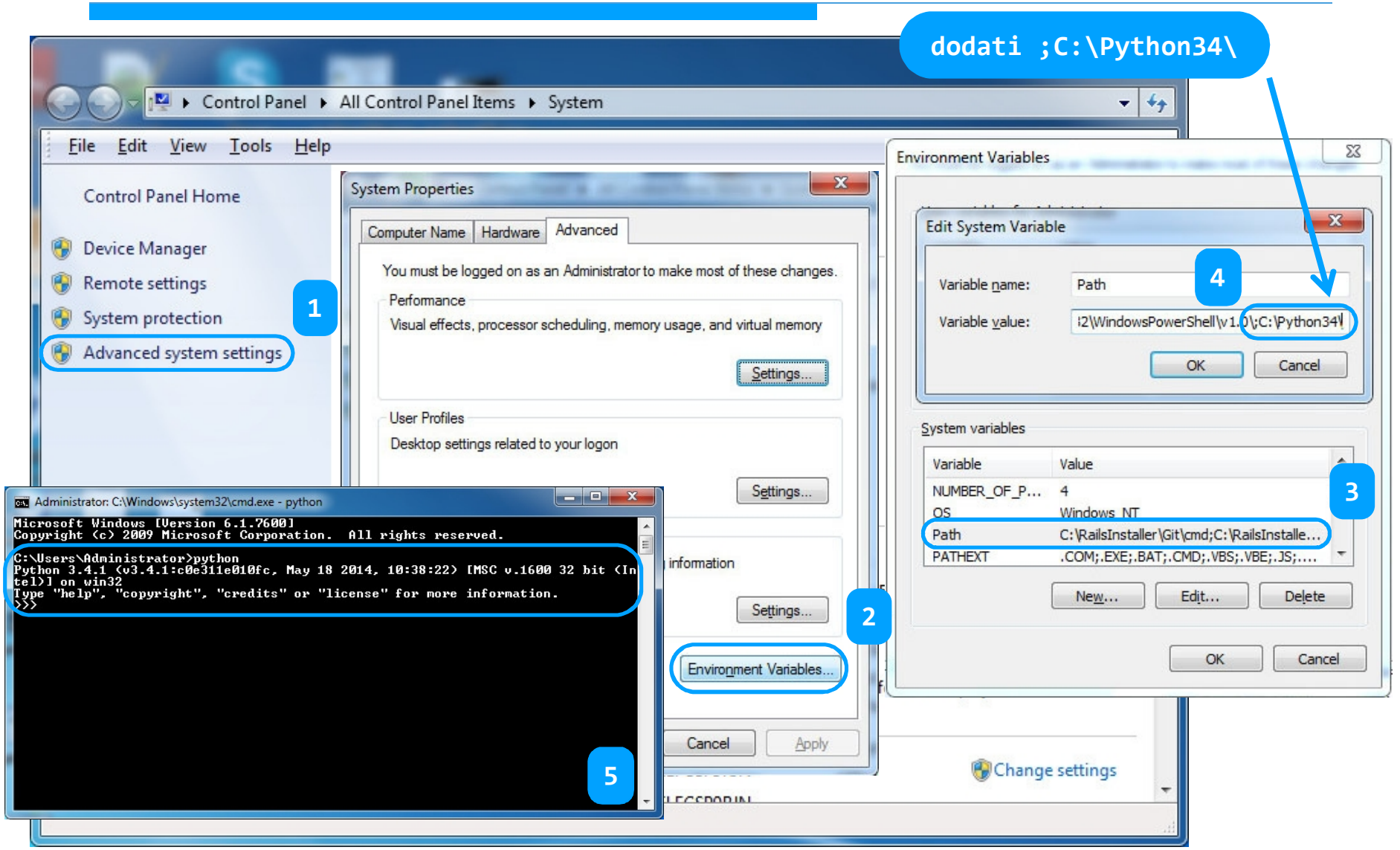


Korištenje Pythona u cmd-u

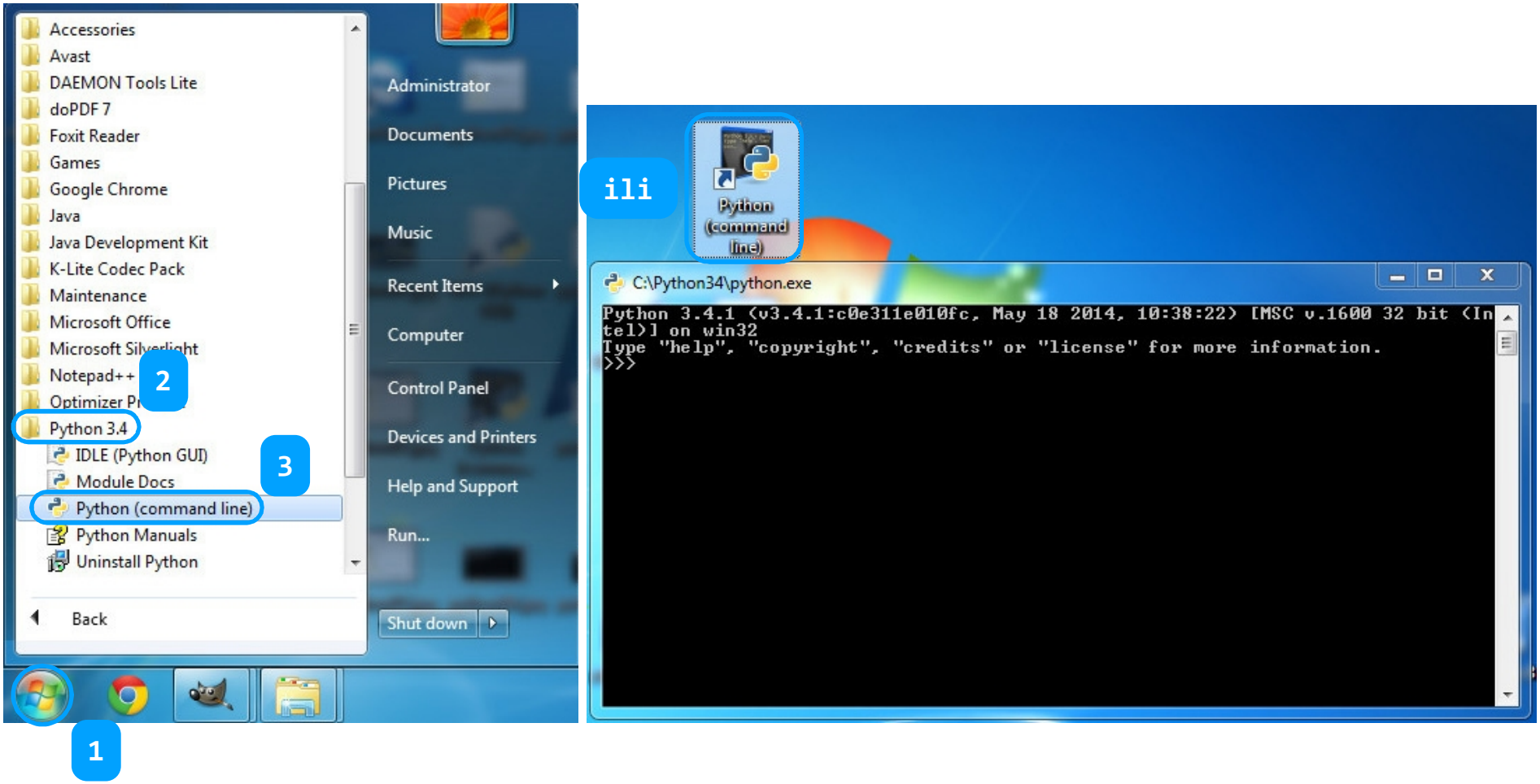
- Ako nije podešena putanja (path) ne može se koristiti u Windows Command Promptu



Dodavanje putanje za Python

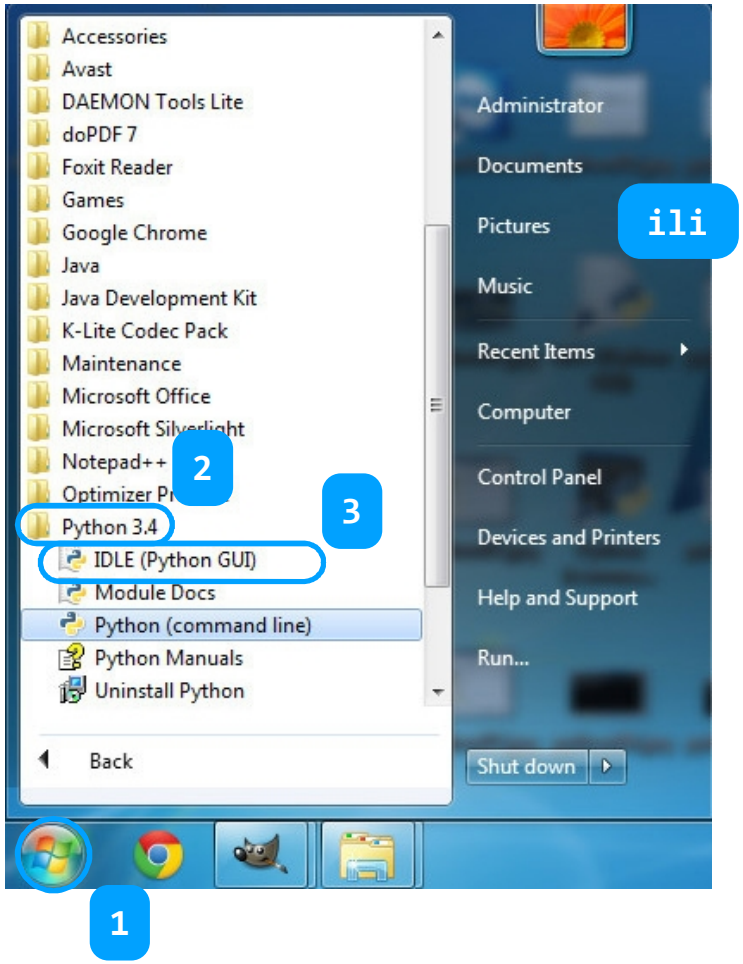


Pokretanje Python (command line)

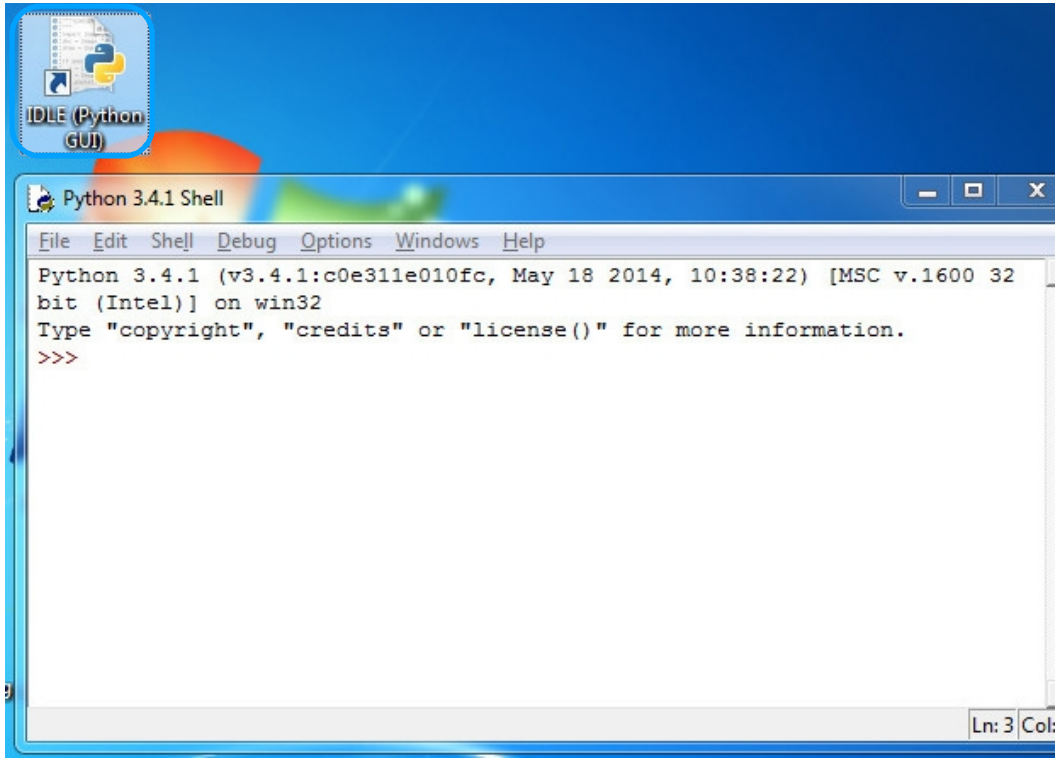


Pokretanje Python IDLE

- **IDLE** (**I**ntegrated **D**eve**L**opment **E**nviroment)



ili



Osnovni tipovi podataka u Pythonu

- **`int`** – cijeli broj

```
>>> type(21)
<class 'int'>
```

```
>>> type(-21)
<class 'int'>
```

- **`float`** – broj s pomičnom točkom

```
>>> type(2.2)
<class 'float'>
```

```
>>> type(-2.2)
<class 'float'>
```

- **`bool`** – logički tip podatka

```
>>> type(True)
<class 'bool'>
```

```
>>> type(False)
<class 'bool'>
```

- **`str`** – niz znakova (string)

```
>>> type('tekst')
<class 'str'>
```

```
>>> type("2.2")
<class 'str'>
```

Cijeli brojevi

- Nije ograničen broj znamenaka cijelog broja

```
>>> 12345678901234567890123456789012345678901234567890123456789012345678901234567890
```

- Binarni brojevi

```
>>> 0b00010101
21
```

```
>>> 0b00010101
240
```

- Heksadekadski brojevi

```
>>> 0x15A
346
```

```
>>> 0xC1F
3103
```

- Pretvaranje u binarni i heksadekadski broj

```
>>> bin(21)
'0b10101'
```

```
>>> hex(346)
'0x15a'
```

Brojevi s pomičnom točkom

- Dvostruka preciznost – 8 bajtova

```
>>> 2.2
2.2
>>> 0.00022
0.00022
>>> 1e15
1000000000000000.0
>>> 2.
2.0
>>> .000022
2.2e-05
>>> 1e16
1e+16
>>> 0.22
0.22
>>> 1e2
100.0
```

- Donja granica brojeva s pomičnom točkom

```
>>> 1.2345678901234567e-323
1e-323
>>> 1.2345678901234567e-324
0.0
```

- Gornja granica brojeva s pomičnom točkom

```
>>> 1.23456789012345678e308
1.2345678901234567e+308
>>> 1.23456789012345678e308
inf
```

Logički tip podatka

- Dvije vrijednosti – istina **`True`** ili laž **`False`**

```
>>> True                          >>> False
True                              False
>>> true
Traceback (most recent call last):
  File "<pyshell#43>", line 1, in <module>
    true
NameError: name 'true' is not defined
```

- Funkcija **`bool`** – može pretvoriti **`int`** u **`bool`**

```
>>> bool(1)                       >>> bool(0)
True                              False
```

- Funkcija **`int`** – može pretvoriti **`bool`** u **`int`**

```
>>> int(True)                     >>> int(False)
1                                 0
```

Nizovi znakova

- Jednostruki ili dvostruki navodnici

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

- Ispis dvostrukih navodnika u nizu znakova

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

- Ispis jednostrukih navodnika u nizu znakova

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

- Funkcija **`print`**

```
>>> print('Python')              >>> print('Predavanje "Python"')
Python                          Predavanje "Python"
```

Nizovi znakova

- 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\nu Pythonu.')
Prelazak
u novi red
u Pythonu.
```

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

```
>>> 4 + 3
7
>>> 4 / 3
1.3333333333333333
>>> 4 // 3
1
>>> 4 - 3
1
>>> 4 % 3
1
>>> 4.0 // 3
1.0
>>> 4 * 3
12
>>> 4 ** 3
64
>>> 4 // 3.
1.0
```

- Složeni aritmetički izrazi

```
>>> 2 + 2 * 2
6
>>> (2 + 2) * 2 + 4 / 3
9.333333333333334
```

- Zadnja izračunata vrijednost

```
>>> 2 + 2
4
>>> _ * 2
8
```

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 **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 + 10
>>> print('x =', x)
x = 20
>>> python = 'Predavanje Python'
>>> print(python)
Predavanje Python
```

```
>>> y = 2.2
>>> print(y)
2.2
>>> y = y  * 2
>>> print('y =', y)
y = 4.4
```

- Null vrijednost

```
>>> varijabla = None
>>> print(varijabla)
None
```

Funkcija print

- Formatirani ispis s **print** funkcijom

```
print('string1{broj1}string2{broj2},…,
stringN{brojN}' .format(var1, var2,…, varN))
```

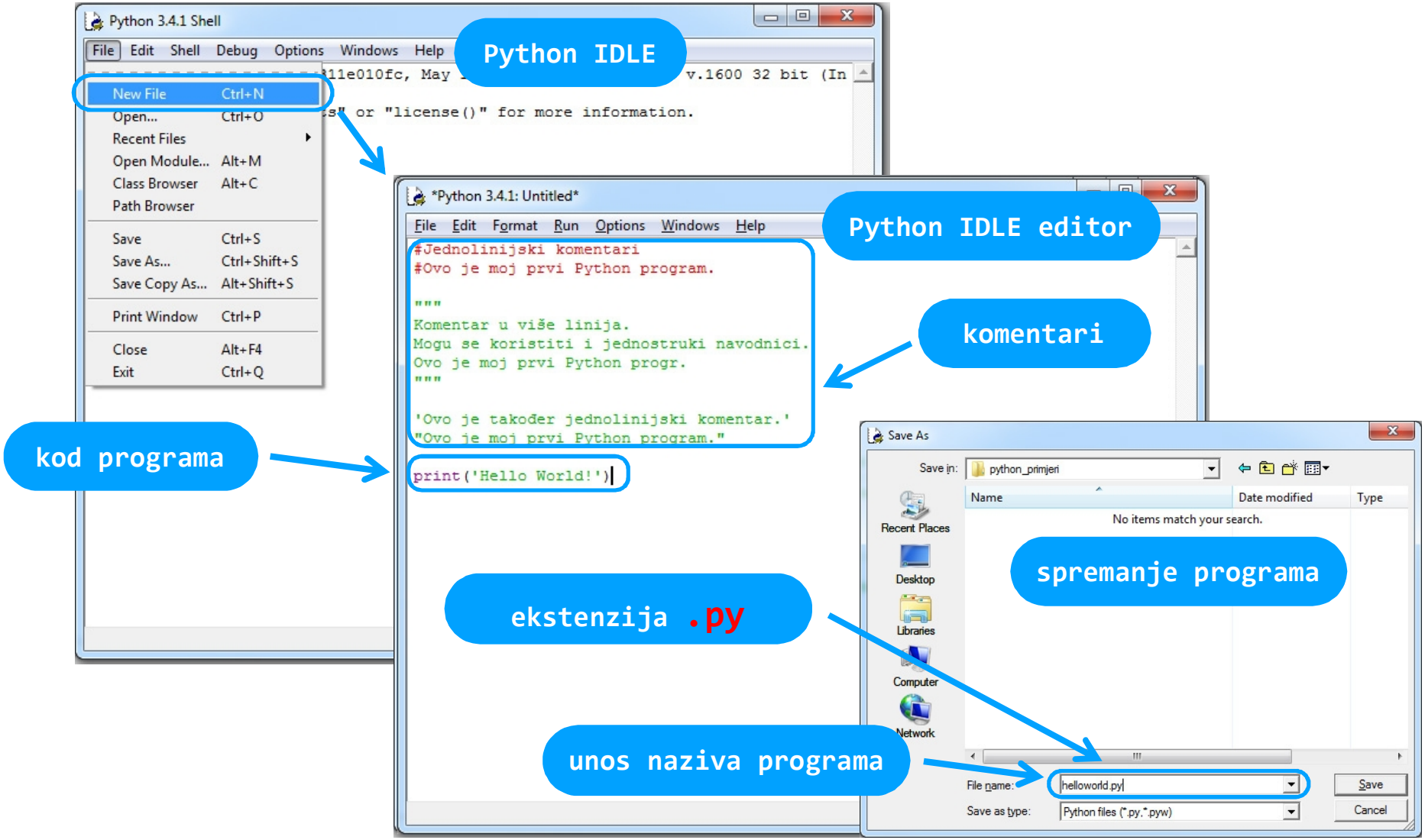
- Primjeri formatiranog ispisa

```
>>> print('Brojevi {0} i {1} su {2}'
.format(1, 2, 'cijeli brojevi'))
Brojevi 1 i 2 su cijeli brojevi
```

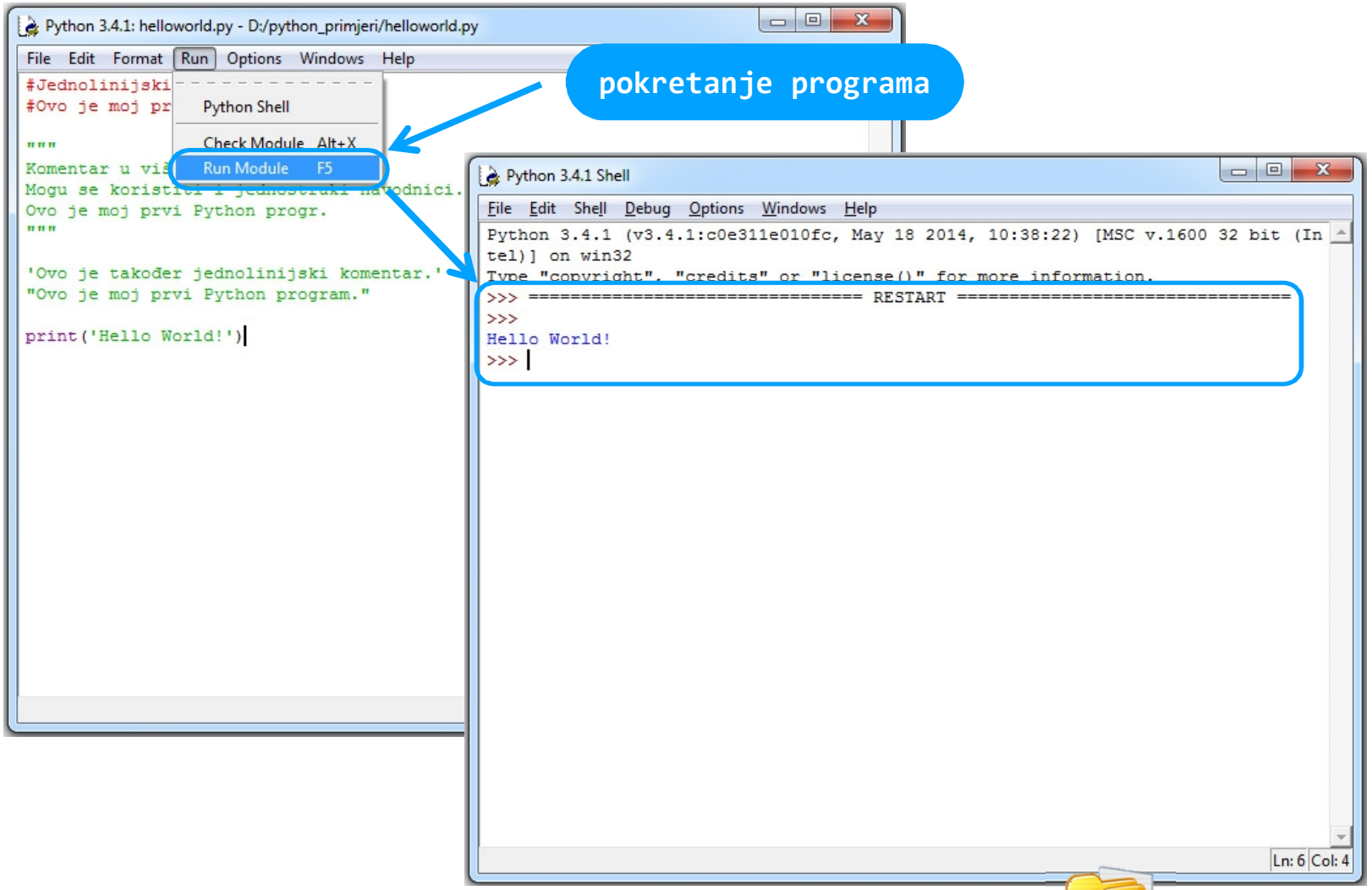
- Oznaka tipa podatka kod ispisa

```
>>> print('{3:s} brojeva {0:d} i {1:d} je {2:f}'
.format(1, 2, 1/2, 'Rezultat dijeljenja'))
Rezultat dijeljenja brojeva 1 i 2 je 0.500000
```

Prvi Python program



Pokretanje programa u Python IDLE-u



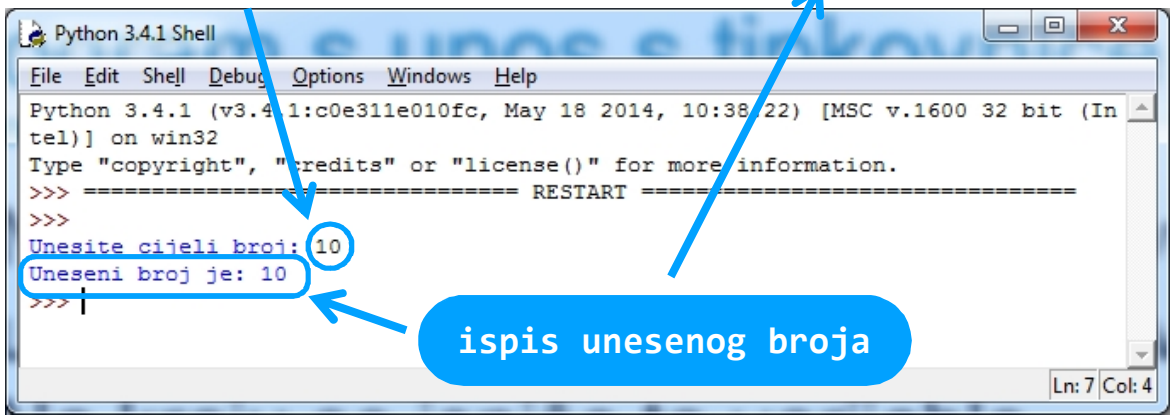
helloworld.py

Unos s tipkovnice

- Napisati program kojim se s tipkovnice unosi jedan cijeli broj i sprema u varijablu. Nakon toga se ispisuje vrijednost varijable.
- Info: za unos s tipkovnice se koristi funkcija **input**

```
#tipkovnica.py
x = input('Unesite cijeli broj: ')
print('Uneseni broj je: ', x, sep='')
```

unosi se broj s tipkovnice



ispis unesenog broja

tipkovnica.py

Relacijski i logički operatori

Relacijski operatori

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

Logički operatori

logička operacija I	`and`
logička operacija ILI	`or`
logička operacija NE	`not`

Relacijski i logički operatori

- Primjeri s relacijskim operatorima:

```
>>> 2 > 5
False
>>> 2 != 5
True
```

```
>>> 2 < 5
True
>>> 2 >= 5
False
```

```
>>> 2 == 5
False
>>> 2 <= 5
True
```

- Primjeri s logičkim operatorima:

```
>>> 0 and 0
0
>>> 0 and 1
0
>>> 1 and 0
0
>>> 1 and 1
1
```

```
>>> False or False
False
>>> False or True
True
>>> True or False
True
>>> True or True
True
```

```
>>> not 0
True
>>> not 1
False
>>> not False
True
>>> not True
False
```

```
0 - False
1 - True
```

Uvjetno grananje

- Ključne riječi: **`if`**, **`else`** i **`elif`**

```
…
if uvjet:
    blok_naredbi
…
```

```
…
if uvjet:
    blok_naredbi1
else:
    blok_naredbi2
…
```

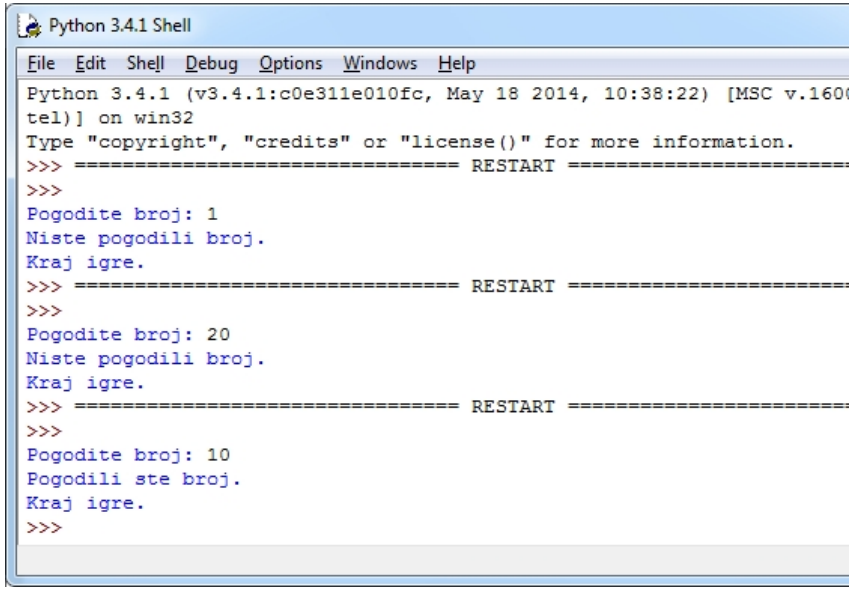
```
…
if uvjet1:
    blok_naredbi1
elif uvjet2:
    blok_naredbi2
else:
    blok_naredbi3
…
```

usporedba vrijednosti varijabli i/ili fiksnih vrijednosti pomoću relacijskih i logičkih operatora

- Ako je uvjet ispunjen, logički izraz je istina pa se izvršava prvi blok naredbi
- Ako uvjet nije ispunjen, logički izraz je laž pa se izvršava drugi blok naredbi

Pogodi broj v1

- Napisati program u kojem se s tipkovnice unosi jedan cijeli broj koji se uspoređuje s nekim fiksnim brojem, npr. brojem 10. Ako su brojevi jednaki treba ispisati tekst „Pogodili ste broj.”, a ako nisu tekst „Niste pogodili broj.”. Radi pojednostavljenja programa pretpostavit ćemo da će se uvijek unositi numerička vrijednost s tipkovnice.



Pogodi broj v1 - rješenje

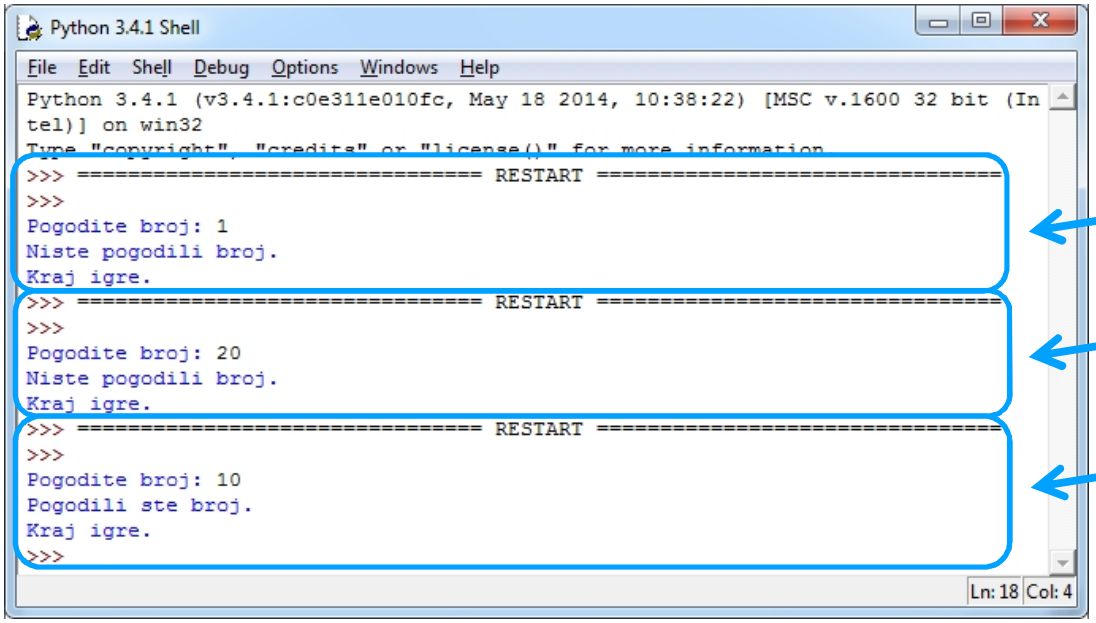
```
b = input('Pogodite broj: ')
broj = int(b)
if broj == 10:
    print('Pogodili ste broj.')
else:
    print('Niste pogodili broj.')
print('Kraj igre.')
```

unos broja s tipkovnice

pretvorba niza znakova u broj

provjera da li je uneseni broj jednak broju 10

VAŽNO: kod Pythona je bitno uvlačenje koda, dio koji se izvršava u if ili else grani mora biti uvučen za 4 razmaka ili 1 tabulator



1. pokretanje programa

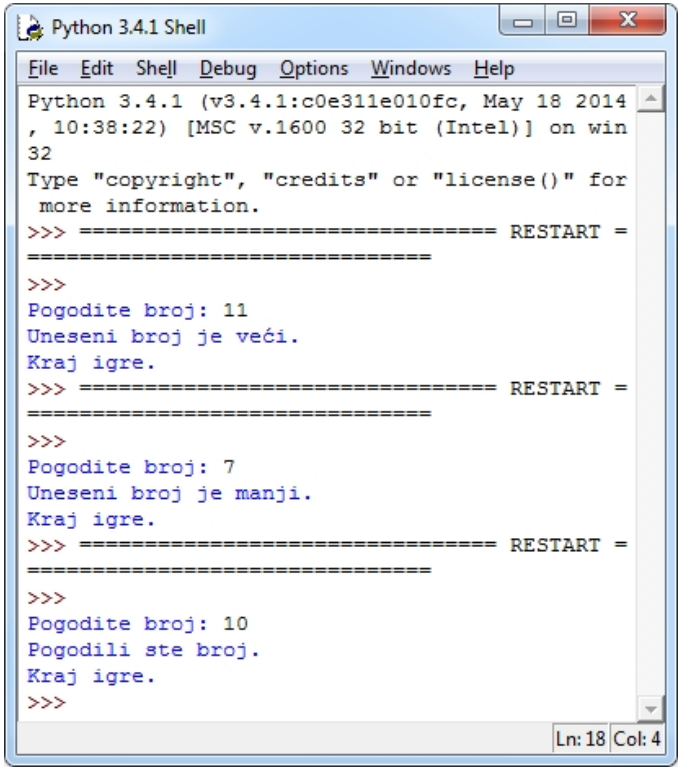
2. pokretanje programa

3. pokretanje programa

broj_v1.py

Pogodi broj v2

- Nadograditi prethodni program tako da se umjesto teksta „Niste pogodili broj.” ispisuje pomoćni tekst „Uneseni broj je manji.” ako je unesen manji broj od fiksnog broja ili „Uneseni broj je veći.” ako je unesen veći broj od fiksnog broja.



Pogodi broj v2 - rješenje

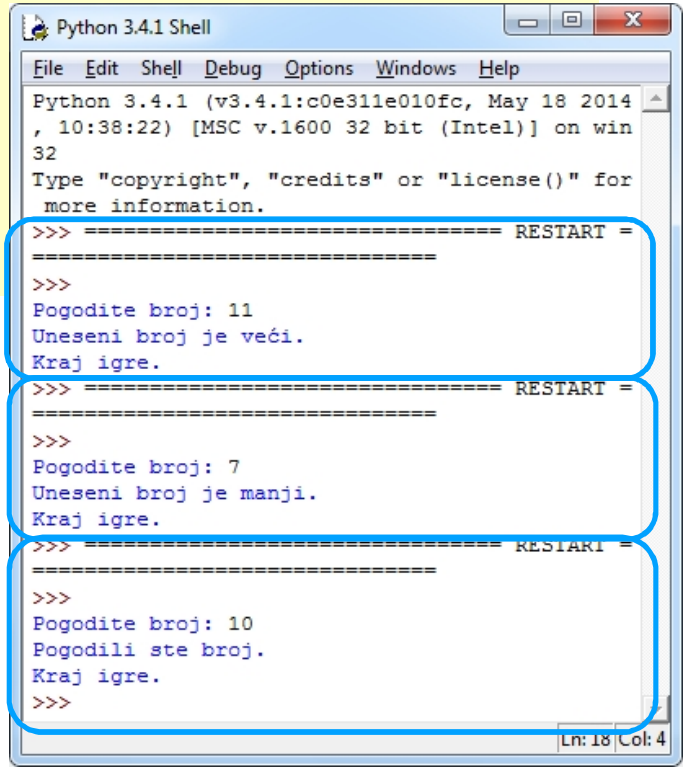
```
broj = int(input('Pogodite broj: '))
if broj == 10:
    print('Pogodili ste broj.')
elif broj < 10:
    print('Uneseni broj je manji.')
else:
    print('Uneseni broj je veći.')
print('Kraj igre.')
```

VAŽNO: ne zaboravite na uvlačenje koda

1. pokretanje programa

2. pokretanje programa

3. pokretanje programa



broj_v2.py

Programske petlje – `for` petlja

- `for` petlja se koristi kad znamo konačan broj ponavljanja dijela koda
- Opći oblik `for` petlje

```
…
for i in range(n):
    blok_naredbi
…
```

i u prvom prolasku kroz petlju ima vrijednost 0, u drugom vrijednost 1,…, a u zadnjem prolasku kroz petlju n-1

- Funkcija **`range`** može poprimiti još 2 oblika:

```
range(a, z)                    range(a, z, k)
```

- **`i`** će poprimati vrijednosti od **`a`** do **`z`** s korakom **`1`** ako nije naveden ili s korakom **`k`**

Primjeri s `for` petljom

```
>>> for i in range(5):
        print(i)
0
1
2
3
4
```

```
>>> for i in range(10):
        print(i, end = ' ')
0 1 2 3 4 5 6 7 8 9
```

```
>>> for i in range(3,12):
        print(i, end = ' ')

3 4 5 6 7 8 9 10 11
```

```
>>> for i in range(3,30,3):
        print(i, end = ' ')

3 6 9 12 15 18 21 24 27
```

```
>>> for i in range(5,1,-1):
        print(i, end = ' ')

5 4 3 2
```

```
>>> for i in range(1,5,-2):
        print(i, end = ' ')
```

Programske petlje – `while` petlja

- `while` petlja se koristi kad ne znamo konačan broj ponavljanja dijela koda
- Opći oblik `while` petlje

```
…
while uvjet:
    blok_naredbi
…
```

- Petlja se izvršava sve dok je uvjet zadovoljen
- Kad uvjet nije zadovoljen završava se izvođenje petlje i nastavlja se dalje program

Primjeri s `while` petljom

```
>>> i = 0
>>> while i < 5:
        print(i, end = ' ')

0 0 0 0 0 0 0 0 0 0 0 Traceback (most recent call last):
File "<pyshell#37>", line 2, in <module>
  print(i, end = ' ')
File "C:\Python34\lib\idlelib\PyShell.py", line 1342, in write
  return self.shell.write(s, self.tags)
KeyboardInterrupt
```

beskonačna petlja:
ova petlja nema kraj, jer je i uvijek 0 i uvijek će biti zadovoljen uvjet petlje

Ctrl + C za prekid programa putem tipkovnice

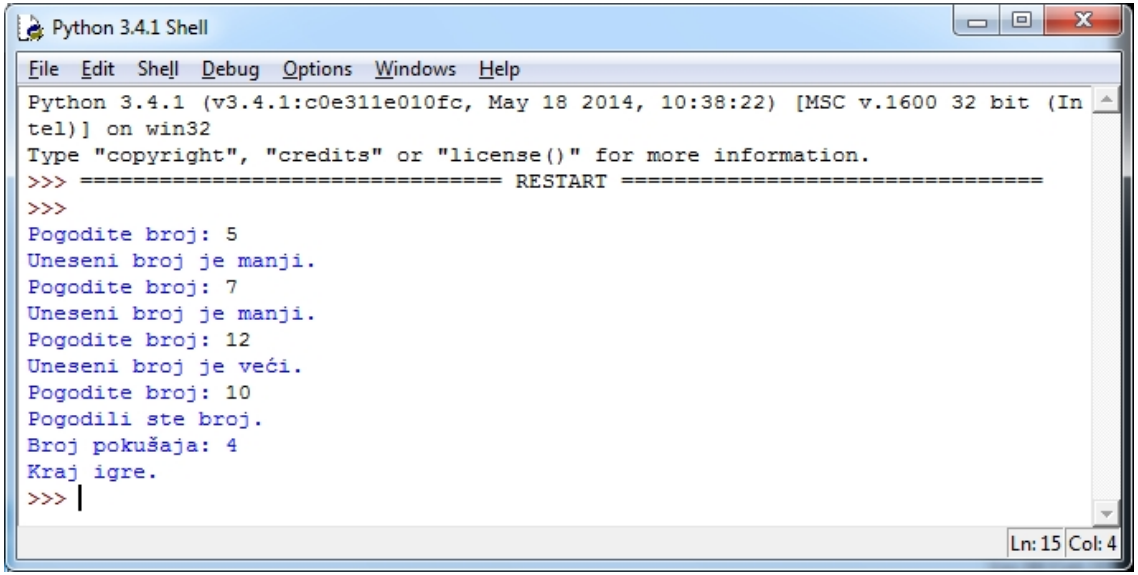
```
>>> i = 0
>>> while i < 5:
        print(i, end = ' ')
        i += 1

0 1 2 3 4
```

inicijalizacija varijable i na vrijednost 0

uvećavanje vrijednosti varijable i za 1, isto kao da piše i = i + 1

Pogodi broj v3

- Nadograditi prethodni program tako da se unos broja ponavlja sve do se ne unese broj koji je jednak fiksnom broju. Izbrojati koliko je pokušaja trebalo da se pronađe zadani fiksni broj, te na kraju ispisati broj pokušaja.



Pogodi broj v3 - rješenje

```
brojac = 0
broj = 0

while broj != 10:
    broj = int(input('Pogodite broj: '))
    if broj == 10:
        print('Pogodili ste broj.')
        brojac += 1
    elif broj < 10:
        print('Uneseni broj je manji.')
        brojac += 1
    else:
        print('Uneseni broj je veći.')
        brojac += 1
print('Broj pokušaja:', brojac)
print('Kraj igre.')
```

inicijalizacija broja i brojača na vrijednost 0

program se vrti u while petlji dok ne pogodimo broj

unos broja s tipkovnice se odvija u petlji

uvećanje brojača za broj pokušaja za 1

VAŽNO: ne zaboravite na uvlačenje koda

broj_v3.py

Moduli

- Najčešće korišteni moduli su **`math`** i **`random`**
- Modul se prije korištenja mora uvesti s naredbom **`import naziv_modula`**
- Funkcija **`sqrt`** 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)
3.0
>>>
```

Moduli

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

```
>>> from math import sqrt, exp
>>> sqrt(9)
3.0
>>> exp(2)
7.38905609893065
```

- 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:

`sqrt(x)`	korijen broja x
`pow(x, y)`	potencija broja x na potenciju y
`exp(x)`	e^x
`log(x, base)`	$\log_{base} x$
`ceil(x)`	zaokruživanje na najmanji cijeli broj veći ili jednak broju x
`floor(x)`	zaokruživanje na najveći cijeli broj veći ili jednak broju x
`sin(x)`	sin x
`cos(x)`	cos x
`tan(x)`	tg x

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

Modul random

- Funkcije za generiranje slučajnih brojeva

`randint(a, b)`	vraća slučajni cijeli broj n koji je a <= n <= b
`random()`	vraća slučajni realni broj n koji je 0.0 <= n < 1.0
`uniform(a, b)`	vraća slučajni realni broj n koji je a <= n <= b ako je a <= b ili je b <= n <= a ako je b < a

- https://docs.python.org/3/library/random.html

```
>>> from random import *
>>> randint(0,10)
5
>>> random()
0.8460300294602602
>>> uniform(0,10)
1.594305867774457
```

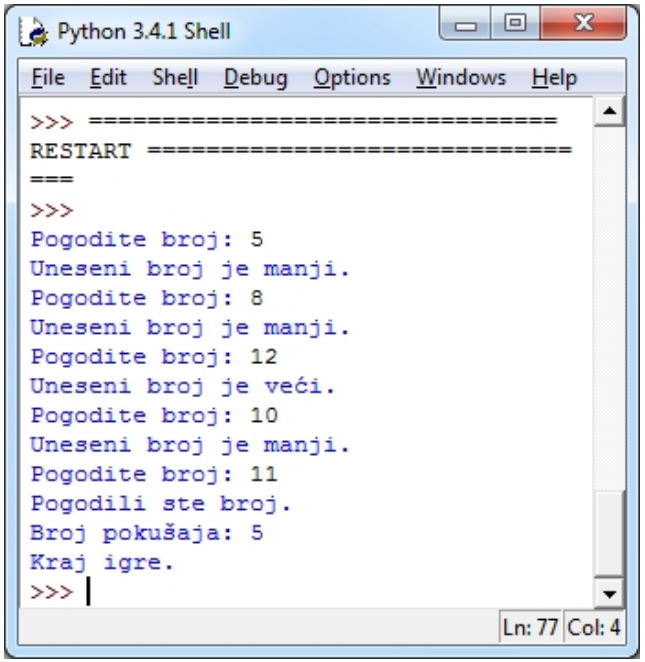
slučajni cijeli broj u intervalu [a, b]

slučajni realni broj u intervalu [0, 1)

slučajni realni broj u intervalu [0, 10]

Pogodi broj v4

- Nadograditi prethodni program tako da se generira slučajni broj iz intervala od 1 do 15 kojeg treba pogoditi umjesto do sada korištenog fiksnog broja. Uneseni broj s tipkovnice uspoređivati s tim slučajnim brojem. Optimizirati kod brojača u petlji iz prethodnog programa.



Pogodi broj v4 - rješenje

```
from random import randint
brojac = 0
broj = 0
zamisljeni = randint(1,15)

while broj != zamisljeni:
    broj = int(input('Pogodite broj: '))
    brojac += 1
    if broj == zamisljeni:
        print('Pogodili ste broj.')
    elif broj < zamisljeni:
        print('Uneseni broj je manji.')
    else:
        print('Uneseni broj je veći.')
print('Broj pokušaja:', brojac)
print('Kraj igre.')
```

uvoz funkcije randint iz modula random

generiranje slučajnog broja iz intervala [1, 15]

uvećanje brojača za broj pokušaja za 1

usporedba unesenog broja sa slučajno generiranim brojem

broj_v4.py