Univerzitet Crne Gore
Elektrotehnički fakultet

Prof. dr Vesna Popović-Bugarin

Uvod u Python

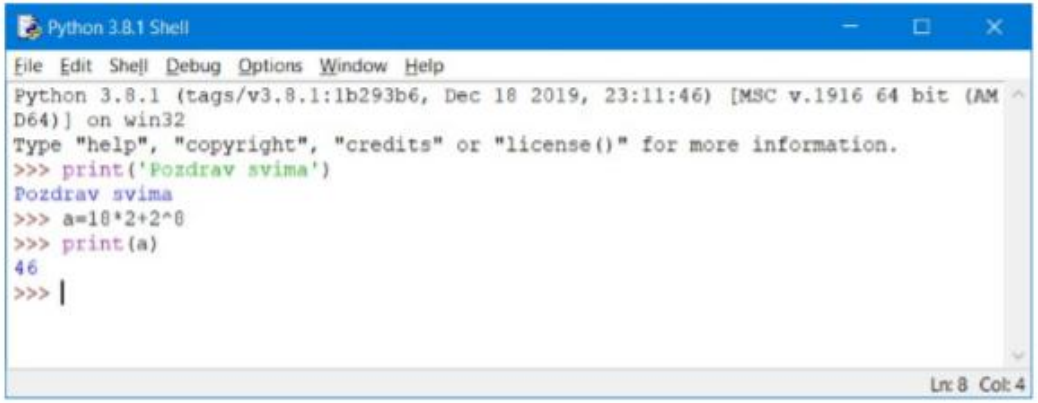
- Dvanaesti termin -

Crash kurs Python-a

- Python je, kao i mnogi drugi jezici (C++, Java, itd.) programski jezik visokog nivoa.
- Instalacija Python-a se može preuzeti sa sajta python.org.
- **IDLE** – integrisano razvojno okruženje (Integrated Development Environment – IDE) koje je dio Python instalacije.
- Pomenimo još neke Python-specifične editore i IDE:
  - **PyCharm** (integrisano razvojno okruženje)
  - **Spyder** – open source IDE optimizovan za data science (distribuira se često sa **Anaconda distribucijom**, koja se kategoriše kao Enterprise Data Science platform, prvobitno namijenjena naučnim proračunima, popularna u oblasti mašinskog učenja). Dobro se povezuje sa *SciPy*, *NumPy* i *Matplotlib*, koje su standardne ,,naučne" biblioteke.
  - **Thonny** – IDE namijenjen početnicima
- Mogu se koristiti i okruženja opštije namjene sa podrškom za Python: *Eclipse + PyDev*, *Sublime Text*, *Atom*, *GNU Emacs*, *Vi / Vim*, *Visual Studio*, *Visual Studio Code*

Crash kurs Python-a

- IDLE sadrži komandnu liniju, poseban interaktivni program (tzv. shell) u kojem se naredbe mogu unositi jedna za drugom, a nakon svake naredbe se može vidjeti rezultat.
- Python je interpreter, što znači da se programi izvršavaju kao sekvence uzastopnih naredbi.
- Moguće je sekvence naredbi čuvati u posebnim fajlovima – programima.



^ nije stepenovanje, već binarno ekskluzivno ili xor

Crash kurs Python-a

- Prepoznaje rad sa listama. Liste se zadaju u uglastim zagradama, navođenjem elemenata odvojenih zarezom.

```
a = [1, 2, 3, 4, 5]
```

- Elementima se pristupa korišćenjem imena liste i indeksa navedenog u uglastim zagradama. Indeks početnog elementa liste je 0. Dakle prvi element liste bi se očitao sa

```
a[0]
```

Za ispis podataka se koristi funkcija print. Može imati više argumenata koji se u tom slučaju odvajaju zarezom.

```
print("Ucimo Python",[2,3,4],5)
```

```
Ucimo Python [2, 3, 4] 5
```

Crash kurs Python-a

- Nema uglastih zagrada za odvajanje blokova koda, Python koristi uvlačenje (engl. identation)

```
for i in [1, 2, 3, 4, 5]:#lista
    print(i) # prva linija "for i" blok
    for j in [1, 2, 3, 4, 5]:
        print(j) # prva linija "for j" blok
        print(i + j) # zadnja linija "for j" blok
    print("\n")
    print(i) # zadnja linija "for i" blok
print("done looping")
```

- Razmaci unutar zagrada se ignorišu

```
In [17]: ignorisi = (1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 )

In [18]: ignorisi
Out[18]: 45
```

Crash kurs Python-a

```
range (start, stop[, step])
```

- Start i step su opcioni – ako start nije dato, generiše niz brojeva o 0 do stop-1

```
In [39]: x = range(10)
         print(x[1])
         x = range(2,10)
         print(x[1])
         x = range(2,10,2)
         print(x[1])
```

```
1
3
4
```

```
prvaDva = (x[:2])
for i in prvaDva:
    print(i)
```

```
2
4
```

```
In [47]: posljednjaDva = x[2:]
         odTrecegDoPetog = x[3:5]
         bezprvogiposljednjeg = x[1:-1]
         DoPretposljednjeg = x[:-1]
```

```
In [48]: for i in DoPretposljednjeg:
             print(i)
```

```
2
4
6
```

- **Moduli**
  - Osnovni elementi Python programskog jezika podrazumijevaju ključne elemente kao što su `for`, `if`, elementarni operatori, funkcije za unos i prikaz rezultata (`input`, `print`).
  - Ostali elementi dostupni su u tzv. **modulima** i moraju se uključiti, korišćenjem `import`-a, u sljedećem obliku:

```
from naziv_modula import sta_ukljucujemo
```

- **Slučajni cijeli brojevi – `randint`**
  - Funkcija `randint(a,b)` koja generiše slučajne cijele brojeve između a i b (uključujući a i b), dostupna je u modulu `random`, i može se uključiti na sljedeći način:

```
from random import randint
```

  - Primjer upotrebe:

```
from random import randint
x = randint(10,90)
print('Slucajan broj izmedju 10 i 90: ', x)
```

  - Svaki ponovni poziv `random` generiše novi slučajni broj između 10 i 90

- **Ugrađene funkcije**
  - Funkcije `abs` i `round` su ugrađene, odnosno, nije neophodno vršiti njihov *import*.
  - Fukcija `abs` računa apsolutnu vrijednost broja. Na primjer, `abs(-11.7)` kao rezultat daje `11.7`.
  - Funkcija `round` vrši zaokruživanje na najbliži cio broj, ili na najbližu decimalu. Zadaje se u obliku `round(a,n)`, gdje je prvi argument broj koji se zaokružuje, a drugi određuje na šta se zaokružuje prvi broj. Ako je $n > 0$, zaokruživanje se vrši n pozicija desno od decimalne tačke, za $n < 0$, zaokruživanje se vrši n pozicija lijevo od decimalne tačke, a za $n = 0$ imamo zaokruživanje na najbliži cio broj.
    - Na primjer, `round(5.8)` kao rezultat daje `6`.
    - `round(5.2)` kao rezultat daje `5`.
    - Sa druge strane, `round(128.5,-2)` kao rezultat daje `100.0`.
- **Modul `math`**
  - Osnovne matematičke funkcije dostupne su u okviru modula `math`.
  - Njihovo učitavanje se može obaviti naredbom `import math`.
  - Listu učitanih funkcija možemo dobiti komandom `dir(math)`. Šta radi funkcija `dir`?

- Pogledajmo spisak funkcija iz modula **math**:

```
>>> import math
>>> dir(math)
['__doc__','__loader__','__name__','__package__',
'__spec__','acos','acosh','asin','asinh',
'atan','atan2','atanh','ceil','copysign',
'cos','cosh','degrees','e','erf','erfc','exp',
'expm1','fabs','factorial','floor','fmod',
'frexp','fsum','gamma','gcd','hypot','inf',
'isclose','isfinite','isinf','isnan','ldexp',
'lgamma','log','log10','log1p','log2','modf',
'nan','pi','pow','radians','remainder','sin',
'sinh','sqrt','tan','tanh','tau','trunc']
```

- Sadržaj modula **math**
  - Uočiti da se pojavljuju nazivi fajlova koji počinju sa __
  - Obratiti pažnju na Euler-ov broj **e** i konstantu **pi**, odnosno, broj π.
  - Trigonometrijske funkcije, **sin**, **cos**, **tan** (argumenti su u radijanima)
  - Inverzne trig. funkcije, **asin**, **acos**, **atan**
  - Hiperboličke funkcije, **sinh**, **cosh**, **tanh**

- Sadržaj modula `math` – nastavak
  - Eksponencijalna funkcija `exp`
  - Logaritamske funkcije `log`, `log2`, `log10`,
  - Kvadratni korijen `sqrt`
  - Stepen, `pow(x,y,z=None)`, treći argument je opcion, `x**y%z`
  - Faktorijel `factorial`
  - Zaokruživanje `ceil`, `floor`
  - Specijalne konstante `inf`, `nan`
  - Najveći zajednički djelilac `gcd`
  - Funkcija za konverziju radijana u stepene `degrees`
  - Funkcija greške `erf` i inverzna funkcija greške, `erfc`
- Obratiti pažnju da se učitavanje modula može obaviti na više načina:

1. ```
   from math import sin, pi, exp
   # * umjesto naziva - ucitava se cio paket
   ```

   kada se funkcije/konstante koriste u obliku u kojem su učitane, `sin(pi/2)` daje `1.0`

2. ```
   import math
   ```

   kada se gornji primjer mora odraditi u obliku: `math.sin(math.pi/2)`

Selekcija

```
Selekcija
if (uslov):
     Naredna ili grupa naredbi
elif uslov:
     Naredba ili grupa naredni
else:
     Naredba ili grupa naredbi
```

- Operatori poređenja su dati u tabeli ispod:

Operacija	Značenje
`a>b`	tačno ako je $a > b$
`a>=b`	tačno ako je $a \geq b$
`a<b`	tačno ako je $a < b$
`a<=b`	tačno ako je $a \leq b$
`a==b`	tačno ako je $a = b$
`a!=b`	tačno ako je $a \neq b$

Crash kurs Python-a

- Rezultat dijeljena dva cijela broja je podrazumijevano cijeli broj u starijim verzijama, u novijim nije

```
In [21]: 5/2
Out[21]: 2.5

In [22]: 5//2
Out[22]: 2
```

Osnovni operatori

Operator	Značenje
+	sabiranje
-	oduzimanje
*	množenje
/	dijeljenje
**	stepenovanje
//	cjelobrojno dijeljenje
%	ostatak pri dijeljenju

zadatak 1

- Odrediti koliko je od prvih 100 prirodnih brojeva takvih da im se kvadrat se završava cifrom 4.

```
broj = 0
for br in range(1,100):
  if(br**2 % 10) == 4:
    broj = broj + 1
  #kraj if-a
#kraj for-a
print("Ima ih", broj)
```

```
Ima ih 20
```

zadatak 1b

- Odrediti sumu prvih 100 prirodnih brojeva čiji kvadrat se završava cifrom 4.

```
[2]  suma = 0
     for br in range(1,100):
       if br**2 % 10 == 4:
         suma = suma + br
         print("broj je ", br, "Kvadrat mu je", br**2)
     print("suma brojeva je: ", suma)
```

```
broj je  2 Kvadrat mu je 4
broj je  8 Kvadrat mu je 64
broj je  12 Kvadrat mu je 144
broj je  18 Kvadrat mu je 324
broj je  22 Kvadrat mu je 484
broj je  28 Kvadrat mu je 784
broj je  32 Kvadrat mu je 1024
broj je  38 Kvadrat mu je 1444
broj je  42 Kvadrat mu je 1764
broj je  48 Kvadrat mu je 2304
broj je  52 Kvadrat mu je 2704
broj je  58 Kvadrat mu je 3364
broj je  62 Kvadrat mu je 3844
broj je  68 Kvadrat mu je 4624
broj je  72 Kvadrat mu je 5184
broj je  78 Kvadrat mu je 6084
broj je  82 Kvadrat mu je 6724
broj je  88 Kvadrat mu je 7744
broj je  92 Kvadrat mu je 8464
broj je  98 Kvadrat mu je 9604
suma brojeva je:  1000
```

zadatak 2

- Napisati fajl Niz kojim se unosi niz X i broj N i koji određuje i štampa koliko se puta broj N pojavljuje u nizu X. (`input` očekuje string kao argument, da bismo ga konvertovali u željeni tip – izvršili izraz pod stringom, koristimo `eval`)

```
x = eval(input('Unesi niz'))
N = eval(input('Unesi broj'))
broj = 0
for br in x:
  if(br == N):
    broj = broj +1;
print("Pojavljuje se", broj, " puta")
```

```
Unesi niz[1,2,6,12,25,56,8,2]
Unesi broj2
Pojavljuje se 2  puta
```

zadatak 3

Kreirati fajl kojim se unosi niz cijelih brojeva **x** i cio broj *n*. Algoritam na izlazu daje sumu onih elemenata niza čiji **broj** cifara sabran sa *n* daje broj 10.

```
x = eval(input('Unesite niz cijelih brojeva'))
n = eval(input('Unesite cio broj'))
suma = 0; #suma elemenata niza koji zadovoljavaju uslov
for el in x:
  br_cif = 0;
  pom = el; #dijelicemo ga s 10 dok ne postane 0
  # a treba nam kasnije da ga sumiramo ako bude
  # ispunjavao trazeni uslov
  while pom != 0:
    pom = pom//10 #cijeli dio dijeljenja //
    br_cif = br_cif +1
  if br_cif + n == 10:
    suma = suma + el
print('Suma zeljenih elemenata je ', suma)
```

```
Unesite niz cijelih brojeva[123,25,564,2,5897,124]
Unesite cio broj7
Suma zeljenih elemenata je  811
```