TUTORIAT 2

Linkuri utile:

- https://www.python.org/
- https://docs.python.org/3/
- https://www.codecademy.com/learn/learn-python-3
- https://www.learnpython.org/
- https://stanfordpython.com/#/

Comentariu multiplu PyCharm: CTRL + /

Ce conține tutoriatul ?

I. Despre secvențe
II. Șiruri de caractere (**clasa str**)
    1. Creare
    2. Accesare elemente și subsecvențe
    3. Metode uzuale
    4. Formatarea șirurilor de caractere
III. Exerciții

I.Despre secvențe

O **secvență** reprezintă un obiect care reunește mai multe elemente indexate și permite memorarea unui volum mare de date, nu neapărat de același tip. (secvență -> set ordonat de elemente)

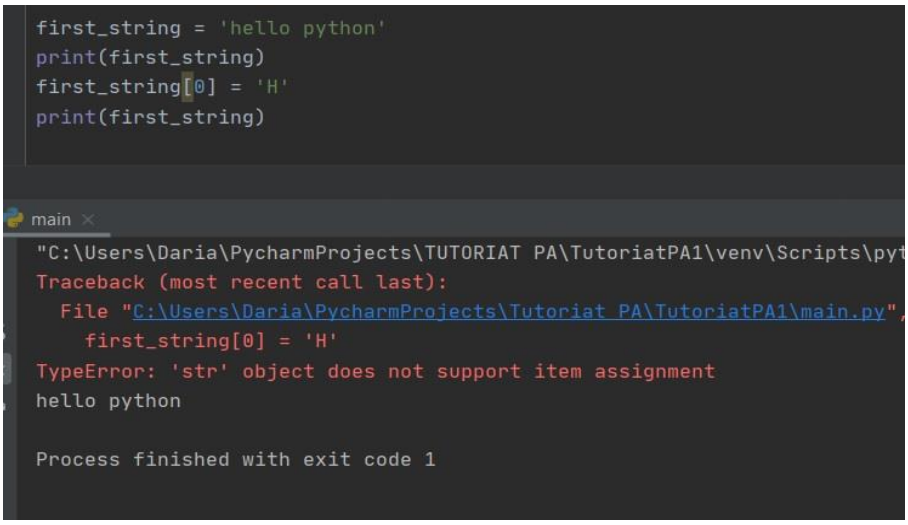
În python cele mai uzuale secvențe sunt: listele, tuplurile, **șirurile de caractere**, obiectele Xrange.

Secvențele se împart în două categorii:

1. **MUTABILE** (li se poate modifica valoarea după creare) : listele
2. **IMUTABILE** (nu li se poate modifica valoarea după creare): tupluri, **șiruri de caractere**

II.Șiruri de caractere (clasa STR)

➔ Secvențe IMUTABILE
➔ **Exemplu:**



II.1. Creare

Șirurile de caracetre sunt delimitate de: ' … ', "…" ; ' ' ' … ' ' ', """…""" . Pot fi create și cu ajutorul constructorului **str().**

Exemple:

first_string = 'Welcome Tutoriat2 '

second_string = "It`s time to study ."

third_string = ' ' ' It`s time to have fun ! ' ' '

another_string = str(1.44)

II.2.Accesare elemente și subsecvențe

Pentru a accesa elementele unui șir de caractere se utilizează [ ]. **!ATENȚIE!** Indexarea începe de la 0.

Exemple:

first_string[i] => elementul de pe poziția i din șirul first_string , (începând cu poziția 0) **!ATENȚIE!** i poate fi și negativ.

first_string[0] => W **!ATENȚIE!** NU EXISTĂ tipul CHAR, first_string[0] este de tip STR.

Tip accesare	Rezultat
nume_sir[i]	elementul de pe poziția i
nume_sir[i:j]	elementele de pe pozițiile i, i+1,..,j-1
nume_sir[:j]	elementele de pe pozițiile 0,1,..,j-1
nume_sir[i:]	elementele de pe pozițiile 0,1,..,len(nume_sir)-1
nume_sir[:]	toată secvența
nume_sir[i:j:k]	elementele de pe pozițiile i,i+k,i+2k,….

II.3. Metode uzuale

Concatenări: +, *

Exemplu: s1 = "Facultatea de Matematica "

s2 = "si Informatica"

s1 = s1 + s2 => s1 = " Facultatea de Matematica si Informatica"

!ATENȚIE! se creează un obiect nou, nu adaugă la șirul inițial s1

s1 = "apa"; n = 3

s1 * n sau n *s1 => apaapaapa

!ATENȚIE! pentru n negativ sau n = 0 se creează o secvență vidă

Funcții comune tuturor secvențelor: **len**(nume_sir) => returnează lungimea șirului de caractere, **min**(nume_sir) => returnează caracterul care apare primul în ordine alfabetică, **max**(nume_sir) => returnează caracterul care apare ultimul în ordine alfabetică

Apartenența: in, not in

Exemplu: nume_sir = "Este luni, ne aflam la tutoriat."

```
if 'z' not in nume_sir:
    print("z nu se gaseste in sir")
if "ne" in nume_sir:
    print("Secventa ne se gaseste in sir")
```

Metode (funcție a unei clasei)

Metoda	Apel	Efect
index	s1.index(s,[,i[,j]])	returnează prima apariție a șirului s (începând cu poziția i, până la j-1, dacă sunt specificați) **ValueError** dacă nu există
rindex	s1.rindex(s,[,i[,j]])	returnează ultima apariție a șirului s (începând cu poziția i, până la j-1, dacă sunt specificați) **ValueError** dacă nu există
count	s1.count(s)	returnează numărul de apariții al șirului s în șirul s1
find (rfind)	s1.find(s,[,i[,j]])	returnează același rezultat ca **index**, dar -1 dacă s nu este găsit în s1
startwith	s1.startwith(s)	returnează True/False dacă șirul s1 începe cu șirul s
endwith	s1.endwith(s)	returnează True/False dacă șirul s1 se termină, respectiv nu se termină cu șirul s
replace	s1.replace(old, new)	returnează o copie a lui s1, cu aparițiile lui old înlocuite cu new
lower	s1.lower()	returnează șirul scris cu minuscule
upper()	s1.upper()	returnează șirul scris cu majuscule
capitalize()	s1.capitalize()	returnează șirul scris cu primul caracter majusculă
strip (rstrip, lstrip)	s1.strip([caractere])	returneză un șir format din eliminarea "caractere" de la începutul și finalul șirului s1; implicit caractere = caracterele albe
islower (isupper)	s1.islower()	returnează True/False dacă literele din s1 sunt, respectiv nu sunt minuscule
isdigit	s1.isdigit()	returnează True/False dacă literele din s1 sunt, respectiv nu sunt cifre
split	s1.split(sep = None, maxsplit = -1)	returnează lista cuvintelor din s1 delimitate de **sep;** dacă masxsplit este specificat se împarte s1 în maxim **maxsplit+1 cuvinte**
join	s1.join(iterabil)	returnează șirul format din concatenarea șirurilor din **iterabil**, având ca separator între aceste șiruri șirul s

II.4.Formatarea șirurilor de caractere

template.format(<positional_arguments>, <keyword_arguments>)

- template -> șirul de formatat; conține secvențe speciale, delimitate de { }
- secvențele dintre parantezele { } vor fi înlocuite cu parametrii metodei **format**
- este returnat șirul template formatat

Exemple: sir = "Numele meu este {} si am {} ani".format("Andrei", 12)

print(sir) => Numele meu este Andrei si am 12 ani

f-stringuri

În câmpurile de formatare se pot folosi nume de variabile sau expresii.

Exemplu: s = f "Tutoriat {materie} semestrul {numar} , anul {anul_calendaristic}"

s = "Tutoriat {} semestrul {} , anul {}".format(materie, numar, anul_calendaristic)

Specificare format la afișare : d: (decimal), f: (virgulă mobilă, implicit 6 cifre după virgulă), b,o,x (întreg în baza 2, 8, 16)

Exemplu:

suma = 6.4 + 5.9

print('{:f}'.format(z)) => 10.300000

print('{:.2f}'.format(z)) => 10.30

III.Exerciții

1.Se citesc de la tastatură două șiruri de caractere, s1 și s2. Se cere să se construiască un șir, s3 format din primul caracter al șirului s1, urmat de ultimul caracter al șirului s2, al doilea caracter din șirul s1, penultimul caracter din șirul s2 și așa mai departe. Orice caracter rămas se plasează la sfârșitul șirului s3. Exemplu: s1 = "xyz", s2 = "abc" => s3 = xcybza.

2. Se dau două șiruri s1 și s2. Se cere să se afișeze de câte ori apare s2 în s1, ignorând modalitatea de scriere a șirului s2. Exemplu: s1 = "AFARA este insorit; copiii vor afara.", s2 = "afara" => Numarul aparitiilor lui s2 in s1 = 2

3.Se citește de la tastură un șir de caractere. Să se calculeze suma, produsul și media aritmetică a cifrelor care apar în șirul dat. Exemplu: s = "Py29*#815"; Suma este: 25. Produsul este: 720. Media aritmetica a cifrelor este: 5.

4. Se dă un șir s1. Să se înlocuiască fiecare caracter special cu simbolul *. Exemplu: s = "/*Is it an #$% important @ day ?!" => **Is it an ***important #*day **

5. Se citește de la tastatură un șir de caractere. Să se insereze șirul s la mijlocul șirului citit. Exemplu: sir = "Care", s = "lculatoa" => sir = "Ca**lculatoa**re"

6. Să se scrie un program care citește un șir format din mai multe cuvinte, separate printr-unul sau mai multe spații și îl modifică astfel încât fiecare cuvânt să înceapă cu literă mică. Exemplu: s = "Programarea este o Disciplina de Viitor" => s = "programarea este o disciplina de viitor"

7.Să se scrie un program care elimită caracterele de pe pozițiile impare dintr-un șir dat. Exemplu: s = "abcdef" => s = "ace"

8.Screți un program care afișează numerele: 3.1415926 și 14.9999 rotunjite cu două zecimale. Rezultat: 3.14, 15.

9. Numele Pre-Nume

Să se scrie un program care citește un șir și decide dacă este sau nu un nume corect. Un nume este considerat corect dacă îndeplinește următoarele proprietăți: orice nume sau prenume:

- ➔ conține doar litere și cel mult o cratimă
- ➔ este format din minim 3 litere
- ➔ începe cu o majuscule;
- ➔ prenumele sunt cel mult două; în cazul în care sunt două, atunci sunt despărțite prin cratimă

10. Se dau două șiruri. Să se efectueze următoarele operații condiționate:

- ➔ dacă ultimul caracter al primului șir este identic cu primul caracter din al doilea șir, atunci se concatenează cele două șiruri, păstrând doar unul dintre cele două caractere identice;
- ➔ dacă al doilea caracter din primul șir este identic cu penultimul caracter din al doilea șir, atunci cele două șiruri sunt concatenate, fără alte condiții suplimentare.

Exemplu: "mere" "eu" => mereu; "acasa" "foca" => acasafoca