IT SCHOOL
from zero to hero

Python Session 9

Today

- List comprehension
- Exercises
- Functions
- Exercises
- Homework

List comprehension

- List comprehension ne ofera o sintaxa folosind o singura line de cod cand dormi sa creem o lista noua bazandu-ne pe o lista(obiect iterabil) deja existenta

- Syntaxa
  - newlist = [expression for item in iterable if condition == True]

List comprehension

newlist = [expression for item in iterable if condition == True]

- Conditia este optionala, poate sa fie vazuta precum un filtru
- Iterable poate sa fie orice obiect iterabil: list, tuple, set etc.
- Expresia este itemul curent din iteratie care o sa fie adaugat la lista noua. Acest item poate sa fie manipulat inainte sa ajung un item in lista noua
- Putem folosi inclusiv if else ca sa manipulam expresia

Example

- Creeaza o lista noua cu frute care au “a” in nume
  - Fara list comprhension :

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
newlist = []

for x in fruits:
  if "a" in x:
    newlist.append(x)

print(newlist)
```

Example

- Cu list comprehension:

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

newlist = [x for x in fruits if "a" in x]

print(newlist)
```

- Manipularea expresiei cu if else :

```
newlist = [x if x != "banana" else "orange" for x in fruits if "a" in x]
```

Exercises

- Creeaza o lista in care sa ai toate numerele de la 0 - 1000
  - Creeaza o noua lista in care sa avem numerele divizibile cu 8
  - Creeaza o noua lista in care apar doar numerele care contin 6
- Pornind de la lista fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
  - Creeaza o noua lista in care avem fructele cu litere mari
- Avem lista my_list =[0, 22, 41, 63, 82, 101, 122, 143, 167, 181]
  - Creeaza o noua lista folosind for in care sa adaugam doar elementele pare
  - Creaza o noua lista folosind list comprehension in care sa fie doar elementele pare
  - Creeaza o lista folosind for loops in care daca elementul este par adaugam “Even” altfel daca elementul este impar adaugam “Odd”
  - Creeaza o lista folosind list comprehension in care daca elementul este par adaugam “Even” altfel daca elementul este impar adaugam “Odd”

Functions

- ## Functie

  O functie sau un subprogram (subrutina) este un set de instructiuni sub un nume.

  Faptul ca noi putem evita sa rescriem cod si putem refolosi o functie deja scrisa, de mai multe ori, este unul din cele mai indicate modalitati de a programa.

  DRY principle - Don't Repeat Yourself

  In loc sa scriem o bucata de cod din nou, putem pur si simplu sa apelam (sa executam) o functie care contine acea bucata de cod.

- ## Exemple de functii built-in

  Functii built-in sunt functii deja existente in Python:

  - print()

  - len()

User-defined functions

- Syntax

  def nume(argumente):

  Instructiune

- Rezultatul

  Daca avem nevoie de rezultatul functiei, cu ajutorul keyword-ului 'return' putem oferi rezultatul functiei catre bucata de cod care a apelat acea functie.

  Return nu inseamna ca printam.

  ex:

  def nume():

  c = a + b

  return c

Apelarea functiilor

- Putem apela functia de oriunde din program cat timp o apelam dupa definire.
  Pentru a apela o functie mentionam numele functiei, urmat de doua paranteze rotunde:

  functie()

- Daca in momentul definirii functiei, am decis ca avem nevoie de parametri, cand apelam functia trebuie sa oferim si argumentele necesare separate prin virgula.

Arguments & Parameters

- Intr-o discutie normala majoritatea oamenilor se refera la acelasi lucru cand vorbesc despre argumente sau parametri.
- Parametrii sunt declarati in interiorul functiei.

  Putem considera parametrii ca fiind variabile goale, care sunt definite in interiorul functiei.

- Argumentele sunt valorile oferite functiei la momentul apelarii.

  In momentul apelarii, parametrii unei functii preiau valorile argumentelor folosite, in ordinea precizata.

Default Arguments

- In cadrul functiilor, putem sa avem definit si o valoare default a argumentului, daca acesta nu este precizat

```
def my_function(country = "Norway"):
  print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
```

Scopes & Variables

- Scope

  Scopul unei variabile este portiunea de cod unde variabila este recunoscuta.

- Local scope

  Daca o variabila este definita inauntrul unei functii, acea variabila nu este vizibila in afara functiei, asta inseamna ca variabila are scop local.

  Daca definim o variabila intr-o functie, durata vietii acelei variabile este cat timp se executa functia.

  Odata cu finalizarea functiei, variabilele sunt distruse, iar la urmatorul apel al functiei, Python nu isi va aminti valorile precedente ale acelor variabile.

- Global scope

  Cat timp o variabila este definita in afara oricarui bloc de cod, putem considera ca variabila are scop global, fiind accesibila din orice bloc de cod.

Exercises

- Sa scriem 4 functii, pentru a calcula:

  adunari

  scaderi

  inmultiri

  impartiri

- Sa scriem o functie:

  Functia poate primi doua argumente:

  - primul este un string in care vom cauta

  - al2lea este un string care il vom cauta in primul string

  Daca s-a gasit string in string, returnam True si printam 'S-a gasit {stringul_cautat}'.

  In caz contrar returnam False si printam 'Nu s-a gasit'.

Exercises

- Scrie o functie care sa returneze daca elementul primit este par
- Scrie o functie care sa iti returneze o lista cu toate elementele pare pana la limita primita ca si argument
- Scrie o functie care sa calculeze suma unei liste de numere

Homework - List comprehension

- Create a list with elements from 0 to 100 using list comprehension
- Given the list lst1=[1,2,3,4,5] create an identical list from the first list using list comprehension.
- Given a list lst1=[2, 4, 6, 8, 10, 12, 14], using list comprehension, construct a new list from the squares of each element in the list lst1.
- Given a list of numbers original_list = [2,3.75,.04,59.354,6,7.7777,8,9], create a new list without float numbers using list comprehension

Homework - Functions

- Create a function to reverse a string.
- Create a function that accepts a string and calculate the number of upper case letters.
- Create a function that squares up a list of numbers and returns the new list with the numbers squared
- Create a function that removes all vowels from a string given as argument and returns the new string.
- Incearca sa rescrii quiz game-ul folosind functii (unde este posibil).