IT SCHOOL
from zero to hero

Python Session 4

For today

- Homework
- Tuples
- Exercises
- Sets
- Exercises

Tuples

- Tuples sunt folosite sa salveze multiple date in aceeasi variablia.
- Tuples sunt una dintre cele 4 modalitati de a tine si de a folosi o colectie de date in Python.
- Un tuple se creeaza folosind paranteze () si despartind elementele prin virgula
  thistuple = ("apple", "banana", "cherry")
  print(thistuple)
- Pentru a crea un tuple putem sa folosim si functia constructor tuple()

Tuples

- Tuples nu se pot schimba dupa creeare
  - Nu putem sa updatam valorile
  - Nu putem adauga elemente
  - Nu putem sterge elemente

- Tuples permite elemente duplicate

- Tuples permite elemente de tipuri diferite

Tuple Items

- Putem accesa elementele unui tuple precum facem si la liste folosind indexul elementului
  - thistuple = ("apple", "banana", "cherry")
  - print(thistuple[1])

- Putem sa folosim indexi negativi thistuple[-1]

- Putem sa folosim range de indexi thistuple[1:3]

- Verificam daca un element se regaseste in tuple folosind operatorul “in”

Tuple Unpaking

- Cand creem un tuple, si asignam valori se numeste “paking” a tuple

```
fruits = ("apple", "banana", "cherry")
```

- Python ne lasa sa “unpack” a tuple, si anume sa asignam valorile din tuple direct variabilelor

```
fruits = ("apple", "banana", "cherry")
(green, yellow, red) = fruits
print(green)
print(yellow)
print(red)
```

Tuple Unpaking

- Folosind Asterisk*
  - Daca numarul variabilelor este mai mic decat numarul valorilor, putem sa folosim * la numele variabilei si restul valorilor o sa fie adaugate variabilei sub forma de lista

```
fruits = ("apple", "banana", "cherry", "strawberry","raspberry")

(green, yellow, *red) = fruits

print(green)
print(yellow)
print(red)
```

Tuple Unpaking

- Folosirea * pus in numele altei variabile, nu in ultima o sa rezulte in Python adaugand valori in lista pana cand valorile ramase sunt egale cu variabilele ramase

```
fruits = ("apple", "mango", "papaya", "pineapple", "cherry")

(green, *tropic, red) = fruits

print(green)
print(tropic)
print(red)
```

Updating a tuple

- Pentru ca un tuple nu poate sa fie modificat dupa creare, nu putem in mod direct sa il updatam.
- Workaround:
  - Transformare in lista
  - Update in lista
  - Transformare inapoi in tuple

```
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)

print(x)
```

Joining Tuples

- Putem alatura 2 tuples folosind operatorul de adunare + , rezultand un nou tuple

```
tuple1 = ("a", "b" , "c")
tuple2 = (1, 2, 3)

tuple3 = tuple1 + tuple2
print(tuple3)
```

Multiplying Tuples

- Folosind operatorul de inmultire *, precum la string o sa ne fie repetate elementele tupelului in functie de numarul cu care inmultim

```
fruits = ("apple", "banana", "cherry")
mytuple = fruits * 2

print(mytuple)
```

Tuples Methods

- La tuples avem doar 2 metode built in si anume:
  - count() - Returneaza numarul de aparitii al unei valori in tuple
  - index() - Returneaza pozitia unde a fost gasita o anumita valoare

Exercitii

- Write a Python program to unpack a tuple in several variables
- Write a Python program to add an item in a tuple.
- Write a Python program to join two tuples
- Write a Python program that prints if a value is present in a tuple
- Access value 20 from the tuple
  - tuple1 = ("Orange", [10, 20, 30], (5, 15, 25))

Sets

- Seturile sunt folosite sa salveze multiple date in aceeasi variablia.
- Seturile sunt una dintre cele 4 modalitati de a tine si de a folosi o colectie de date in Python.
- Un set se creeaza folosind {}
  - thisset = {"apple", "banana", "cherry"}
- Pentru a crea un set putem sa folosim si functia constructor set(("apple", "banana", "cherry"))

Sets

- Seturile suporta tipuri de date multiple
- Seturile nu suporta elemente duplicate
- Seturile nu sunt indexate, asadar nu putem accesa un element folosind indexul
- Seturile nu au o ordine

Adaugarea in seturi

- Odata creat un set ,nu putem sa modificam valorile existente dar putem sa adaugam altele noi.
  - Folosind metoda add() adaugam un element

```
thisset = {"apple", "banana", "cherry"}
thisset.add("orange")
print(thisset)
```

  - Folosind metoda update () adaugam un nou set sau orice obiect iterabil (tuples, lists, etc.).

```
thisset = {"apple", "banana", "cherry"}
mylist = ["kiwi", "orange"]

thisset.update(mylist)
print(thisset)
```

Eliminare elemente din seturi

- Putem sa eliminam elemente din seturi folosind metoda remove() sau discard()

```
thisset = {"apple", "banana", "cherry"}

thisset.remove("banana")
thisset.discard("banana")
print(thisset)
```

- Metoda remove() o sa arunce o exceptie daca nu gaseste elementul
- Metoda discard() nu o sa arunce nimic daca nu gaseste elementul

- Se poate folosii si metoda pop(), dar nu este recomandata deoarece setul nu e ordonat => nu stim ce scoatem

Sets methods

- add() - Adds an element to the set
- clear() - Removes all the elements from the set
- copy() - Returns a copy of the set
- difference() - Returns a set containing the difference between two or more sets
- difference_update() - Removes the items in this set that are also included in another, specified set
- discard() - Remove the specified item
- intersection() - Returns a set, that is the intersection of two other sets
- intersection_update() - Removes the items in this set that are not present in other, specified set(s)
- isdisjoint() - Returns whether two sets have a intersection or not
- issubset() - Returns whether another set contains this set or not
- issuperset() - Returns whether this set contains another set or not
- pop() - Removes an element from the set
- remove() - Removes the specified element
- symmetric_difference() - Returns a set with the symmetric differences of two sets
- symmetric_difference_update() - Inserts the symmetric differences from this set and another
- union() - Return a set containing the union of sets
- update() - Update the set with the union of this set and others

Exercitii

- Write a Python program to create a set.
- Write a Python program to add member(s) in a set.
- Write a Python program to check if a set is a subset of another set.
- Write a Python program to find the elements in a given set that are not in another set.
  - sn1 = {1,2,3,4,5}
  - sn2 = {4,5,6,7,8}

Next session

1. Dictionare
2. Exercitii