ENSIAS

جامعة محمد الخامس بالرباط
Université Mohammed V de Rabat

Langage Python

Lecture4: Contrôle de flux

Abderrahim MESBAH
a.mesbah@um5r.ac.ma

Plan

- Bloc d'instructions
- Condition
- If-then-else
- For
- While

Bloc d’instructions

Bloc d'instructions

- **Pour identifier les instructions composées, Python utilise la notion d'indentation significative. Cette syntaxe, légère et visuelle, met en lumière un bloc d'instructions et permet d'améliorer grandement la présentation et donc la lisibilité des programmes sources.**

```
if condition:
        Instruction1
        …
        InstructionN

else :
        Instruction1
        …
        InstructionM

Instructions
```

Condition

Condition

- **Les valeurs de python sont des objets spéciaux True et False encastrables:**

```
In [1]: a = True
In [2]: type(a)
Out[2]: bool


In [3]: b=False
In [4]: type(b)
Out[4]: bool
```

Condition

- **Evaluation d'une expression logique: Opérateur and**

```
In [1]: True and True
Out[1]: True

In [2]: True and False
Out[2]: False

In [3]: a=True
In [4]: b=False
In [5]: c=a and b
In [6]: c
Out[6]: False
```

Condition

- **Evaluation d'une expression logique(Prédicat): Opérateur or et not:**

```
In [5]: True or False
Out[5]: True

In [6]: False or False
Out[6]: False

In [7]: not True
Out[7]: False

In [8]: True and not False
Out[8]: True
```

Condition

- **Evaluation d'une expression logique: Opérateurs de comparaison**

```
In [9]: x=35
In [10]: x>15
Out[10]: True

In [11]: x>42
Out[11]: False

In [12]: x==42
Out[12]: False

In [13]: not x==42 #x!=42
Out[13]: True

In [15]: x>35
Out[15]: False

In [16]: x>=35
Out[16]: True
```

If-then-else

If-then-else

- **L'instruction if permet l'exécution conditionnelle du code.**

```
In [18]: a = 34
   ...:     if a > 0:
   ...:         print("a is positive ")
   ...:
a is positive
```

- **L'instruction if peut également avoir une branche else qui est exécutée si la condition est fausse:**

```
In [19]: a = 34
   ...:     if a > 0:
   ...:         print ("a is positive ")
   ...:     else :
   ...:         print ("a is non - positive (i.e. negative or zero )")
   ...:
a is positive
```

If-then-else

- **Le mot-clé elif (lu comme \ else if ") permet de vérifier plusieurs possibilités :**

```
a = 17
if a == 0:
        print ("a is zero ")
elif a < 0:
        print ("a is negative ")
else :
        print ("a is positive ")
```

For

For

- **La boucle for permet de parcourir une séquence (ex: une chaîne ou une liste).**

```
In [21]: for animal in ['dog ','cat ','mouse ']:
   ...:         print (animal , animal . upper ())
   ...:
   ...:
dog DOG
cat CAT
mouse MOUSE
```

```
In [22]: for i in range (5 ,10):
   ...:         print (i)
   ...:
5
6
7
8
9
```

While

While

- **Le mot-clé while permet de répéter une opération lorsqu'une condition est vraie.**

```
In [25]: mymoney = 100
   ...:     rate = 1.05
   ...:     years = 0
   ...:     while mymoney < 200:
   ...:         mymoney = mymoney * rate
   ...:         years = years + 1
   ...:     print ('We need ', years , 'years to reach ', mymoney , 'pounds .')

We need 15 years to reach 207.89281794113688 pounds .
```

ENSIAS

Exercice 1 :

Quel serait le résultat du code suivant ?

- i = 10
  while i > 1:
  print (i)
  i /= 2
- i = 0
  value = 0
  while value < 20:
  value += i
  i += 3
  print(value)
- for i in range(4):
  print (i)
- for i in range(3,5):
  print (i)
- for i in range (1,10,3):
  print (i)
- for i in range (1, 10, -3):
  print (i)
- for i in range (10, 1, -3):
  print (i)

Exercice 2:

Écrivez du code pour effectuer chacune des opérations suivantes :

a. Obtenez un nombre de l'utilisateur, puis compter de 1 à ce nombre. Essayez de l'écrire en utilisant à la fois une boucle while et une boucle for.

b. Convertissez la boucle while suivante en boucle for.

```
i = 2
while(i<7):
    print(i)
    i = i + 3
```

c. Écrivez un programme qui définit un nombre de 1 à 10, puis continuez à demander à l'utilisateur de deviner ce nombre jusqu'à ce que le nombre correct soit deviné.