Université de Tlemcen
Faculté des Sciences

1ère licence SM
Année Universitaire
2019-2020

Matière : Informatique 2 (S2)

TP N°1: Structures répétitives: boucle for (2)

Semaine 2 (SM5, SM6, SM7 et SM8)

Rappel:

for **compteur** in range (**valeur de départ**, **valeur d'arrêt**, **pas)**:

    instruction 1
    instruction 2
    .
    .
    instruction N

Remarques:

1. **compteur** est une variable qui va recevoir les valeurs de **valeur de départ** jusqu'à **valeur d'arrêt** selon le pas qui lui est donné. La première valeur donné au compteur est **valeur de départ** ensuite **compteur=compteur+pas** jusqu'à ce que **compteur** arrive à la **valeur d'arrêt**.
   A chaque fois que le compteur reçoit une valeur, les instructions qui suivent la boucle à savoir: **instruction1, instruction2, …instruction N** sont exécutées.

2. Si on utilise un **pas positif** :
   valeur de départ < valeur d'arrêt
   La dernière valeur que va recevoir le compteur est **valeur d'arrêt-1 (si vous voulez allez jusqu'à 10 il faut mettre 11 comme valeur d'arrêt)**
   Exemple: **for i in range (0,11,2):** i reçoit 0 ensuite 2 ensuite 4 ensuite 6 ensuite 8 ensuite 10 car le pas est 2 et la valeur d'arrêt est: 11-1 et donc 10.

3. Si on utilise un **pas négatif**:
   valeur de départ > valeur d'arrêt.
   la dernière valeur que va recevoir le compteur est valeur d'arrêt+1 (si vous voulez allez de 10 jusqu'à 0 il faut mettre -1 comme valeur d'arrêt).
   Exemple: **for i in range (10,-1,-2):**i reçoit 10 ensuite 8 ensuite 6 ensuite 4 ensuite 2 ensuite 0 car le pas est -2 et la **valeur d'arrêt** est -1+1 et donc 0

4. Si on ne précise pas le pas par exemple **for i in range (0,10):** le pas est considéré égale à 1.

5. Si on ne précise pas la valeur de départ par exemple **for i in range (10):** la première valeur que prendra le compteur est 0 et le pas est considéré égale à 1.

Exercice 1: Exécutez chaque boucle for présentée dans le tableau ci-dessous et notez la valeur de départ, la valeur d'arrêt et le pas correspondant.

Boucle	Valeur de départ	Valeur d'arrêt	Pas
for i in range(5,20): print(i)	**5**	**19**	**1**
for i in range(8): print(i)	**0**	**7**	**1**
for i in range(-5,11,2): print(i)	**-5**	**10**	**2**
for i in range(15,0,-4): print(i)	**15**	**1**	**-4**

- ✓ **La valeur d'arrêt correspond à la (valeur limite) de l'intervalle de valeurs que peut recevoir le i.**
- ✓ **Dans les deux premiers exemple la borne supérieur correspond à la dernière valeur affiché pour le i.**
- ✓ **Dans la troisième boucle** 10 correspond à la valeur limite que peut avoir le i (borne supérieur) par contre la dernière valeur affiché pour le i est 9.
- ✓ **Dans la boucle** for i in range(15,0,-4) : 1 correspond à la valeur limite que peut prendre le i le pas étant négatif 1 est la borne inférieur de l'intervalle de valeurs pour le i ; la dernière valeur affiché pour le i est 3.

Exercice 2: Testez chaque boucle for dans le tableau qui suit déduisez la valeur de départ et la valeur d'arrêt. Si le pas est positif déduisez la boucle correspondante avec un pas négatif et réciproquement.
Les deux boucle doivent afficher les meme valeur, il n'ya que l'ordre (croissant/décroissant) d'apparition des valeurs qui change.

Boucle avec un pas positif	V D	V A	**Boucle avec un pas négatif**	VD	VA
for i in range(0,15): print(i)	**0**	**14**	**for i in range(14,-1,-1): print(i)**	**14**	**0**
for i in range(0,31): print(i)	**0**	**30**	for i in range(30,-1,-1): print(i)	**30**	**0**
for i in range(-100,101,2): print(i)	**-100**	**100**	**for i in range(100,-101,-1): print(i)**	**100**	**-100**
for i in range(-29,-19,3): print(i)	**-29**	**-20 correspond à la borne supérieur (valeur limite de i)**	for i in range(-20,-31,-3): print(i)	**-20**	**-30 correspond à la borne inférieur pas à la dernière valeur affiché pour le i**

Exercice 3:
Ecrire un programme qui demande à l'utilisateur de saisir un entier N, le programme doit afficher ensuite:

1) Si N est **positif** et **pair**: les ***entiers positifs pairs inférieurs*** ou égale à N inclus dans [0,N], dans l'ordre croissant.
   Le programme doit afficher la somme **S1** des entiers positifs pairs inclus dans [0,N].
2) Sinon si N est **négatif** et **impair**: les ***entiers négatifs impairs supérieur*** ou égale à N inclus dans [N,-1], dans l'ordre décroissant.
   Le programme doit afficher la somme **S2** des entiers négatifs inclus dans [N,-1].

Par exemple:

- ✓ si l'utilisateur saisit N=10 le programme doit afficher les valeurs: 0 2 4 6 8 10 ainsi que la somme des valeurs S1= 30
- ✓ si l'utilisateur saisit N=-9 le programme doit afficher les valeurs: -1 -3 -5 -7 -9 ainsi que la somme des valeurs S2=-25

Exercice3.py - C:\Python 2019-2020\EXOS PYTHON\Exercice3.py (3.6.3)

File Edit Format Run Options Window Help

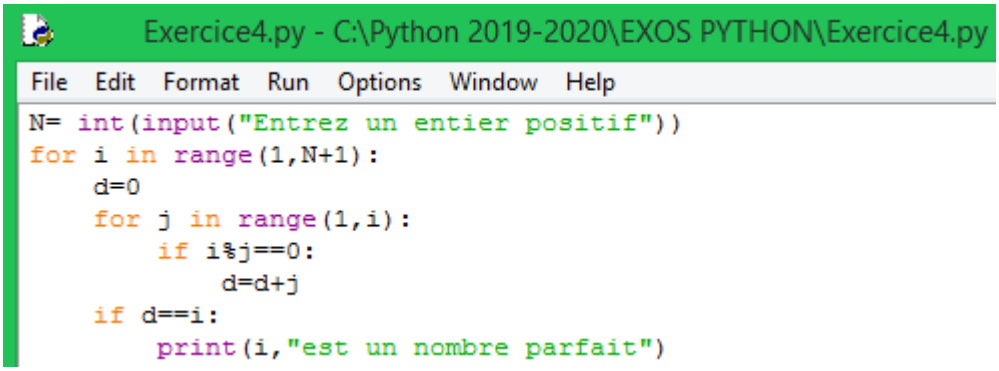
```
N=int(input("Entrez un entier"))
if N>0 and N%2==0:
    S1=0
    print("Voici les entiers positifs pairs <=",N," dans un ordre croissant")
    for i in range(0,N+1,2):
        print(i)
        S1=S1+i
    print("La somme des entiers positifs pairs <=",N,"=",S1)
elif N<0 and N%2!=0:
    S2=0
    print("Voici les entiers négatifs impairs >=",N," dans un ordre décroissant")
    for i in range(-1,N-1,-2):
        print(i)
        S2=S2+i
    print("La somme des entiers négatifs impairs >=",N,"=",S2)
```

Ci-dessous deux exemples d'exécution:

```
========== RESTART: C:\Python 2019-2020\EXOS PYTHON\Exercice3.py ==========
Entrez un entier10
Voici les entiers positifs pairs <= 10  dans un ordre croissant
0
2
4
6
8
10
La somme des entiers positifs pairs <= 10 = 30
>>>
========== RESTART: C:\Python 2019-2020\EXOS PYTHON\Exercice3.py ==========
Entrez un entier-9
Voici les entiers négatifs impairs >= -9  dans un ordre décroissant
-1
-3
-5
-7
-9
La somme des entiers négatifs impairs >= -9 = -25
>>>
```

Exercice 4:

Ecrire un algorithme permettant d'afficher les nombres parfaits compris entre 1 et N.
Un nombre parfait est un entier naturel non nul qui est égal à la somme de ses diviseurs stricts (le nombre lui-même ne fait pas partie des diviseurs).
Exemple : 28 est un nombre parfait puisque 1, 2, 4, 7 et 14 sont les diviseurs stricts de 28 et 14 + 7 + 4 + 2 + 1 = 28.
Exemple:
Si l'utilisateur saisit 1000 pour N, le programme devra afficher les nombres parfaits compris entre 1 et 1000 à savoir: 6, 28 et 496.



Remarques

- ✓ **Lorsque deux boucles sont imbriquées on fixe le compteur de la première boucle (i) et on déroule le compteur de la seconde boucle (j) de bout en bout (première valeur jusqu'à la dernière). Ensuite on incrémente le i on déroule le j et ainsi de suite jusqu'à ce que le i arrive à la valeur d'arrêt.**
- ✓ **Donc pour chaque i inclus dans [1,N]. on va tester s'il est divisible par un j , j étant un entier strictement inférieur à i; on va aller de 1 jusqu'à i-1 (le nombre lui-même ne doit pas figurer parmi les diviseurs).**
- ✓ **Par exemple pour i=6, j=1 6%1==0 et donc d=1**
  - **j=2 6%2==0 et don d=3**
  - **j=3 6%3==0 et donc d=d+j d=6**
  - **j=4 6%4!=0 d= 6**
  - **j=5 6%5!=0 d=6**
- ✓ **à la sorti de la boucle j on compare i et d (la somme de ses diviseurs)**
- ✓ **6==6 alors 6 est un nombre parfait**
- ✓ **le i reçoit i+1 et donc i=7 et ….**
- ✓ **On fait la somme des diviseurs de i on obtient le d puis on compare le d à i avant de passer au i suivant.**