

Exercices de 1ere NSI sur les boucles for et while

1.1 Exercice 1

Selon l'itérable proposé, définir la séquence de valeurs que prend le variant.

Exemple :

instruction	séquence de valeurs
<code>for i in range(4):</code>	<code>{0,1,2,3}</code>
<code>for i in range(1,4):</code>	<code>{1,2,3}</code>

1. `for i in range(10):`
2. `for i in "abacab":`
3. `for i in [6,4,2]:`
4. `for i in range(4,10):`

1.2 Exercice 2

Dans chaque cas, on utilise le système d'instructions suivant :

```
1 for i in .... :  
2     print(i)
```

Compléter l'instruction avec l'itérable pour obtenir les affichages proposés :

1. cas n°1

```
1 0  
2 1  
3 2
```

2. cas n°2

```
1 T  
2 G  
3 V
```

3. cas n°3

```
1 2001  
2 2011
```

1.3 Exercice 3

Quels sont les affichages générés pour chacun des scripts ?

1. script 1

```
1 for i in range(4):  
2     if i > 2:  
3         print(i)
```

2. script 2

```
1 for j in range(4):  
2     if j % 2 == 0:  
3         print(i)
```

3. script 3

```
1 for k in [2012, 2019, 2024]:  
2     if k % 3 == 0:  
3         print(i)
```

1.4 Exercice 4

L'instruction `while i < 4` : a pour condition d'exécution `i < 4` et pour condition d'arrêt `i >= 4`
Définir la condition d'arrêt pour chacune des instructions `while` :

1. `while i > 10`:
2. `while i <= 3`:
3. `while i > j and i != 0`

1.5 Exercice 5

Reprendre les scripts de l'exercice 3. Transformer le script utilisant une boucle `for` en un nouveau script utilisant une boucle `while`

Exemple : Les scripts suivants sont équivalents :

```
1 # script avec for  
2 for i in range(10):  
3     print(i*2)  
4  
5 # script avec while  
6 i = 0
```

```
7 while i < 10:  
8     print(i*2)  
9     i = i + 1
```

1.6 Exercice 6 : Variant de boucle

Dans chaque cas, définir le variant de boucle et préciser si la boucle termine ou non.

- Script 1

```
1 i = 0  
2 a = 10  
3 while i < 3:  
4     i = i + 1  
5     a = 10 * i
```

- Script 2

```
1 a = 4  
2 b = 10  
3 while a > 3:  
4     a = a - 1  
5     b = b + 1
```

- Script 3

```
1 i = 0  
2 a = 10  
3 while a > 3:  
4     i = i + 1  
5     a = a + 10
```

1.7 Exercice 7 : Traitement automatisé

Compléter les scripts proposés afin de ...

1. réaliser la somme des valeurs d'une liste. Puis l'afficher.

```
1 L = [65400, 31654, 1026]  
2 s = 0  
3 for valeur in L:  
4     s += ...  
5 print(..
```

2. créer une chaîne de caractères ascii à partir de leur rang (les valeurs de la liste)

```

1 L = [110, 99, 111]
2 chaine_caracteres = ""
3 for rang in L:
4     caractere_ascii = chr(...)
5     chaine_caracteres += ...
6 print(chaine_caracteres)

```

On rappelle que la conversion *int* → *ascii* se fait avec la fonction *chr*, alors que la conversion *ascii* → *int* se fait avec la fonction *ord*.

3. obtenir une liste de rangs ; chaque rang est celui du caractère dans la table ascii (on fait l'inverse du 2.)

```

1 message = "abacab"
2 L = []
3 for c in message:
4     rang = ord(...)
5     L.append(...)
6 print(...)

```

On rappelle que l'ajout d'une valeur dans une liste, à la suite, se fait avec la méthode *append*

1.8 Exercice 8 : Multiplication

La multiplication de *a* par *b* revient à ajouter *a* + *a* + *a* ... un nombre *b* de fois. Il faudra réaliser des additions successives.

On peut utiliser alors une *variable* pour stocker les résultats de ces additions successives, que l'on appellera *produit*. Ainsi qu'un *compteur* du nombre d'addition. Ce sera *b*. Il faudra diminuer *b* d'une unité à chaque *itération*.

1. Compléter le script suivant qui réalise la multiplication de *a* par *b*, en n'utilisant que l'opérateur + et/ou -.

```

1 a = 3
2 b = 8
3 produit = 0
4 while b > ... :
5     produit = produit + a
6     b = b - ...

```

2. Recopier et compléter le tableau de suivi des variables

b avant le test conditionnel	test b > ...	produit
------------------------------	--------------	---------

3. Prouver la terminaison de ce programme à l'aide d'arguments simples, en utilisant le tableau (Q.2)

1.9 Exercice 9 : Division

Ecrire un script qui réalise la division entière de a par b, en n'utilisant que l'opérateur `+`, et/ou `-`.

1.10 Compteur simple avec while

1. Recopier et compléter le programme suivant pour compter le nombre de fois où l'utilisateur saisit successivement le caractère "1".

Le programme s'arrête lorsque l'utilisateur entre "0". Ce sera la variable `i` qui sera utilisée comme compteur.

```

1 c = "100"
2 i = ...
3 while c != "...":
4     c = input("entrer un nombre 0/1 :")
5     ...
6 print(...)
```

2. Pourquoi écrit-on `c="100"` à la première ligne ?
3. On suppose que l'utilisateur entre successivement les valeurs "1", "1", "1", "0". Compléter le tableau de suivi des variables :

c avant le test conditionnel	test c != 0	saisie de c	i
------------------------------	-------------	-------------	---