TP n°2 : Boucles

Instruction générale : hormis pour les exercices corrigés collectivement, vous ferez valider votre travail par l'enseignant.

1 Modification raccourcie

Dans la programmation et notamment dans les boucles, il arrive fréquemment que l'on souhaite modifier une valeur en fonction d'elle, en lui ajoutant 1 par exemple ; ce qui donnerait `valeur=valeur+1` en Python. Cette notation est un peu lourde et il existe des notations raccourcies en Python pour ce genre d'opérations que nous nous efforcerons maintenant d'utiliser.

Opération	Algorithme	Python	Python raccourci
Addition	$a \leftarrow a + k$	`a=a+k`	`a+=k`
Soustraction	$a \leftarrow a - k$	`a=a-k`	`a-=k`
Multiplication	$a \leftarrow a \times k$	`a=a*k`	`a*=k`
Division	$a \leftarrow a/k$	`a=a/k`	`a/=k`
Puissance	$a \leftarrow a^k$	`a=a**k`	`a**=k`

Il existe d'autres notations raccourcies, notamment pour la division euclidienne, donc pour les entiers : `a//=k` où `a` devient le quotient de sa division euclidienne par `k` et `a%=k` pour le reste.

2 La boucle `for`

La boucle inconditionnelle ou boucle Pour correspond en Python à l'instruction `for`. Pour une variable appartenant à un certain ensemble, on exécute une séquence d'instruction marquée par une indentation. Pour commencer, nous considérerons les ensembles créés à l'aide de l'instruction `range()`.

`range()` peut admettre un, deux ou trois paramètres.

1. Si elle n'a qu'un paramètre n, elle crée une liste de nombre allant de 0 à $n-1$; par exemple `range(10)` crée la liste des nombres allant de 0 à 9.
2. Si elle a deux paramètres, le premier donne le début de la liste et le second la fin moins un (attention !) ; par exemple `range(2,8)` donne la liste des nombres de 2 à 7 (et non à 8 !).
3. Si elle a trois paramètres, les deux premiers ne changent pas et le troisième donne l'incrément, i.e. le nombre que l'on additionne à chaque terme de la liste pour obtenir le suivant ; par exemple, `range(1,21,3)` crée la liste 1, 4, 7, 10, 13, 16, 19.

Exemple : Le programme suivant permet d'afficher tous les entiers de 0 à 99.

```
for nbr in range(100):
    print(nbr)
```

Exercice 1. [Punition] Vous venez de tenter de vendre des bâtons de la mort à un mystérieux inconnu encapuchonné dans un bar du coin. Celui-ci passe la main devant vous en vous disant « Tu ne vends pas de bâtons de la mort. Tu vas rentrer chez toi et copier mille fois « je ne vends pas de bâtons de la mort » ». Contraint par une force mystérieuse, vous rentrez chez vous accomplir la punition. Cependant, mille, ça fait beaucoup, ça serait plus pratique qu'un programme le fasse à votre place. Écrivez ce programme !

Exercice 2. Écrire un programme affichant :

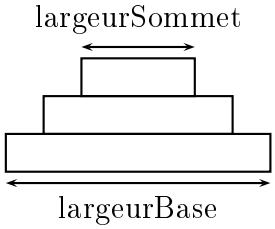
1. tous les entiers pairs strictement inférieurs à 100 ;
2. tous les entiers impairs strictement inférieurs à 100.

Exercice 3. Les programmes de l'exercice précédent ont écrit chacun des nombres sur une nouvelle ligne formant ainsi une colonne. Chercher comment les écrire sur une seule ligne à l'aide de la commande `print()`.

Exercice 4. Écrire un programme prenant en entrée un entier naturel n puis renvoyant :

1. la somme des n premiers entiers : $1 + 2 + \cdots + n$;
2. le produit des n premiers entiers : $1 \times 2 \times \cdots \times n$.

Exercice 5. [Conquête spatiale] Les Goa'ulds sont une espèce colonisant la galaxie grâce des vaisseaux spatiaux. Ces vaisseaux ont un inconvénient, ils ne peuvent se poser que sur des socles en forme de pyramides comme dans l'exemple ci-dessous. Les Goa'ulds étant des parasites volant les technologies des autres, ils ne savent pas faire grand chose par eux-mêmes. Vous êtes un employé de la multinationale du BTP ayant remporté l'appel d'offre lancé par les Goa'ulds pour leur construire des pyramides afin qu'ils puissent enfin poser leurs vaisseaux, débarquer leurs troupes et conquérir la planète. Vous devez écrire un programme calculant l'exact quantité de béton nécessaire pour construire la pyramide. La pyramide est à base carrée, chaque étage fait un mètre de haut et un mètre de moins en largeur que celui sur lequel il repose. Votre programme doit prendre deux entiers en entrée : largeurBase et largeurSommet qui indique respectivement la largeur de la base et du sommet de la pyramide (dans quel cas la pyramide est-elle pointue ?) puis renvoyer le volume total de béton nécessaire.



Exercice 6. Écrire un programme permettant d'afficher les entiers de 1 à 9 sous forme de triangle comme dans les exemples ci-dessous.

1	1	9	9
22	12	88	98
333	123	777	987
4444	1234	6666	9876
.....	.....	.....	.....

Exercice 7. [Randonnée] Un groupe de randonneurs souhaiterait avoir une application dédiée lors de ses sorties. Ils souhaiteraient notamment que l'application calcule les dénivelés totaux positifs et négatifs lorsqu'ils créent leurs parcours. Les parcours sont constituées de plusieurs étapes qui sont soit de montées, soit de descentes, soit de plats. L'application doit d'abord prendre en entrée le nombre d'étapes puis pour chaque étape le dénivelé de celle-ci (positif, négatif ou nul) et enfin donner en sortie les dénivelés totaux positifs et négatifs. Par exemple, pour un parcours en 4 étapes, l'utilisateur doit d'abord rentrer 4 puis par exemple les dénivelés suivants : 120, -30, 50, 0 ; dans ce cas, elle devra renvoyer « dénivelé positif : 170 » et « dénivelé négatif : 30 ». Coder en Python un programme réalisant cette fonctionnalité.

3 La boucle `while`

La boucle conditionnelle ou boucle Tant que correspond en Python à l'instruction `while`. Tant qu'une certaine condition est vérifiée, on exécute une séquence d'instruction marquée par une indentation. La plupart des boucles `for` peuvent en réalité remplacer par des boucles `while`, lesquelles sont en général plus rapides. Toutefois, il faut rester vigilant à la condition de définition de la boucle ; celle-ci doit être vraie avant la boucle pour que celle-ci puisse débuter et devenir fausse lors de son exécution afin que la boucle s'arrête et ne soit pas infinie.

Exemple : Le programme suivant permet aussi (comme dans la section précédente) d'afficher tous les entiers de 0 à 99.

```
nbr=0
while nbr<100:
    print(nbr)
    nbr+=1
```

Exercice 8. Écrire un programme utilisant des boucles `while` affichant :

1. tous les entiers pairs strictement inférieurs à 100 ;
2. tous les entiers impairs strictement inférieurs à 100.

Exercice 9. Écrire un programme utilisant des boucles `while` prenant en entrée un entier naturel n puis renvoyant :

1. la somme des n premiers entiers : $1 + 2 + \cdots + n$;
2. le produit des n premiers entiers : $1 \times 2 \times \cdots \times n$.

Exercice 10. [« It's over 9000 ! »]

1. Écrire un programme trouvant le plus grand multiple de 42 qui soit inférieur à 9000.
2. Écrire un programme trouvant le plus petit multiple de 117 qui soit plus grand que 9000.

Exercice 11. [« It's over 1 000 000 ! »]

1. Écrire un programme déterminant la dernière puissance p de 3 telle que la somme de celle-ci et des précédentes soit supérieure à 1 000 000. Par exemple, p ne vaut pas 3 puisque $3^0 + 3^1 + 3^2 + 3^3 = 1 + 3 + 9 + 27 = 40 < 1000000$.
2. Modifier le programme précédent pour que 3 et 1 000 000 soient des paramètres que puisse rentrer l'utilisateur.

Exercice 12. Déterminer si chacun des programmes suivants est correct et sinon le corriger.

1.
```
while i>0:
      print(nbr)
      i-=1
```

2.
```
j=1
while j<42:
     j*=2
print(j)
```

3.
```
k=128
while k>0:
     k/=2
print(k)
```

Exercice 13. [Encore des pyramides] Les Goa'ulds sont de retours et voudraient construire de nouvelles pyramides, cette fois-ci décoratives et en pierres (ça fait plus authentique). À l'image de leur puissance, ils les veulent les plus grandes possibles. Elles sont toujours à base carrée, chaque étage faisant un mètre de moins en largeur que celui sur lequel il repose mais cette fois elles doivent être « pointues » : il y a une unique pierre au sommet. Chaque pierre est un cube de 1m d'arête. Votre multinationale du BTP souhaite répondre à l'appel d'offre des Goa'ulds mais vous devez pour cela savoir quelle est la taille maximale des pyramides que vous pouvez construire. Pour cela vous contactez vos fournisseurs de pierres et leur demandez quel est le nombre maximal de pierres dont vous pouvez disposer. Écrivez un programme prenant en entrée ce nombre maximal de pierres et vous renvoyant le nombre d'étages maximal de la pyramide que vous pourrez construire avec ainsi que le nombre exact de pierres nécessaires (votre $n + 1$ va vous licencier si vous achetez des pierres inutiles).

Exercice 14. [Analyse médicale] On souhaite programmer un logiciel de surveillance médicale analysant les données produites par divers capteurs mesurant les caractéristiques d'un patient (température, pouls, taux oxygène dans le sang, etc) lors d'une période observation. Pour une caractéristique choisie, l'utilisateur doit donner quatre entrées au logiciel : 1. la borne d'alerte inférieure la caractéristique mesurée ; 2. la borne d'alerte supérieure ; 3. la durée de l'observation ; 4. la fréquence des mesures. Il devra ensuite calculer le nombre de mesures à effectuer et pour chacune de ces mesures (qu'il prendra en entrée), afficher « Normal » tant que les mesures sont dans l'intervalle donné et afficher « Alerte ! » dès que les mesures sortent de l'intervalle. Écrire un programme répondant aux besoins de ce logiciel et proposer des jeux de données permettant de le tester.

Exercice 15. [Mission Mölkky] L'entreprise MölkkyCorp, fabriquant de jeux de Mölkky, souhaite fournir une application permettant de calculer le score des joueurs à ses clients.

Le principe du jeu est de faire tomber des quilles en bois à l'aide d'un lanceur appelé Mölkky. Les quilles sont marquées de 1 à 12. Le premier joueur arrivant à totaliser exactement 50 points gagne la partie. Il y a deux façons de marquer des points :

— si un joueur fait tomber plusieurs quilles, il gagne autant de points que de quilles abattues ;
— si un joueur fait tomber une seule quille, il gagne autant de points que le nombre inscrit dessus.

Par ailleurs, il y a deux règles à respecter :

— si un joueur dépasse les 50 points, son score retombe à 25 ;
— si un joueur ne touche pas de quilles sur trois lancers consécutifs, il est éliminé.

Votre mission, si vous l'acceptez, est d'écrire puis coder en Python un programme répondant aux besoins de la MölkkyCorp ainsi que de fournir un jeu de données permettant de tester votre programme. Cet énoncé s'autodétruira dans cinq secondes.

4 Ressources supplémentaires

— Les cours sur Python d'OpenClassRooms
— Le cours du W3 Schools
— Vidéos sur les boucles