Boucles

Exercices

Exercice 0 *QCM – vérification des prérequis* Pour chaque question, une seule réponse est correcte.

1. Quelles valeurs prend `i` dans la boucle `for i in range(3, 7):` ?

— A. 3, 4, 5, 6, 7

— B. 3, 4, 5, 6

— C. 4, 5, 6

— D. 3, 4, 5

2. Combien de fois la boucle suivante s'exécute-t-elle ?

```
for i in range(5):
    print(i)
```

— A. 4 fois

— B. 5 fois

— C. 6 fois

— D. Cela dépend de la valeur de `i`

3. Quelle est la différence entre `for` et `while` ?

— A. `for` est plus rapide que `while`

— B. `for` s'utilise quand on connaît le nombre d'itérations, `while` quand on ne le connaît pas à l'avance

— C. `while` ne peut pas faire de boucles infinies

— D. `for` et `while` sont strictement équivalents et interchangeables dans tous les cas

4. Que vaut `s` après l'exécution du code suivant ?

```
s = 0
for i in range(1, 4):
    s = s + i
```

— A. 3

— B. 6

— C. 10

— D. 4

Exercice 1 *Boucles consécutives et imbriquées* Sans exécuter les programmes, dire combien de fois chacun affiche le symbole #. Expliquer, puis vérifier.

Programme 1

```python
for i in range(4):
    print("#", end="")
for j in range(6):
    print("#", end="")
print()
```

Programme 2

```python
for i in range(4):
    for j in range(6):
        print("#", end="")
print()
```

Programme 3

```python
for i in range(4):
    for j in range(i):
        print("#", end="")
print()
```

Exercice 2 *Lancer un dé jusqu'au 6* Le programme suivant lance un dé à six faces une seule fois :

```python
import random

de = random.randint(1, 6)
print(f"Le dé affiche {de}")
```

Le modifier pour qu'il relance le dé **tant que** le 6 n'est pas sorti, en affichant chaque lancer, puis qu'il affiche le nombre de lancers qu'il a fallu.

Exercice 3 *Sommes de deux dés* On lance deux dés et on additionne leurs valeurs. Écrire un programme qui affiche le tableau de toutes les sommes possibles : une ligne pour chaque valeur du premier dé, une colonne pour chaque valeur du second.

```
2 3 4 5 6 7
3 4 5 6 7 8
4 5 6 7 8 9
5 6 7 8 9 10
6 7 8 9 10 11
7 8 9 10 11 12
```

Exercice 4 *Puissances de 2* Écrire un programme qui affiche les puissances de 2, de 2^0 à 2^{16}, sous la forme `2 puissance 10 = 1024`. Quelle puissance de 2 correspond au nombre de valeurs différentes d'un octet ?

Exercice 5 *Croissance de bactéries* Une culture contient 100 bactéries, et leur nombre double toutes les 20 minutes. Écrire un programme qui affiche, toutes les 20 minutes, le temps écoulé et le nombre de bactéries, tant que ce nombre ne dépasse pas un million. Il affiche ensuite au bout de combien de minutes le million est dépassé.

Exercice 6 *Rebonds d'une balle* Une balle lâchée d'une hauteur de 2 mètres remonte, à chaque rebond, à 80 % de la hauteur précédente. Écrire un programme qui affiche la hauteur atteinte après chacun des 10 premiers rebonds, arrondie au centimètre.

Exercice 7 *Années bissextiles* On rappelle qu'une année est bissextile si elle est divisible par 4 sans être divisible par 100, ou si elle est divisible par 400. Écrire un programme qui affiche toutes les années bissextiles de 2000 à 2100, puis leur nombre.

Exercice 8 *Triangle de chiffres* Écrire un programme qui affiche le motif suivant :

```
1
12
123
1234
12345
```

Exercice 9 *Moyenne et meilleure note* Écrire un programme qui demande combien de notes un élève a obtenues, puis demande ces notes une par une, et affiche leur moyenne et la meilleure note.

Exercice 10 *Tabulation et piège des flottants* On s'intéresse à la fonction $f(x) = x^2 - 2$, qui s'annule pour $x = \sqrt{2}$.

1. Écrire un programme qui affiche $f(x)$ pour les entiers x de -3 à 3.
2. Écrire un programme qui affiche $f(x)$ pour x allant de -10 à 10 avec un pas de 2.
3. Écrire un programme qui affiche $f(x)$ pour x allant de 0 à 2 avec un pas de 0,1. Entre quelles valeurs de x le signe de $f(x)$ change-t-il ? Qu'en déduit-on pour $\sqrt{2}$?
4. Écrire la même chose avec une boucle `while` : `x` part de 0 et augmente de 0,1 à chaque tour, tant que `x <= 2`. Comparer les valeurs de `x` affichées et le nombre de lignes avec la question 3. Que remarque-t-on ?

Exercice 11 *Mot à l'envers* Écrire un programme qui demande un mot et construit, lettre par lettre, la chaîne `envers` qui contient ce mot écrit à l'envers, puis l'affiche. Par exemple, `python` donne `nohtyp`. Tester avec `radar` : que remarque-t-on ?

Exercice 12 *Compte à rebours* Écrire un programme qui demande un nombre entier positif n à l'utilisateur, puis affiche un compte à rebours de n à 0. **Exemple :** si l'utilisateur entre 5, afficher : 5, 4, 3, 2, 1, 0.

Exercice 13 *Somme des entiers* Écrire un programme qui calcule la somme des entiers de 1 à n (avec n saisi par l'utilisateur) en utilisant une boucle `while`. **Exemple :** pour $n = 5$, la somme est $1+2+3+4+5 = 15$.

Exercice 14 *Devine le nombre* Créer un jeu où l'ordinateur choisit un nombre aléatoire entre 1 et 100, et l'utilisateur doit le deviner. À chaque tentative, indiquer si le nombre cherché est plus grand ou plus petit. **Indice :** utiliser `import random` et `random.randint(1, 100)`.

Exercice 15 *Suite de Fibonacci* Afficher les n premiers termes de la suite de Fibonacci avec une boucle `while`.

Rappel sur la suite de Fibonacci : c'est une suite de nombres célèbre qui commence par 0 et 1. Ensuite, chaque nombre est obtenu en additionnant les deux nombres précédents.

— Le premier nombre est 0

— Le deuxième nombre est 1

— Le troisième nombre est $0 + 1 = 1$

— Le quatrième nombre est $1 + 1 = 2$

— Le cinquième nombre est $1 + 2 = 3$

— Le sixième nombre est $2 + 3 = 5$

— Le septième nombre est $3 + 5 = 8$

— et ainsi de suite...

La suite donne donc : 0, 1, 1, 2, 3, 5, 8, 13, 21, 34...

Pour calculer cette suite, vous aurez besoin de trois variables : une pour le nombre actuel, une pour le nombre précédent, et une pour compter combien de nombres ont été affichés.

Exercice 16 ***Validation de mot de passe*** Écrire un programme qui demande un mot de passe à l'utilisateur. Le mot de passe correct est `"NSI2026"`. Le programme doit redemander le mot de passe tant qu'il n'est pas correct, avec un maximum de 3 tentatives.

Exercice 17 ***Jeu du plus ou moins*** Créer un jeu où deux joueurs s'affrontent :

— Le joueur 1 choisit un nombre secret entre 1 et 50

— Le joueur 2 doit le deviner en 7 tentatives maximum

— Après chaque tentative, indiquer si le nombre est plus grand ou plus petit

— Compter le nombre de tentatives et afficher un message de victoire ou défaite

Exercice 18 ***Des bornes exactes (pour aller plus loin)*** Pour chaque tâche, écrire la boucle `for` avec `range`, puis **justifier** les bornes : donner le nombre de tours et la dernière valeur prise par la variable de boucle.

1. Afficher les multiples de 7, de 7 à 70 compris.
2. Afficher un compte à rebours de 10 à 1.
3. Afficher les nombres de 100 à 0 de 5 en 5 : 100, 95, 90, et ainsi de suite jusqu'à 0 compris.
4. Pour un entier `n` saisi au clavier, afficher les carrés des entiers de 1 à `n` compris.

Exercice 19 ***Estimer une probabilité par simulation (pour aller plus loin)*** On lance deux dés à six faces et on additionne leurs valeurs. On cherche la probabilité que la somme vaille 7.

1. Écrire un programme qui simule `n = 10000` lancers des deux dés avec `random.randint`, compte ceux dont la somme vaut 7 et affiche la fréquence obtenue, arrondie à trois chiffres après la virgule.
2. En s'aidant du tableau des sommes de deux dés, calculer la probabilité exacte d'obtenir 7, puis la comparer à la fréquence.
3. Lancer plusieurs fois le programme avec `n = 100`, puis avec `n = 100000`. Que constate-t-on ?