Devoir surveillé n°1

Première spécialité NSI

Bases de la programmation en Python : variables, types, conditions, boucles

Durée : 1 h

Calculatrice et ordinateur interdits

Nom : _____ Prénom : _____ Classe : _____

Consignes

- Les exercices 1 et 2 se font **directement sur le sujet**, les exercices 3 et 4 sur copie.
- Le code écrit doit être correctement **indenté** : l'indentation fait partie du programme.
- Les exercices sont indépendants et peuvent être traités dans l'ordre de votre choix.
- Toute trace de recherche pertinente sera valorisée.

Barème

- Exercice 1 : .../ 5
- Exercice 2 : .../ 5
- Exercice 3 : .../ 4
- Exercice 4 : .../ 6

Remarque

Rappel : pour deux entiers `a` et `b`, `a // b` donne le quotient entier et `a % b` le reste de la division euclidienne de `a` par `b`.

Exercice 1 – QCM (5 points)

Pour chaque question, une seule réponse est correcte. Reporter la lettre choisie dans le tableau en fin d'exercice. Une bonne réponse rapporte 0,5 point, une réponse fausse ou une absence de réponse n'enlève aucun point.

1. Quel est le type de la valeur de l'expression `7 / 2` ?

a) `int` **b)** `float` **c)** `str` **d)** `bool`

2. On exécute `x = input("Nombre ? ")` et l'utilisateur tape `5`. Qu'affiche ensuite `print(x * 2)` ?

a) `10` **b)** `10.0` **c)** `55` **d)** une erreur

3. Quelles valeurs prend la variable `i` dans la boucle `for i in range(2, 8, 2):` ?

a) 2, 4, 6 **b)** 2, 4, 6, 8 **c)** 2, 3, 4, 5, 6, 7 **d)** 0, 2, 4, 6

4. Qu'affiche le programme suivant ?

```
1  a = 3
2  b = a
3  a = a + 1
4  print(b)
```

a) `4` **b)** `a` **c)** `b` **d)** `3`

5. Qu'affiche l'instruction `print(17 // 5, 17 % 5)` ?

a) `3.4 2` **b)** `3 2` **c)** `2 3` **d)** `3 3.4`

6. Quelle est la valeur de `i` après l'exécution de ce programme ?

```
1 i = 10
2 while i > 0:
3     i = i - 3
```

a) `0` **b)** `1` **c)** `-2` **d)** la boucle est infinie

7. Quelle condition est vraie exactement lorsque `n` est compris entre 10 et 20, bornes incluses ?

a) `10 <= n and n <= 20` **b)** `10 <= n or n <= 20`

c) `n >= 10 and n < 20` **d)** `10 < n and n < 20`

8. Que vaut l'expression `"3" + "4"` ?

a) `7` **b)** `"7"` **c)** `"34"` **d)** une erreur

9. Lequel de ces programmes ne s'arrête jamais ?

a)
```
i = 0
while i < 5:
    i = i + 1
```

b)
```
i = 0
while i < 5:
    print(i)
```

c)
```
i = 5
while i > 0:
    i = i - 1
```

d)
```
for i in range(1000):
    print(i)
```

10. Que vaut l'expression `"a" in "Python"` ?

a) `True` **b)** `1` **c)** `"a"` **d)** `False`

Question	1	2	3	4	5	6	7	8	9	10
Réponse										

Exercice 2 – Suivre l'évolution des variables (5 points)

Partie A

On considère le programme suivant :

```
1 s = 0
2 for k in range(1, 6):
3     if k % 2 == 1:
4         s = s + k
5 print(s)
```

1. Compléter le tableau ci-dessous, qui donne, à chaque tour de boucle, la valeur de `k`, la valeur de la condition `k % 2 == 1` et la valeur de `s` **à la fin** du tour.

Tour	k	k % 2 == 1	s
1			
2			
3			
4			
5			

2. Qu'affiche ce programme ?

3. Décrire en une phrase ce que calcule ce programme.

Partie B

On considère le programme suivant :

```
1  n = 50
2  compteur = 0
3  while n > 1:
4      n = n // 2
5      compteur = compteur + 1
6  print(n, compteur)
```

4. Compléter le tableau ci-dessous avec les valeurs de `n` et de `compteur` à la fin de chaque tour de boucle (toutes les lignes ne seront pas forcément utilisées).

	n	compteur
Avant la boucle	50	0
Tour 1		
Tour 2		
Tour 3		
Tour 4		
Tour 5		
Tour 6		
Tour 7		

5. Qu'affiche ce programme ?

6. Pourquoi est-on certain que cette boucle `while` finit par s'arrêter, quelle que soit la valeur entière positive de départ de `n` ?

Exercice 3 – Trouver et corriger les erreurs (4 points)

Le programme suivant devrait demander 5 notes à l'utilisateur, calculer leur moyenne, puis afficher « Admis » si la moyenne est supérieure ou égale à 10, et « Non admis » sinon. Il contient **quatre erreurs**.

```
1  for i in range(5)
```

```
2       somme = 0
3       note = input("Note : ")
4       somme = somme + note
5   moyenne = somme / 5
6   if moyenne >= 10:
7       print("Admis")
8   else:
9   print("Non admis")
```

1. Pour chacune des quatre erreurs, donner le numéro de la ligne concernée, expliquer le problème en une phrase, et proposer une correction.
2. Parmi ces quatre erreurs, l'une n'empêche pas le programme de s'exécuter jusqu'au bout (une fois les trois autres corrigées), mais produit un résultat faux. Laquelle ? Quel résultat obtiendrait-on alors avec les notes 12, 8, 15, 10, 14 ?

Exercice 4 – Écrire des programmes (6 points)

Les trois questions sont indépendantes. Pour chacune, écrire le programme Python demandé.

1. *(2 points)* Écrire un programme qui demande à l'utilisateur un entier n, puis calcule et affiche la somme $1 + 2 + \cdots + n$.
   Exemple : si l'utilisateur tape `4`, le programme affiche `10`.
2. *(2 points)* Écrire un programme qui demande un mot (en minuscules), puis affiche ce mot privé de toutes ses voyelles (`a`, `e`, `i`, `o`, `u`, `y`).
   Exemple : si l'utilisateur tape `informatique`, le programme affiche `nfrmtq`.
3. *(2 points)* Le code secret d'une carte bancaire est `"1234"`. Écrire un programme qui demande le code à l'utilisateur, en lui laissant **au maximum 3 essais**. Dès que le code saisi est correct, le programme affiche « Accès autorisé » et s'arrête. Si les 3 essais sont faux, il affiche « Carte bloquée ».
   Exemple d'exécution :

```
Code : 1111
Code : 1234
Accès autorisé
```

Correction de l'exercice 1

Question	1	2	3	4	5	6	7	8	9	10
Réponse	b	c	a	d	b	c	a	c	b	d

1. L'opérateur `/` renvoie toujours un `float` (`7 / 2` vaut `3.5`).
2. `input` renvoie une chaîne : `"5" * 2` répète la chaîne, d'où `55`.
3. `range(2, 8, 2)` part de 2, avance de 2 en 2 et s'arrête strictement avant 8.
4. `b` reçoit la valeur de `a` au moment de l'affectation (3) ; modifier `a` ensuite ne change pas `b`.
5. $17 = 5 \times 3 + 2$: quotient 3, reste 2.
6. `i` prend les valeurs 10, 7, 4, 1, puis -2 ; la condition `i > 0` devient fausse et la boucle s'arrête.
7. Il faut les deux conditions à la fois (`and`), avec des inégalités larges pour inclure les bornes.

8. `+` entre deux chaînes les concatène.
9. En b), `i` n'est jamais modifiée dans la boucle : la condition `i < 5` reste toujours vraie.
10. La lettre `a` n'apparaît pas dans `"Python"`.

Correction de l'exercice 2

Partie A

1.

Tour	k	k % 2 == 1	s
1	1	True	1
2	2	False	1
3	3	True	4
4	4	False	4
5	5	True	9

2. Le programme affiche `9`.
3. Il calcule la somme des entiers impairs de 1 à 5 ($1+3+5=9$).

Partie B

4.

	n	compteur
Avant la boucle	50	0
Tour 1	25	1
Tour 2	12	2
Tour 3	6	3
Tour 4	3	4
Tour 5	1	5

Après le tour 5, `n` vaut 1 : la condition `n > 1` est fausse, la boucle s'arrête. Les lignes 6 et 7 restent vides.

5. Le programme affiche `1 5`.
6. À chaque tour, `n` est divisé par 2 (quotient entier) : c'est un entier qui **diminue strictement** tant qu'il est supérieur à 1. Il finit donc forcément par valoir 1 (ou moins), et la condition `n > 1` devient fausse.

Remarque

Pour aller plus loin avec la classe : `compteur` est le nombre de divisions par 2 nécessaires pour ramener `n` à 1. Comme $50 = (110010)_2$ s'écrit avec 6 bits, on trouve $6-1=5$: chaque division entière par 2 supprime le dernier bit.

Correction de l'exercice 3

1. Les quatre erreurs :
   - **Ligne 1** : il manque les deux-points à la fin de l'en-tête de la boucle. Correction : `for i in range(5):`
   - **Ligne 2** : `somme = 0` est à l'intérieur de la boucle, donc remise à zéro à chaque

tour. Correction : placer cette ligne **avant** la boucle, sans indentation.

- **Ligne 3** : `input` renvoie une chaîne ; `somme + note` additionne un nombre et une chaîne, ce qui provoque une erreur. Correction : `note = float(input("Note : "))` (ou `int`).
- **Ligne 9** : l'instruction du `else` n'est pas indentée. Correction : indenter `print("Non admis")`.

Programme corrigé :

```
1  somme = 0
2  for i in range(5):
3      note = float(input("Note : "))
4      somme = somme + note
5  moyenne = somme / 5
6  if moyenne >= 10:
7      print("Admis")
8  else:
9      print("Non admis")
```

2. C'est l'erreur de la ligne 2 (initialisation dans la boucle) : c'est une erreur de **logique**, que Python ne signale pas. À la fin de la boucle, `somme` ne contient que la dernière note, 14. On obtient donc une moyenne de $14/5 = 2{,}8$ et le programme affiche « Non admis », alors que la vraie moyenne vaut $59/5 = 11{,}8$.

Correction de l'exercice 4

1.

```
1  n = int(input("n = "))
2  somme = 0
3  for i in range(1, n + 1):
4      somme = somme + i
5  print(somme)
```

Points attendus : conversion avec `int`, accumulateur initialisé avant la boucle, borne `n + 1` (pour inclure n).

2.

```
1  mot = input("Mot : ")
2  resultat = ""
3  for lettre in mot:
4      if lettre not in "aeiouy":
5          resultat = resultat + lettre
6  print(resultat)
```

Accepter aussi `if not lettre in "aeiouy"`, ou un `if lettre in "aeiouy"` avec la concaténation dans le `else`.

3.

```
1  essais = 0
2  code = ""
3  while essais < 3 and code != "1234":
4      code = input("Code : ")
5      essais = essais + 1
6  if code == "1234":
7      print("Accès autorisé")
8  else:
```

```
9       print("Carte bloquée")
```

Points attendus : boucle `while` dont la condition combine les deux causes d'arrêt avec `and`, compteur d'essais mis à jour à chaque tour, test **après** la boucle pour savoir laquelle des deux causes l'a arrêtée. Le code est comparé à la **chaîne** `"1234"` (pas de conversion nécessaire). Accepter toute solution correcte, par exemple une boucle `for` de 3 tours avec une variable booléenne, pourvu qu'on ne redemande plus le code une fois qu'il est trouvé.