Partie 1 – Les bases : comprendre les instructions

Exercice 1 – Un premier calcul

On exécute le programme suivant :

```
x = float(input("x = "))
y = x + 8
print(y)
```

1. Quelle valeur s'affiche si l'utilisateur saisit `5` ?
2. Quelle valeur s'affiche s'il saisit `-3` ?
3. Que calcule ce programme à partir de la valeur saisie ?

Exercice 2 – Compléter un programme

On souhaite saisir un nombre, le multiplier par 3, puis ajouter 2 et afficher le résultat.

Compléter le programme :

```
x = float(input("x = "))
y = ...............
print(...............)
```

Exercice 3 – Comprendre les affectations

On exécute :

```
x = 4
x = x + 3
x = 2*x
print(x)
```

1. Quelle est la valeur de `x` après la première instruction ?
2. Quelle est sa valeur après la deuxième ?
3. Quelle valeur est affichée à la fin ?

Exercice 4 – Écrire un algorithme simple

Écrire un programme Python qui demande l'âge d'une personne et affiche son âge dans 5 ans.

DEBORD

Partie 2 – Les conditions : choisir entre deux possibilités

Exercice 5 – Positif ou négatif ?

```
x = float(input("x = "))
if x >= 0:
  print("Positif ou nul")
else:
  print("Negatif")
```

Indiquer ce qui s'affiche pour chacune des valeurs suivantes : `7`, `-2` et `0`.

Exercice 6 – Un tarif simple

Un cinéma propose le tarif suivant :

— moins de 12 ans : 6 euros ;

— à partir de 12 ans : 9 euros.

Compléter le programme :

```
age = int(input("Age : "))
if age < ...............:
  prix = ...............
else:
  prix = ...............
print(prix)
```

Exercice 7 – Une réduction

Un magasin accorde une réduction de 5 euros si le montant des achats est supérieur ou égal à 50 euros.

```
prix = float(input("Montant des achats : "))
if prix >= 50:
  prix = prix - 5
print(prix)
```

1. Quel montant s'affiche pour un achat de 40 euros ?
2. Quel montant s'affiche pour un achat de 60 euros ?
3. Quel montant s'affiche pour un achat de 50 euros ?

DEBORD

Exercice 8 – Écrire une condition

Un élève est admis si sa moyenne est supérieure ou égale à 10.

Écrire un programme qui demande sa moyenne et affiche `Admis` ou `Non admis`.

Partie 3 – La boucle `for` : répéter un calcul

Exercice 9 – Comprendre une boucle

```
for i in range(1, 5):
  print(i)
```

1. Quelles valeurs prend la variable `i` ?
2. Combien de fois l'instruction `print(i)` est-elle exécutée ?
3. Qu'affiche le programme ?

Exercice 10 – Une tirelire

Une tirelire contient initialement 20 euros. On y ajoute 5 euros chaque semaine.

```
x = 20
for i in range(1, 4):
  x = x + 5
print(x)
```

1. Combien de fois ajoute-t-on 5 euros ?
2. Quelle somme contient la tirelire à la fin ?
3. Que faudrait-il modifier pour simuler 6 semaines ?

Exercice 11 – Compléter une boucle

On souhaite afficher les nombres de 1 à 10.

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

Compléter l'instruction `range` pour obtenir le résultat souhaité.

DEBORD

Exercice 12 – Calculer une somme

On souhaite calculer la somme des cinq premiers entiers strictement positifs.

```
S = 0
for i in range(1, 6):
  S = S + i
print(S)
```

1. Compléter le tableau suivant.

Valeur de `i`	Valeur de `S` après l'ajout
1	
2	
3	
4	
5	

2. Quelle valeur est affichée ?

Partie 4 – La boucle `while` : répéter jusqu'à une condition

Exercice 13 – Une somme qui augmente

```
x = 10
n = 0
while x < 25:
  x = x + 5
  n = n + 1
print(n)
```

1. Quelle est la valeur initiale de `x` ?
2. Quelle est la valeur de `x` après le premier passage dans la boucle ?
3. Combien de fois la boucle est-elle exécutée ?
4. Quelle valeur est affichée ?

DEBORD

Exercice 14 – Compléter une boucle

Une tirelire contient 10 euros. On ajoute 2 euros chaque semaine. On veut déterminer le nombre de semaines nécessaires pour atteindre ou dépasser 20 euros.

Compléter le programme :

```
x = 10
n = 0
while x ..............        20:
  x = ..............
  n = ..............
print(n)
```

Exercice 15 – Comparer `for` et `while`

Pour chaque situation, indiquer la boucle la plus adaptée.

1. Répéter un calcul exactement 8 fois.
2. Ajouter 10 euros chaque mois jusqu'à atteindre 100 euros.
3. Afficher les nombres de 1 à 20.
4. Doubler un nombre jusqu'à ce qu'il dépasse 1 000.

Partie 5 – Les fonctions Python

Exercice 16 – Utiliser une fonction

```
def carre(x):
return(x**2)
print(carre(3))
```

1. Quel calcul réalise la fonction `carre` ?
2. Quelle valeur affiche le programme ?
3. Que faudrait-il écrire pour afficher le carré de 7 ?

DEBORD

Exercice 17 – Compléter une fonction

On souhaite créer une fonction qui reçoit un nombre `x` et renvoie le nombre obtenu en multipliant `x` par 4 puis en retirant 1.

```
def calcul (x):
y = ...............
return(...............)
```

Compléter les deux lignes manquantes.

Exercice 18 – Une fonction avec deux paramètres

```
def somme(a, b):
return(a+b)
print(somme(4, 6))
```

1. Combien de paramètres possède la fonction ?
2. Quelle valeur renvoie-t-elle pour `somme(4,6)` ?
3. Quelle valeur afficherait `print(somme(10,3)` ?

Partie 6 – Les listes

Exercice 19 – Lire une liste

On considère le programme :

```
L = [4, 7, 3, 9]
print(L[0])
print(L[2])
```

1. Combien d'éléments contient la liste `L` ?
2. Quelles valeurs sont affichées ?
3. Quel est l'indice du premier élément ?
4. Comment afficher le nombre 9 ?

DEBORD

Exercice 20 – Construire et parcourir une liste

On exécute :

```
L = []
for k in range(5):
  L.append(k)
print(L)
```

1. Que signifie `L = []` ?
2. Quelles valeurs prend `k` ?
3. Quelle liste obtient-on à la fin ?

On considère maintenant le programme suivant :

```
→L = [2, 4, 6]
→S = 0
→for x in L:
→S = S + x
→print(S)
```

4. Que calcule ce programme ?
5. Quelle valeur affiche-t-il ?

DEBORD