

Programmation – Algorithmique – L1 – S1

Contrôle Continu 1 – lundi 4 mai 2026

Durée : 30 minutes (40 min Tiers-Temps)

Nom :

Prénom :

Num étudiant :

Exercice 1 (8 points) - QCM

Chaque question est sur 1 point. Il n'y a qu'une seule réponse correcte par question.

- Réponse correcte = 1 point,
- Réponse incorrecte = -0,25 point,
- Aucune réponse ou plusieurs réponses = 0 point.

La note globale, si elle est négative, est ramenée à 0.

question 1 :

Que vaut l'expression $(7 \% 3 != 2) \text{ and } (3 // 2 == 1) \text{ or } 3 + 1 == 10 \% 6$

True ☐

False ☐

question 2 :

Que vaut l'expression $7 / 2 * 3 + 24 // 5 \% 2$?

9 ☐

10.5 ☐

une autre valeur ☐

question 3 :

mot = "abcdefgh". Que vaut mot[-7 : -3] ?

"bcde" ☐

"bcdef" ☐

une autre chaîne ☐

question 4 :

mot = "abcdEFgHIJkL". Que vaut mot[3 : 7] + mot[-6 : -3] + mot[-1] ?

"dEFgHgHIJL" ☐

"dEFggHIL" ☐

une autre chaîne ☐

question 5 :

range(8, 5, -1) génère une suite de nombres entiers. Laquelle ?

8 ; 7 ; 6 ; 5 ☐

8 ; 7 ; 6 ; 5 ; 4 ; 3 ; 2 ; 1 ; 0 ; -1 ☐

une autre suite ☐

question 6 :

mot = "aeiouyeiueoayau". Que vaut mot.count(mot[1 : 3]) ?

0 ☐

1 ☐

2 ☐

3 ☐

question 7 :

phrase = "bonjour tout le monde !". Que vaut len(phrase.split("o")) ?

5 ☐

4 ☐

3 ☐

2 ☐

question 8 :

liste = ["a", "bc", "def"]. Que vaut "k".join(liste.pop()) ?

["a", "bc", "def", "k"] ☐

["a", "bc", "k"] ☐

"abck" ☐

"dkekf" ☐

Exercice 2 (4 points) – Analyse de code

Voici un code :

```
1 def f1(i, j) :  
2     return 2 * i + 3 * j  
3  
4 def f2(i, j) :  
5     return 2 * j + 3 * i  
6  
7 for i in range(2, 6) :  
8     for j in range(3, 8, 2) :  
9         if f2(i, j) > f1(i, j) or f1(i, j) > 26 :  
10             print("A")  
11         else :  
12             print("A", end = "-")
```

1. Que vaut $f1(5, 7)$? Que vaut $f2(5, 7)$?
2. Quel est l'affichage produit par le programme ?

Exercice 3 (8 points) – Ecriture de code

On souhaite écrire un programme permettant de gérer le stock d'une petite boutique.

Les quantités disponibles pour différents produits sont stockées dans une liste. La position d'un produit dans la liste correspond à son numéro.

Par exemple : `stock = [12, 5, 0, 8, 3]`

On considère ici cinq produits :

- produit 0 : 12 exemplaires
- produit 1 : 5 exemplaires
- produit 2 : 0 exemplaire
- produit 3 : 8 exemplaires
- produit 4 : 3 exemplaires

Question 1 — Rechercher un produit

Écrire une fonction **def disponible(stock, numProduit)** qui reçoit en arguments une liste **stock** et un numéro de produit **numProduit**. La fonction doit retourner True si le produit possède au moins un exemplaire en stock, et False sinon.

Exemples :

`disponible([12, 5, 0, 8], 1)` retourne True.

`Disponible([12, 5, 0, 8], 2)` retourne False.

Question 2 — Compter les produits en rupture

Écrire une fonction **def nombre_ruptures(stock)** qui retourne le nombre de produits dont la quantité en stock est égale à 0.

Exemple :

`nombre_ruptures([12, 0, 5, 0, 3])` retourne 2

Question 3 — Réapprovisionnement

Écrire une fonction **def reapprovisionner(stock, seuil)** qui reçoit en arguments une liste **stock** et un entier **seuil**.

La fonction doit retourner une **nouvelle liste** contenant les numéros des produits dont la quantité est strictement inférieure au seuil.

Exemple :

reapprovisionner([12, 5, 0, 8, 3], 6) doit retourner [1, 2, 4]

Attention : la liste stock ne doit pas être modifiée.

Question 4 — Programme principal

Écrire un programme principal qui :

1. crée la liste suivante : stock = [12, 5, 0, 8, 3]
2. demande à l'utilisateur le numéro d'un produit ;
3. indique si ce produit est disponible ;
4. affiche le nombre de produits en rupture ;
5. demande un seuil de réapprovisionnement ;
6. affiche les numéros des produits à réapprovisionner.