Correction DS bases de la programmation

Durée de l'épreuve : **01h50**

L'usage de la calculatrice n'est pas autorisé.
Le candidat répond sur feuilles doubles numérotées et garde l'énoncé.
Les traces de recherche, même incomplètes ou infructueuses, seront valorisées.
La qualité de la rédaction, la clarté et la précision des raisonnements seront prises en compte.

Exercice 1 (10 points)

1. ***Valeur de variables (1 pt)***
Déterminer la valeur des variables a, b, c, d après exécution du programme ci-dessous :

```
a = 3
b = a
c = a + b
a = a + 1
b = c % b
d = c / a
```

a = 4 , b = 0 , c = 6 , d = 1.5

2. ***Listes (1 pt)***
Déterminer la valeur des variables a, b, c, d après exécution du programme ci-dessous :

```
liste_1 = [1, 2, 3, 4]
a = liste_1[1]
liste_2 = [4, 3, 2, 1]
liste_3 = [liste_1, liste_2, liste_1]
b = liste_3[1]
c = liste_3[1][2]
d = b[3]
```

a = 2 , b = [4, 3, 2, 1] , c = 2 , d = 1

3. ***Erreurs (2 pts)***
Trouver toutes les erreurs dans le programme ci-dessous :

```
i = 0
while i < 3
annee = input("annee de naissance")
age = 2024 - annee
print("vous avez", anne, "ans", prenom)
```

```
i = 0
while i < 3:   # manque les deux points + indentations après le if
    annee = input("annee de naissance")
    age = 2024 - int(annee)          # manque la conversion d'année en int
    print("vous avez", annee, "ans")
    # manque un e à la variable annee + on veut âge + variable prenom n'existe pas
    i += 1 # manque l'incrémentation : boucle infinie
```

4. ***Parité (2 pts)***
Implémenter une fonction ***estPair***, qui prend en paramètre un entier ***n***, et qui renvoie ***True*** si ***n*** est pair, et ***False*** sinon :


```
assert estPair(10) == True
assert estPair(15) == False
```

```
def estPair(n: int) -> bool:
    if n % 2 == 0:
        return True
    else:
        return False
```

5. ***Retards (2 pts)***

Écrire un programme qui affiche un message à l'élève selon son heure d'arrivée :

- avant 7h45, le programme affichera : "Bienvenue" ;
- entre 7h45 et 7h50, le programme affichera : "Tu dois arriver à 7h45" ;
- entre 7h50 et 7h52, le programme affichera : "Carnet" ;
- après 7h52, le programme affichera : "Siiir"

On dispose d'une fonction ***heure()*** qui renvoie l'heure sous format d'un entier, par exemple s'il est 7h40, la fonction heure renvoie l'entier 740.

```
arrivee = heure()
... (plusieurs lignes à compléter)
```

```
arrivee = heure()
if arrivee < 745:
    print("Bienvenue")
elif arrivee < 750:
    print("Tu dois arriver à 7h45")
elif arrivee < 752:
    print("Carnet")
else:
    print("Siiir")
```

6. ***Mot de passe (2 pts)***

Implémenter une fonction ***changeMdp***, qui prend en paramètre un mot de passe ***mdp (str)***, et qui :

- demande à l'utilisateur son mot de passe jusqu'à qu'il soit correct (c'est à dire identique à ***mdp***, le paramètre de la fonction) ;
- lui demande de saisir un nouveau mot de passe ;
- lui demande de resaisir son nouveau mot de passe pour confirmation jusqu'à qu'il corresponde ;
- retourne le nouveau mot de passe.

```
def changeMdp(mdp: str) -> str:
    # vérification du mot de passe
    rep = input('Mot de passe : ')
    while rep != mdp:
        print('Mot de passe incorrect')
        rep = input('Mot de passe : ')
    # nouveau mot de passe
    new_mdp = input('Nouveau mot de passe : ')
    # confirmation du nouveau mot de passe
    new_mdp2 = input('Confirmer votre nouveau mot de passe : ')
    while new_mdp2 != new_mdp:
        print('Le mot de passe saisi est différent')
        new_mdp2 = input('Confirmer votre nouveau mot de passe : ')
    return new_mdp
```


Exercice 2 (5 points)

1. Écrire un programme qui affiche les nombres 1 à 30 sur une ligne : (1 pt)
   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

   Pour rappel, par défaut, après exécution de print, on saute une ligne.
   Pour changer ce comportement, on peut utiliser le paramètre ***end***.
   Par exemple : ***print("bonjour", end=" ")*** ajoutera un espace après bonjour et continuera sur la même ligne au prochain ***print***.

```
for i in range(1, 31):
    print(i, end=' ')
print()
```

2. Écrire, à l'aide d'une boucle, un programme qui affiche les nombre 0, 1, et 2 en boucle sur une ligne avec un total de 40 nombres : (1 pt)
   0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0

```
for i in range(40):
    print(i % 3, end=' ')
print()
```

3. Écrire, à l'aide d'une boucle, un programme qui affiche le motif ci après : (1pt)

```
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
```

```
for ligne in range(10):
    print('* ' * 10)
print()
```

4. Écrire un programme qui, à l'aide d'une boucle, affiche le motif ci après : (1pt)

```
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
```

```
for ligne in range(1, 11):
    print('* ' *  ligne)
print()
```

5. Écrire un programme qui, à l'aide de boucles, affiche le tableau ci après : (1pt)

   11 12 13 14 15 16 17 18 19
   21 22 23 24 25 26 27 28 29
   31 32 33 34 35 36 37 38 39
   41 42 43 44 45 46 47 48 49
   51 52 53 54 55 56 57 58 59
   61 62 63 64 65 66 67 68 69
   71 72 73 74 75 76 77 78 79

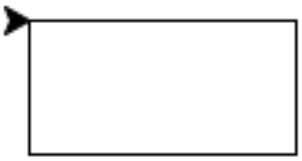
81 82 83 84 85 86 87 88 89
91 92 93 94 95 96 97 98 99

```
for ligne in range(1,10):
    for colonne in range(1,10):
        print(10*ligne + colonne, end=' ')
    print()
print()
```

Exercice 3 (5 points)

1. Écrire une fonction ***rectangle(longueur, largeur)*** qui, à l'aide de ***turtle***, dessine un rectangle : (1 pt)

```
rectangle(100,50)
```

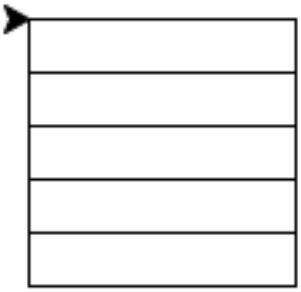


```
from turtle import *

def rectangle(longueur, largeur):
    for i in range(2):
        forward(longueur)
        right(90)
        forward(largeur)
        right(90)
```

2. Écrire une fonction ***tiroirs(n, longueur, hauteur)*** qui, à l'aide de ***turtle***, dessine ***n*** tiroirs : (2 pts)

```
tiroirs(5, 100, 20)
```

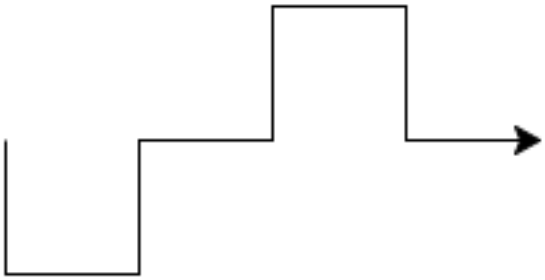


```
def tiroirs(n, longueur, hauteur):
    for i in range(n):
        rectangle(longueur, hauteur)
        penup()
        right(90)
        forward(hauteur)
        left(90)
        pendown()
    penup()
    left(90)
    forward(n*hauteur)
    right(90)
    pendown()
```

3. Écrire une fonction ***parcoursGaucheDroite(directions, longueur)*** qui, à l'aide de ***turtle*** déplace la tortue : (2 pts)
   - la liste ***directions*** contient des directions sous le format ***"G"*** pour gauche et ***"D"*** pour droite ;

- pour chaque élément de la liste ***directions***, la tortue s'oriente à gauche de 90° ou à droite de 90° et avance de ***longueur***.

```
parcoursGaucheDroite(['D', 'G', 'G', 'D', 'G', 'D', 'D', 'G'], 50)
```

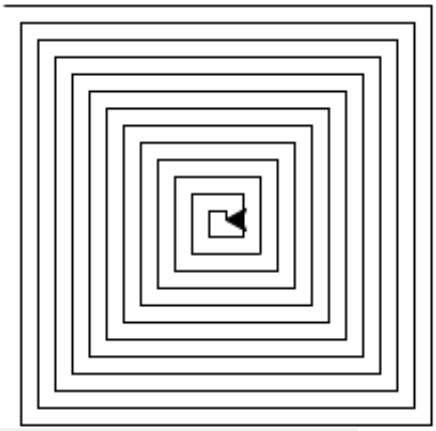


```
def parcoursGaucheDroite(directions, longueur):
    for direction in directions:
        if direction == 'G':
            left(90)
            forward(longueur)
        elif direction == 'D':
            right(90)
            forward(longueur)
```

Exercice bonus (optionnel)

Écrire une fonction ***escargot(longueur, esp)*** qui, à l'aide de ***turtle***, dessine le motif ci-après : (1 pt)

```
escargot(200, 4)
```



```
def escargot(longueur, esp):
    while longueur > 0 :
        forward(longueur)
        right(90)
        longueur = longueur - esp
```

