

Spécialité NSI Terminale
Correction du contrôle n° 1
Sujet A
Lundi 25 septembre 2023

A. Structures élémentaires

1. On peut écrire le programme suivant :

```
n = int(input("Entrez le nombre entier positif n : "))
somme = 0
for k in range(0, n+1, 5) :
    somme = somme + k
print("La somme est égale à", somme)
```

ou bien :

```
n = int(input("Entrez le nombre entier positif n : "))
somme = 0
for k in range(0, n+1) :
    if n%5 == 0 :
        somme = somme + k
print("La somme est égale à", somme)
```

2. a. On peut écrire le programme suivant :

```
montant = 3000
annee = 2015
for a in range(2016, 2025) :
    montant = montant*1.04 + 25
print("Au 2 janvier 2024 le montant est de", montant,
      "euros")
```

b. On peut écrire le programme suivant :

```
montant = 3000
annee = 2015
while montant < 4000 :
    montant = montant*1.04 + 25
    annee = annee + 1
print("La somme dépassera 4000 € en", annee)
```

B. Fonctions

3. a. Les variables locales a, b et c évoluent ainsi :

a	b	c
3	4	5
-7	12	5

Le fonction `maFct` renvoie donc $5 * (-7) + 12$ donc -23.

Il s'affichera donc le message suivant dans la console :

Le résultat est -23.

b. Les variables globales p, q et r n'ont pas évolué au cours de l'exécution on a donc finalement :

p : 5 , q : 4, r : 3

4. a. On peut écrire le programme suivant :

```
def factorielle(n) :  
    fact = 1  
    for k in range(1, n+1) :  
        fact = fact*k  
    return fact
```

b. On peut écrire le programme principal suivant :

```
res = factorielle(5)  
print("La factorielle de 5 est", res)
```

C. Tableaux et fonctions (6 points)

5. On peut compléter le code de la façon suivante :

```
def sommeTab(tab) :  
    somme = 0  
    for n in tab :  
        somme = somme + n  
    return somme
```

ou bien :

```
def sommeTab(tab) :  
    somme = 0  
    for n in range(len(tab)) :  
        somme = somme + tab[n]  
    return somme
```

6. On peut écrire le code suivant :

```
def mini(tab) :  
    minimum = tab[0]  
    indMin = 0  
    for i in range(len(tab)) :  
        if tab[i] < minimum :  
            minimum = tab[i]  
            indMin = i  
    return minimum, indMin
```

7. a. On peut écrire le code suivant :

```
def diviseurs(n) :  
    tabDiv = []  
    for k in range(1, n+1) :  
        if n%k == 0 :  
            tabDiv.append(k) # ou tabDiv = tabDiv + [k]  
    return tabDiv
```

b. On peut écrire le programme principal suivant :

```
res = diviseurs(48615)  
print("Le tableau des diviseurs de 48615 est", res)
```

D. Matrices

8. Après exécution du code les variables ont les valeurs suivantes :

n1 : 2 , n2 : 5 et x : 9

9. On peut écrire le code suivant :

```
def sommeColonne(mat, n)
    lg = len(mat)
    somme = 0
    for k in range(lg)
        somme = somme + mat[k][n-1]
    return somme
```

E. Épreuve pratique

10. On peut compléter le code de la façon suivante :

```
def moyenne(notes) :
    total_points = 0
    somme_coefs = 0
    for n in range(len(notes)) :
        coef = n[1]
        val = n[0]
        total_points = total_points + val*coef
        somme_coefs = somme_coefs + coef
    if somme_coefs != 0 :
        return total_points/ somme_coefs
    else :
        return None
```