

# **BACCALAURÉAT**

**SESSION 2023**

---

**Épreuve de l'enseignement de spécialité**

## **NUMÉRIQUE et SCIENCES INFORMATIQUES**

**Partie pratique**

**Classe Terminale de la voie générale**

---

**Sujet n°17**

---

**DURÉE DE L'ÉPREUVE : 1 heure**

**Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3  
Dès que le sujet vous est remis, assurez-vous qu'il est complet.**

*Le candidat doit traiter les 2 exercices.*

### EXERCICE 1 (4 points)

Écrire une fonction `moyenne(liste_notes)` qui renvoie la moyenne pondérée des résultats contenus dans la liste `liste_notes`, non vide, donnée en paramètre. Cette liste contient des couples `(note, coefficient)` dans lesquels :

- `note` est un nombre de type `float` compris entre 0 et 20 ;
- `coefficient` est un nombre entier strictement positif.

Ainsi, l'expression `moyenne([(15, 2), (9, 1), (12, 3)])` devra renvoyer 12.5 :

$$\frac{15 \times 2 + 9 \times 1 + 12 \times 3}{2 + 1 + 3} = 12,5$$

### EXERCICE 2 (4 points)

On cherche à déterminer les valeurs du triangle de Pascal (Figure 1).

Dans le triangle de Pascal, chaque ligne commence et se termine par le nombre 1. Comme l'illustre la Figure 2, on additionne deux valeurs successives d'une ligne pour obtenir la valeur qui se situe sous la deuxième valeur.

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

Figure 1 : triangle de Pascal

```
1
1 + 1
1 2 1
1 3 + 3 1
1 + 4 6 4 1
1 5 10 10 5 1
```

Figure 2 : méthode de calcul

Compléter la fonction `pascal` ci-après prenant en paramètre un entier `n` supérieur ou égal à 2. Cette fonction doit renvoyer une liste correspondant au triangle de Pascal de la ligne 0 à la ligne `n`. Le tableau représentant le triangle de Pascal sera contenu dans la variable `triangle`.

```
def pascal(n):
    triangle= [[1]]
    for k in range(1,...):
        ligne_k = [...]
        for i in range(1, k):
            ligne_k.append(triangle[...][i-1] + triangle[...][...])
        ligne_k.append(...)
        triangle.append(ligne_k)
    return triangle
```

Pour `n = 4`, voici ce que l'on devra obtenir :

```
>> pascal(4)
[[1], [1, 1], [1, 2, 1], [1, 3, 3, 1], [1, 4, 6, 4, 1]]
```

Et pour `n = 5`, voici ce que l'on devra obtenir :

```
>> pascal(5)
[[1], [1, 1], [1, 2, 1], [1, 3, 3, 1], [1, 4, 6, 4, 1],
 [1, 5, 10, 10, 5, 1]]
```