

# BACCALAURÉAT

SESSION 2025

---

Épreuve de l'enseignement de spécialité

## NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

---

Sujet n°20

---

DURÉE DE L'ÉPREUVE : 1 heure

**Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4  
Dès que le sujet vous est remis, assurez-vous qu'il est complet.**

*Le candidat doit traiter les 2 exercices.*

## EXERCICE 1 (10 points)

Écrire une fonction `min_et_max` qui prend en paramètre un tableau de nombres `tab` non vide, et qui renvoie la plus petite et la plus grande valeur du tableau sous la forme d'un dictionnaire à deux clés `min` et `max`.

Les tableaux seront représentés sous forme de liste Python.

L'utilisation des fonctions natives `min`, `max` et `sorted`, ainsi que la méthode `sort` n'est pas autorisée.

Exemples :

```
>>> min_et_max([0, 1, 4, 2, -2, 9, 3, 1, 7, 1])
{'min': -2, 'max': 9}
>>> min_et_max([0, 1, 2, 3])
{'min': 0, 'max': 3}
>>> min_et_max([3])
{'min': 3, 'max': 3}
>>> min_et_max([1, 3, 2, 1, 3])
{'min': 1, 'max': 3}
>>> min_et_max([-1, -1, -1, -1, -1])
{'min': -1, 'max': -1}
```

## EXERCICE 2 (10 points)

On dispose d'une classe `Carte` permettant de créer des objets modélisant des cartes à jouer.

Compléter la classe `Paquet_de_cartes` suivante en respectant les spécifications données dans les chaînes de documentation.

Ajouter une assertion dans la méthode `recuperer_carte` de la classe `Paquet_de_cartes` afin de vérifier que le paramètre `pos` est correct.

On rappelle que l'instruction

**assert** condition, message

permet de vérifier que la condition est vraie. Si ce n'est pas le cas, le programme s'arrête et affiche le message d'erreur fourni.

```
class Carte:
    def __init__(self, c, v):
        """Initialise les attributs couleur (entre 1 et 4),
        et valeur (entre 1 et 13). """
        self.couleur = c
        self.valeur = v

    def recuperer_valeur(self):
        """ Renvoie la valeur de la carte :
        As, 2, ..., 10, Valet, Dame, Roi """
        valeurs = ['As', '2', '3', '4', '5', '6', '7', '8',
                   '9', '10', 'Valet', 'Dame', 'Roi']
        return valeurs[self.valeur - 1]

    def recuperer_couleur(self):
        """ Renvoie la couleur de la carte
        (parmi pique, coeur, carreau, trèfle). """
        couleurs = ['pique', 'coeur', 'carreau', 'trèfle']
        return couleurs[self.couleur - 1]

class Paquet_de_cartes:
    def __init__(self):
        """ Initialise l'attribut contenu avec une liste des 52
        objets Carte possibles rangés par valeurs croissantes en
        commençant par pique, puis cœur, carreau et trèfle. """
        ...
        ...
        ...

    def recuperer_carte(self, pos):
        """ Renvoie la carte qui se trouve à la position pos
        (entier compris entre 0 et 51). """
        ...
        ...
```

Exemple :

```
>>> jeu = Paquet_de_cartes()
>>> carte1 = jeu.recuperer_carte(20)
>>> carte1.recuperer_valeur() \
    + " de " + carte1.recuperer_couleur()
"8 de coeur"
>>> carte2 = jeu.recuperer_carte(0)
>>> carte2.recuperer_valeur() \
    + " de " + carte2.recuperer_couleur()
"As de pique"
>>> carte3 = jeu.recuperer_carte(52)
AssertionError : paramètre pos invalide
```