

BACCALAUREAT

SESSION 2022

Épreuve de l'enseignement de spécialité

NUMERIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°19

DUREE 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)

Programmer la fonction `multiplication` prenant en paramètres deux nombres entiers `n1` et `n2`, et qui renvoie le produit de ces deux nombres.

Les seules opérations autorisées sont l'addition et la soustraction.

Exemples :

```
>>> multiplication(3,5)
15
>>> multiplication(-4,-8)
32
>>> multiplication(-2,6)
-12
>>> multiplication(-2,0)
0
```

EXERCICE 2 (4 points)

Soit `T` un tableau non vide d'entiers triés dans l'ordre croissant et `n` un entier.

La fonction `chercher`, donnée à la page suivante, doit renvoyer un indice où la valeur `n` apparaît éventuellement dans `T`, et `None` sinon.

Les paramètres de la fonction sont :

- `T`, le tableau dans lequel s'effectue la recherche ;
- `n`, l'entier à chercher dans le tableau ;
- `i`, l'indice de début de la partie du tableau où s'effectue la recherche ;
- `j`, l'indice de fin de la partie du tableau où s'effectue la recherche.

La fonction `chercher` est une fonction récursive basée sur le principe « diviser pour régner ».

Le code de la fonction commence par vérifier si $0 \leq i$ et $j < \text{len}(T)$. Si cette condition n'est pas vérifiée, elle affiche "Erreur" puis renvoie `None`.

Recopier et compléter le code de la fonction `chercher` proposée ci-dessous :

```
def chercher(T,n,i,j):
    if i < 0 or ??? :
        print("Erreur")
        return None
    if i > j :
        return None
    m = (i+j) // ???
    if T[m] < ??? :
        return chercher(T, n, ??? , ???)
    elif ??? :
        return chercher(T, n, ??? , ??? )
    else :
        return ???
```

L'exécution du code doit donner :

```
>>> chercher([1,5,6,6,9,12],7,0,10)
Erreur
>>> chercher([1,5,6,6,9,12],7,0,5)
>>> chercher([1,5,6,6,9,12],9,0,5)
4
>>> chercher([1,5,6,6,9,12],6,0,5)
2
```



```
def chercher(T, n, i, j):
    if i < 0 or j >= len(T) :
        print("Erreur")
        return None
    if i > j :
        return None
    m = (i + j) // 2
    if T[m] < n :
        return chercher(T, n, m + 1 , j)
    elif n < T[m] :
        return chercher(T, n, i , m - 1 )
    else :
        return m

assert chercher([1,5,6,6,9,12],7,0,10) == None
assert chercher([1,5,6,6,9,12],7,0,5) == None
assert chercher([1,5,6,6,9,12],9,0,5) == 4
assert chercher([1,5,6,6,9,12],6,0,5) == 2
```