BACCALAUREAT

SESSION 2022

Épreuve de l'enseignement de spécialité

NUMERIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°08

DUREE DE L'EPREUVE : 1 heure

Le sujet comporte 2 pages numérotées de 1 / 2 à 2 / 2
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 `recherche` qui prend en paramètres `elt` un nombre entier et `tab` un tableau de nombres entiers, et qui renvoie l'indice de la première occurrence de `elt` dans `tab` si `elt` est dans `tab` et `-1` sinon.

Exemples :

```
>>> recherche(1, [2, 3, 4])
-1
>>> recherche(1, [10, 12, 1, 56])
2
>>> recherche(50, [1, 50, 1])
1
>>> recherche(15, [8, 9, 10, 15])
3
```

EXERCICE 2 (4 points)

On considère la fonction `insere` ci-dessous qui prend en argument un entier `a` et un tableau `tab` d'entiers triés par ordre croissant. Cette fonction insère la valeur `a` dans le tableau et renvoie le nouveau tableau. Les tableaux seront représentés sous la forme de listes python.

```
def insere(a, tab):
    l = list(tab) #l contient les mêmes éléments que tab
    l.append(a)
    i = ...
    while a < ... and i >= 0:
      l[i+1] = ...
      l[i] = a
      i = ...
    return l
```

Compléter la fonction `insere` ci-dessus.

Exemples :

```
>>> insere(3,[1,2,4,5])
[1, 2, 3, 4, 5]
>>> insere(10,[1,2,7,12,14,25])
[1, 2, 7, 10, 12, 14, 25]
>>> insere(1,[2,3,4])
[1, 2, 3, 4]
```

```python
from doctest import testmod


def recherche(elt, tab):
    """ Recherche la valeur elt dans le tableau.

    Args:
        elt (int) : nombre à rechercher

    Returns:
        int : indice de la valeur recherchée ou
            -1 si absent du tableau

    Tests et Exemples:
    >>> recherche(1, [2, 3, 4])
    -1
    >>> recherche(1, [10, 12, 1, 56])
    2
    >>> recherche(50, [1, 50, 1])
    1
    >>> recherche(15, [8, 9, 10, 15])
    3
    """
    n = len(tab)
    for i in range(n):
        if tab[i] == elt:
            return i
    return -1


assert recherche(1, [2, 3, 4]) == -1
assert recherche(1, [10, 12, 1, 56]) == 2
assert recherche(50, [1, 50, 1]) == 1
assert recherche(15, [8, 9, 10, 15]) == 3
testmod()
```

```python
from doctest import testmod


def insere(a, tab):
    """Fait une copie d'un tableau trié par ordre
    croissant en y insérant la valeur a.

    Args:
    a (int): valeur à insérer
    tab (list): tableau trié

    Returns:
    list: copie du tableau tab avec
    la valeur a insérée

    Tests et Exemples:
    >>> insere(3,[1,2,4,5])
    [1, 2, 3, 4, 5]
    >>> insere(10,[1,2,7,12,14,25])
    [1, 2, 7, 10, 12, 14, 25]
    >>> insere(1,[2,3,4])
    [1, 2, 3, 4]
    """
    l = list(tab) #l contient les mêmes éléments que tab

    # l'algorithme choisi ressemble au tri à bulle :
    #   on ajoute la valeur à insérer à la fin du tableau
    #   puis on la fait descendre petit à petit jusqu'à sa place correcte
    l.append(a)

    # initialisation de la boucle avec la première valeur
    # à tester, c'est-à-dire la case située à gauche de a
    i = len(l) - 2

    # Précondition :
    #  * a est à la position i+1
    #  * a est < que toutes les valeurs des cases
    #  situées après lui (i+2, i+3, ...)
    # Condition d'arrêt :
    #  * il n'y a plus de case avant a
    # OU
    #  * a >= à la valeur située avant lui
    while a < l[i] and i >= 0:
        # puisque a est plus petit que l[i]
        # permuter l[i] et a
        l[i+1] = l[i]
        l[i] = a

        # mise à jour du compteur de boucle
        i = i - 1

    return l


assert insere(3,[1,2,4,5]) == [1, 2, 3, 4, 5]
assert insere(10,[1,2,7,12,14,25]) == [1, 2, 7, 10, 12, 14, 25]
assert insere(1,[2,3,4]) == [1, 2, 3, 4]
testmod()
```