DEVOIR SURVEILLÉ 2 – 1 HEURE

CI 2 : ALGORITHMIQUE ET PROGRAMMATION

Objectifs :
- comprendre un algorithme et expliquer ce qu'il fait ;
- modifier un algorithme existant pour obtenir un résultat différent ;
- concevoir un algorithme répondant à un problème précisément posé ;
- expliquer le fonctionnement d'un algorithme.

Vous pourrez répondre aux questions en utilisant du pseudo code ou du code Python.

Exercice

L'objectif est de trier un tableau dans le but de diminuer le temps nécessaire à la recherche d'un ou de plusieurs éléments. Une étape préalable à la recherche d'un élément est le tri du tableau.

Avant tout on se propose d'écrire la fonction permute qui permet de permuter deux éléments d'un tableau. Si on veut permuter le premier élément et le second élément d'un tableau, la fonction doit avoir le comportement suivant :

```
>>> tab=[10,20,30]
>>> permute(tab,0,1)
>>> print(tab)
                    [20,10,30]
```

Question 1

Écrire la fonction permettant de permuter les valeurs du tableau. Cette fonction devra correspondre aux spécifications suivantes :

- *nom de la fonction : permute ;*
- *arguments de la fonction : un tableau, les deux indices à permuter.*

On propose maintenant un algorithme permettant de trier le tableau à proprement parlé. Ce tri est appelé tri par sélection. Le voici :

```
def tri(tab):
    for i in range(0,len(tab)):
        indice = i
        for j in range(i+1,len(tab)):
            if tab[j]<tab[indice]:
                indice = j
        permute(tab,i,indice)
    return tab
```

Question 2

Soit le tableau tab $= [2,3,1,4]$. *Pour chaque valeur de* i *et pour chaque valeur de* j, *indiquer le contenu du tableau tab. Pour répondre à la question on utilisera le tableau donné en fin de sujet. On le remplira à partir de la double barre. Les colonnes* i, j, *indice et tab seront à remplir intégralement. Les * sont à remplacer par les valeurs correspondantes.*

Question 3

Écrire la fonction qui vérifie si un tableau est trié. Cette fonction devra correspondre aux spécifications suivantes :

- *nom de la fonction : is_sorted ;*
- *argument de la fonction : un tableau ;*
- *retour de la fonction : la fonction doit retourner la valeur booléenne True si le tableau est trié. Elle devra retourner la valeur booléenneFalse si le tableau n'est pas trié.*

On utilisera une boucle ***Pour.***

Question 4

Réécrire la fonction précédente en utilisant exclusivement une boucle ***Tant que****.*

Question 5

Des deux structures proposées, estimez laquelle peut être le plus efficace ? (c'est à dire, laquelle nécessite, le cas échéant, le moins d'opérations).

Dans le but de diminuer le temps d'exécution lors de la recherche d'éléments dans le tableau, on propose un algorithme permettant de vérifier qu'une valeur se trouve dans un tableau.

La méthode consiste en découper le tableau en 2. On regarde ensuite dans quelle partie du tableau est susceptible de se trouver la valeur recherchée. On redivise en 2 la partie de tableau considérée et on regarde dans quelle partie du tableau est susceptible de se trouver la valeur recherchée *etc.* La fonction doit renvoyer à l'utilisateur l'index de l'élément recherché s'il existe et qui renvoie None sinon. On donne le programme suivant :

```python
1  def recherche_dichotomique(x, a):
2      g, d = 0, len(a)-1
3      while g <= d:
4          m = (g + d) // 2
5          if a[m] == x:
6          if a[m] < x:
7              g = m+1
8          else:
9              d = m-1
```

Question 6

De quel type sont les variables x *et* a *?*

Question 7

Quelle opération est effectuée ligne 4 ? Expliquer ce choix.

Question 8

Pour chaque itération de la boucle while, quelle est l'étendue de la zone de recherche ?

Question 9

Compléter l'algorithme (en recopiant les lignes qui vous semble nécessaire) pour qu'il renvoie l'index de l'élément recherché.

Question 10

L'algorithme a-t-il le comportement souhaité ? Si ce n'est pas le cas, compléter l'algorithme.

Question 11

A partir des fonctions définies précédemment, écrire une fonction permettant, à partir d'un tableau d'entier quelconque (trié ou non trié), de dire si un élément appartient au tableau ou non :

- *données d'entrées de la fonction : un tableau, un nombre ;*
- *données de sortie de la fonction : un booléen.*

Question 12

Quel est l'intérêt de trier un tableau dans le cas où on cherche une valeur dans le tableau ? Quel est l'intérêt de le trier lorsqu'on cherche plusieurs valeurs ? Commenter.

NOM :

Commentaires			i	j	indice	tab
Instant initial			–	–	–	[2,3,1,4]
Pour i allant de 0 à 3 :			–	–	–	[2,3,1,4]
Pour $i=0$			0	–	–	[2,3,1,4]
Affectation			0	–	0	[2,3,1,4]
Pour j allant de 1 à 3			0	–	0	[2,3,1,4]
	Pour $j=1$		0	1	0	[2,3,1,4]
	tab[1]<tab[0]	$\Rightarrow$ 3<2 est faux	0	1	0	[2,3,1,4]
	Pas d'affectation de l'indice		0	1	0	[2,3,1,4]
	Pour $j=2$		0	1	0	[2,3,1,4]
	tab[2]<tab[0]	$\Rightarrow$ 1<2 est vrai	0	2	0	[2,3,1,4]
	Affectation de indice		0	2	2	[2,3,1,4]
	Pour $j=*$					
	tab[*]<tab[*]					
	Affectation ?					
Permutation ?						
Pour $i=*$						
Affectation						
Pour j allant de ** à **						
	Pour $j=*$					
	tab[*]<tab[*]					
	Affectation ?					
	Pour $j=*$					
	tab[*]<tab[*]					
	Affectation ?					
	Pour $j=*$					
	tab[*]<tab[*]					
	Affectation ?					
Permutation ?						