Université Cadi Ayad- Marrakech

Faculté des Sciences – Semlalia

Département Informatique

TD4

Algorithmique II SMI3

Exercice 1 :

On se propose ici d'écrire une variante du tri par insertion vu en cours en améliorant la phase d'insertion. On rappelle le principe du tri par insertion : à chaque étape de la boucle, on suppose que les k premiers éléments sont déjà triés et on cherche à trouver où insérer le (k + 1)-ième élément, on l'insère alors en décalant vers la droite les éléments plus grands. L'insertion dichotomique fonctionne ainsi : on commence par regarder si T[k] doit être inséré entre l'indice 0 et k/2 ou entre l'indice k/2 et k-1. Ensuite on recommence en coupant de nouveau en deux les sous tableaux obtenu, et ainsi de suite, jusqu'à qu'on ait exactement identifié la bonne position pour T(k).

T =	2	6	12	13	16	18	28	30	3

- Écrire un algorithme qui fait l'insertion dichotomique du k-ième élément de façon itérative. La fonction prendra en entrée T et k et renverra l'entier correspondant à la bonne position de T(k).
- Ecrire un algorithme qui fait la même chose de façon récursive.

Exercice 2 :

Déclarer La structure suivante :

1. **Un article** est défini par : numéro (entier : short), libellé (chaîne de 29 caractères), quantité en stock (entier : short), prix (réel : float).
2. Ecrivez une fonction, nommée **SaisieArticle**, qui saisit les champs d'une variable article passé en paramètre
3. Ecrivez une fonction, nommée **AfficheArticle**, qui affiche le contenu des champs d'une variable article passé en paramètre
4. Ecrivez une fonction nommée **SaisieTabArticle**, qui remplit un tableau T de n articles. T et n sont des paramètres de la fonction.
5. Ecrivez une fonction **AfficheTabStock**, qui affiche les articles de T ayant une quantité en stock >= à une valeur q. T, n et q sont des paramètres de la fonction

6. Ecrivez un programme qui fait appel aux fonctions **SaisieTabArticle** et **AfficheTabStock**

Exercice 3 :

Etudier la complexité des algorithmes suivants :

1.

```
x = 0
for(int i = 1 ; i < tab.length-1 ; i++) {
  for(int j = 0 ; j < 3 ; j++) {
     x = x + tab[i - 1 + j] * (j+1)
  }
}
```

2.

```
i = 0
j = 0
while(i < n) {
  if( i % 2 == 0) {
    j = j + 1
  } else {
    j = j / 2
  }
  i = i + 1
}
```

Début

p:= a_0; q:=1;

pour i:=1 à n **faire**

q := q* x;

p:= p + a_i * q

fait

fin