Licence Sciences et Technologies	**ANNEE : 2006/2007** **SESSION DE JUIN 2007** **ETAPE :** INF2 MAT2 MAI4 **UE :** INF102 **Epreuve :** Initiation à l'algorithmique **Date :** 29 Juin 2007 **Heure :** 11 H 00 **Durée :** 1 H 30 **Documents :** Tous documents interdits **Vous devez répondre directement sur le sujet qui comporte 4 pages.** **Insérez ensuite votre réponse dans une copie d'examen comportant tous les renseignements administratifs** **Epreuve de** M^{me} Delest.

Indiquez votre code **d'anonymat** : N° : .

La notation tiendra compte de la clarté de l'écriture des réponses.

Barème

- Question 1 – Connaissances générales : 4 points
- Question 2 – Compréhension d'algorithme : 3 points
- Question 3 – Compréhension d'algorithme récursif : 4.5 points
- Question 4 – Ecriture de fonctions : 4 points
- Question 5 – Manipulation de listes : 4.5 Points

Question 1. Cochez les affirmations qui sont correctes :

☐ Le corps d'une boucle *tantque* est toujours exécuté une fois.
☐ L'*indice de boucle* peut-être une variable de type structure.
☐ Une variable passée par *référence* à une fonction est recopiée dans l'environnement de la fonction.
☒La récursivité est un mécanisme qui peut conduire à une complexité en mémoire très pénalisante.
☒ La complexité minimale des algorithmes de tri vu en cours est *O(n)*
☐ La complexité minimale du tri rapide est *O(n)*
☐ Les listes simplement chainées ont une complexité mémoire moins élevée que les listes doublement chainées.
☐ Dans le tri à sélection, les *k* premiers éléments du tableau ne sont pas à leur place définitive après *k* itérations.

Question 2. Soit la fonction suivante :

```
fonction mystere(ref T:tableau[1..NMAX] d'entier, val clé :entier) : entier
    var d,f :entier ;
    début
        d=1 ;f=NMAX ;
        répéter
            tant que d<=NMAX et T[d]=<clé faire
                d=d+1 ;
            fintantque
            tantque f>=1 et T[f]>clé faire
                f=f-1 ;
            fintantque
            si d<f alors
                echange(T,d,f) ;
                d=d+1 ;
                f=f-1 ;
            finsi
        jusqu'à d>f ;
        retourner(f) ;
    fin ;
finfonction
```

où la fonction *echange* est la fonction vue en cours qui échange deux éléments du tableau.

1. On considère le tableau A de dimension 8 contenant la séquence 2,8,7,4,3,6,5,1. Quel est l'effet de l'appel mystere(A, 6) sur le tableau A? Quelle est la valeur retournée par la fonction ?
   On a dans A la séquence 2, 1, 5, 4, 3, 6, 7, 8 et l'algorithme retourne la valeur 6. Pour information, l'algorithme range en début de tableau les valeurs inférieures ou égales à la clé et en fin de tableau les valeurs supérieures à la clé. La valeur renvoyée est la position de fin de la première zone.
2. Quel est la valeur retournée par la fonction si tous les nombres contenus dans le tableau sont inférieurs à la clé ? même question si tous les nombres sont supérieurs à la clé ?
   Si tous les nombres sont inférieurs à la clé, la valeur retournée est N. Si tous les nombres sont supérieurs à la clé, la valeur retournée est 0,
3. Expliquez et donnez la complexité maximale de cette fonction en nombre de tests.
   *La complexité maximale est O(n). En effet, il y a bien deux boucles imbriquées mais d (resp. f) est croissant (resp. décroissant). La boucle externe contrôle d et f, plus précisément si d>f la boucle s'arête donc le nombre de test de cette boucle ne peut dépasser N. Au total, il y aura donc 2*N tests d'où la complexité annoncée.*

Question 3. Soit la fonction suivante :

```
fonction recurse(ref T:tableau[1..NMAX] d'entier, val clé :entier, val p :entier) : entier ;
    début
        si (p>NMAX) alors
           retourner(0) ;
        sinon
            si T[p]<=clé alors
                retourner(1+recurse(T,clé,p+1))
            sinon
                retourner(recurse(T,clé,p+1));
            finsi
        finsi
    fin ;
finfonction
```

1. On considère le tableau A de dimension 4 contenant la séquence 2,8,7,4. Quel est le résultat de l'appel recurse(T,6,1) ? *…2……*
2. Donner la suite des appels récursifs provoqués par l'appel de la question précédente ainsi que l'état du contexte de la fonction à chaque appel.

	T	*clé*	*P*	*Val.Fonc*
recurse(T,6,1)	*@T*	*6*	*1*	
recurse(T,6,2)	*@T,@T*	*6,6*	*1,2*	
recurse(T,6,3)	*@T,@T,@T*	*6,6,6*	*1,2,3*	
recurse(T,6,4)	*@T,@T,@T,@T*	*6,6,6,6*	*1,2,3,4*	
recurse(T,6,5)	*@T,@T,@T,@T,@T*	*6,6,6,6,6*	*1,2,3,4,5*	
->0				*0*
	@T,@T,@T,@T	*6,6,6,6*	*1,2,3,4*	
->0				*0*
	@T,@T,@T	*6,6,6*	*1,2,3*	
->1+0				*1*
	@T,@T	*6,6*	*1,2*	
->1+1				*2*
	@T	*6*	*1*	
->2				*2*

3. Réécrire cette fonction de manière itérative.

```
fonction recurselter(ref T:tableau[1..NMAX] d'entier,val clé :entier) : vide ;
    var p,tmp :entier ;
    début
        tmp=0 ;
        pour p allant de 1 à NMAX faire
            si T[p]<=clé alors
                tmp=tmp+1 ;
            finsi ;
        finpour ;
        retourner(tmp) ;
     fin ;
finfonction
```

4. Comparer la complexité en mémoire de cette nouvelle fonction par rapport à la fonction récursive? Justifiez. Comparer la complexité en nombre de tests de cette nouvelle fonction par rapport à la fonction récursive? Justifiez.

   La complexité en mémoire est plus élevée dans la fonction récursive puisque chaque appel récursif donne lieu à la création d'un environnement. La complexité en nombre de tests est inchangée car si la boucle pour fait k tours dans la fonction itérative, il y aura aussi k appels récursifs et donc k tests dans la fonction récursive.

Question 4.

Soit *T* un tableau de dimension *N* contenant des points en mouvement dans R^3 decrits par leur coordonnées et leur vitesse exprimée par un nombre réel (on considère que les points se déplacent tous dans la même direction). Donnez une définition du type *pointEnMouvement* qui permet de décrire ces points. Ecrire une fonction *tropLoinTropVite* qui modifie le tableau *T* en supprimant les points qui sont à une distance supérieure à d de l'origine ou à une vitesse supérieure à v. Cette fonction retournera le nombre d'éléments ayant cette propriété.

On choisit la représentation d'un point vue en TD à laquelle on ajoute un champs vitesse.

```
type pointEnMouvement =structure
                    x,y,z,vitesse:réel
                finstructure
fonction tropLoinTropVite(ref T:tableau[1..NMAX] de pointEnMouvement,val d,v :réél) : entier ;
    var iT,cpt, :entier ;
    fonction distanceCarre(p :pointEnMouvement):reel;
        début
            retourner(p.x*p.x+p.y*p.y+p.z*p.z) ;
        fin
    finfonction
    début
        cpt=0 ;j=1 ;
        pour iT allant de 1 à NMAX faire
            si distanceCarre(T[iT])<=d*d ou T[iT].vitesse>v alors
                cpt=cpt+1;
            sinon
                T[j]=T[i] ;
                j=j+1 ;
            finsi
        finpour
        retourner(cpt) ;
     fin ;
finfonction
```

Question 5.

1. Il possible de gérer des listes simplement chainé en utilisant un tableau de dimension fixe ? Justifiez et donnez les inconvénients

   Voir cours Implémentation d'une liste par un tableau

2. Soit L une liste doublement chaînée d'entiers.
   - Que contient L et P après la séquence d'instructions suivante

```
var L :listeDC d'entier ;
var P :^entier ;
creerListe(L) ;
insererEnTete(7,L) ;
insererEnTete(4,L) ;
P=dernier(L) ;
insererAvant (6,L,P) ;
P=precedent(L,P) ;
insererAprès(12,L,P) ;
supprimerAprès(L,P) ;
P=premier(L) ;
supprimerAprès(L,P) ;
P=suivant(L,P) ;
```

   P=…… *adresse du second (@7)*….

   Contenu(P)= *7*……..

   L=…4, *7*……..……

3. Ecrire une fonction *puissanceDeux* dont le résultat est une liste contenant les puissances de deux dans l'ordre croissant et inférieures à un entier *x*.

```
fonction puissanceDeux (val x :entier) :listeSC d'entiers ;
    var L: listeSC d'entier;
    var p :entier ;
    var a :^entier ;
    début
       creerListe(L) ;
       si x>=1 alors
            insérerEnTete(1,L) ;
            a=premier(L) ;
            p=1 ;
            tantque p<x faire
                p=2*p;
                insererApres(p,L,a) ;
                p=suivant(L,p) ;
            fintantque ;
       finsi ;
       retourner(L);
     fin ;
finfonction
```

4. Quel est le nombre d'éléments de la liste construite à la question 3 pour un entier *x*?

…partie entière de $\log_2(x)$…($\lfloor \log_2(x) \rfloor$).……

FIN