TELECOM Lille1 **Mai 2012**

Nom : Prénom : N° : Durée : 2 heures

QROC Module A31TTN - Structures de données

Documents non autorisés, les réponses sont à donner sur la présente feuille.

Exercice 1 – *Récursivité*

Soit la relation de récurrence de la fonction Mystere définie par le code en C suivant :

```
void Mystere(int N)
{
if (N >0)
        {
        if (N/8 >0)
                        mystere(N/8) ;
                        printf("%d", N%8) ;
        }
}
```

1) Donner le résultat de Mystere(100). En déduire ce que fait cette fonction.

2) Sans faire de calcul de complexité, pouvez-vous intuitivement donner une idée sur le la complexité de cette fonction.

Exercice 2 – *Listes chaînées*

*On considère la déclaration de la structure de données ainsi que les fonctions **X**, **Y** et **Z** suivantes :*

```
struct element
{
   int val;
   struct element *nxt;
};
typedef element* llist;
```

```
llist X(llist liste)
{
   if(liste != NULL)
   {
       element* aRenvoyer = liste->nxt;
       free(liste);
      return aRenvoyer;
   }
   else
      return NULL;
}
```

1) Illustrer à l'aide d'un exemple (et un schéma) ce que fait cette fonction ***X***. Donnez un non significatif.

```
llist Y(llist liste)
{
   if(liste == NULL)
      return NULL;

  if(liste->nxt == NULL)
   {
      free(liste);
      return NULL;
   }
    element* tmp = liste;
   element* ptmp = liste;
```

```
    while(tmp->nxt != NULL)
    {
        ptmp = tmp;
        tmp = tmp->nxt;
    }
    ptmp->nxt = NULL;
    free(tmp);
    return liste;
}
```

2) Illustrer à l'aide d'un exemple (et un schéma) ce que fait cette fonction ***Y***. Donnez un non significatif.

```
llist Z(llist liste, int valeur)
{
    element *tmp=liste;
    while(tmp != NULL)
    {
        if(tmp->val == valeur)
            return tmp;
        tmp = tmp->nxt;
    }
    return NULL;
}
```

3) Illustrer à l'aide d'un exemple (et un schéma) ce que fait cette fonction ***Z***. Donnez alors un non significatif.

Exercice 3 – *Les arbres binaires*

On considère la suite de nombres suivante : 93, 96, 43, 61, 55, 17, 7, 60, 88, 5.

1) Partant d'un arbre binaire de recherche vide, on ajoute successivement les valeurs de la suite ci-dessus. Donnez en justifiant votre raisonnement l'arbre binaire résultat *(il ne s'agit pas de donner la version équilibrée de l'arbre)*.

2) Vérifier si l'arbre obtenu est équilibré, s'il n'est pas équilibré, équilibrez-le en recommençant de nouveau l'insertion de la séquence 93, 96, 43, 61, 55, 17, 7, 60, 88, 5. Expliquez les opérations de rotation effectuées.

3) On considère la structure de données suivante pour représenter cet arbre :

```
typedef struct Arbre {
        int Noeud;
        struct Arbre * SAG;
        struct Arbre * SAD;
} Arbre;
```

3.1) Proposez une fonction récursive qui permet de calculer la somme des feuilles d'un arbre.

3.2) Proposez une fonction récursive qui permet de calculer la hauteur d'un arbre.

3.3) Proposez une fonction récursive qui retourne si deux arbres sont différents.

Bonne chance.