Université Sultan Moulay Slimane **Année Universitaire 2015/2016**
Faculté des Sciences et Techniques
Béni Mellal
Département d'Informatique

Correction TD2 MIPC
Structures de Données en Langage C

```
#include <stdio.h>
#include <stdlib.h>
#define Max 30

typedef struct etudiant{
    char *nom;
    char *prenom;
    float note;
    struct etudiant *suivant;
} Etudiant;

typedef struct {
    Etudiant *Debut;
    int Nb_etudiants;
} LSC;
```

initialisation(LSC *L) {L->Debut=NULL; L->Nb_etudiants=0;}

```
Etudiant *preparerNoeud(char *nom,char *prenom, float note) {
Etudiant *e;
if ((e=(Etudiant *)malloc(sizeof(Etudiant)))==NULL) return NULL;
if ((e->nom=(char *)malloc(50*sizeof(char)))==NULL) return NULL;
if ((e->prenom=(char *)malloc(50*sizeof(char)))==NULL) return NULL;
strcpy(e->nom,nom);
strcpy(e->prenom,prenom);
e->note=note;
e->suivant=NULL;
return e;
}
```

```
int Insertion_debut_etudiant(LSC *L,Etudiant *e) {
if (e==NULL) {printf("pas d espace memoire pour ajouter un nouvel etudiant \n");return 0;}
if (L->Nb_etudiants>=Max) {printf("La classe est pleine \n"); return 0;}
if (L->Nb_etudiants==0) {L->Debut=e; L->Nb_etudiants=1;}
else {e->suivant=L->Debut; L->Debut=e; L->Nb_etudiants++; }
return 1;
}
```

```
int Insertion_Fin_etudiant(LSC *L,Etudiant *e) {
Etudiant *save;
if (e==NULL) {printf("pas d espace memoire pour ajouter un nouvel etudiant \n");return 0;}
if (L->Nb_etudiants>=Max) {printf("La classe est pleine \n"); return 0;}
if (L->Nb_etudiants==0) {L->Debut=e; L->Nb_etudiants=1;}
else {
    save=L->Debut;
    while(save->suivant!=NULL)save=save->suivant;
    save->suivant=e;
    L->Nb_etudiants++;
}
return 1;
}

int Detruit_classe(LSC *L) {
Etudiant *supp;
if (L->Debut==NULL) { printf("La classe est vide \n"); return 0; }
while(L->Debut!=NULL){
      supp=L->Debut;
      L->Debut=L->Debut->suivant;
      L->Nb_etudiants--;
      free(supp->nom);
      free(supp->prenom);
      free(supp);
}
return 1;
}

Affiche_Liste(LSC L) {
Etudiant *courant;
if (L.Debut==NULL) printf("La classe est vide \n");
else {
   courant=L.Debut;
   while(courant!=NULL) {
   printf("nom %s prenom %s a eu la note %f \n",courant->nom, courant->prenom,
                                                                    courant->note);
   courant=courant->suivant;
   }
   printf("Fin liste\n");
}
}

Divise_classe(LSC L, LSC *Admis, LSC *NAdmis){
Etudiant *s, *p;
initialisation(Admis); initialisation(NAdmis);
if (L.Debut==NULL) printf(" la classe est vide, impossible de faire la division \n");
else {
```

```
    p=L.Debut;
    while(p!=NULL){
        s=preparerNoeud(p->nom, p->prenom, p->note);
        if (p->note>=10) Insertion_debut_etudiant(Admis, s);
        else Insertion_debut_etudiant(NAdmis, s);
    }
   }
}
```

trier(LSC *L) {

```
Etudiant *s , *q, *t;
t=(Etudiant *)malloc(sizeof(Etudiant));
t->nom=(char *)malloc(50*sizeof(char));
t->prenom=(char *)malloc(50*sizeof(char));
int en_ordre = 0;
while(en_ordre==0) {
   en_ordre = 1;
       q=L->Debut; s=q->suivant;
       while(s->suivant!=NULL){
           if(q->note < s->note) { *t=*s; *s=*q; *s=*t; en_ordre=0;}
           q=s; s=s->suivant;
       }
   }
}
```

int main () {

```
int i,n,note,Nb_etudiants_Admis, Nb_etudiants_NAdmis;
char *nom,*prenom;
Etudiant *ListeAdmis, *ListeNAdmis,*etu;
printf("Donner le nombre d etudiants de votre classe"); scanf("%d",&n);
for(i=0;i<n;i++) {
            nom=malloc(20*sizeof(char));
            prenom=malloc(20*sizeof(char));
            printf("Donner le nom\n"); gets(nom);
            printf("Donner le prenom\n"); gets(nom);
            printf("Donner la note\n"); scanf("%d",&note);
            etu = preparerNoeud(nom, prenom, note);
            Insertion_debut_etudiant(etu,Debut,&Nb_etudiants);
}
affiche(Debut);
Divise_classe(Debut,ListeAdmis,ListeNAdmis,&Nb_etudiants_Admis,&Nb_etudiants_NAdmis);
trier(ListeAdmis);
affiche(ListeAdmis);
system("pause");
}
```