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

TD3 MIPC
Structures de Données en Langage C
(1 séance)

Exercice 1 :

```
# include <stdio.h>
# include <string.h>

typedef struct pile {
          char donnee;
          struct pile *precedent;
                                  } Pile;

     /* les variables globales*/
          Pile *sommet;
          int taille;

void initialisation () {
                  sommet = NULL;
                  taille = 0;
}

int Pile_vide() {
                  return (sommet == NULL);
}

void affiche() {
Pile *courant;  int i;
if (Pile_vide()) printf("la pile est vide \n");
else {  courant = sommet;
        for(i=1; i<=taille; i++) {
```

```
                    printf("   %c\n", courant->donnee);
                     courant = courant->precedent;
          }
    }
}

int pile_push (char donnee) {
Pile *element;
if ((element = (Pile *) malloc(sizeof(Pile))) ==NULL) return -1;
element->donnee= donnee;
element->precedent = sommet;
sommet = element;
taille++;
return 0;
}

char pile_pop ( ) {
Pile *supp_element;
char elt;
if (taille == 0)    return -1;
supp_element = sommet;
sommet = sommet->precedent;
elt=  supp_element->donnee;
free (supp_element);
taille--;
return elt;
}

main() {
char expression[40] ;
char S;
int lon,i, Valide;

printf("Entrer un mot "); gets(expression);
lon=strlen(expression);
initialisation();
if (((lon%2)==1)&&(expression[(lon/2)]=='c')) {
     i=0;
     while (i<(lon/2)) {
     if ((expression[i]=='a')||(expression[i]!='b')) pile_push(expression[i]);
```

```
     else break;
     i++;
     }

     if (i==(lon/2)) {
          Valide=1; i++;
          while ((i<=(lon-1)) && (Valide==1)) {
          S=pile_pop();
          if (S!=expression[i]) Valide=0;
          i++;
          } /*fin de while*/
     }
    if (Pile_vide() && (Valide==1)) printf("le mot est du langage \n");
    else printf("le mot n'est pas du langage \n");
}
else printf("le mot n'est pas du langage \n");
main();
system("pause");
}
```

Exercice 2 :

```
# include <stdio.h>
# include <string.h>

typedef struct file {
           int donnee ;
            struct file * suivant ;
           } File;

    typedef struct  {
         File *debut;
         File *fin;
          int taille;
                    } Var_File;

int file_push(Var_File *F, int donnee) {
File *nouvel_element;
if ((nouvel_element=(File *) malloc (sizeof(File)))==NULL) return -1;
nouvel_element->donnee=donnee;
nouvel_element->suivant = NULL;
```

```
if (F->taille == 0)  {F->fin = nouvel_element; F->debut = nouvel_element;}
else { F->fin->suivant = nouvel_element; F->fin = nouvel_element; }
F->taille++;
return 0;
}

int file_pop(Var_File *F) {
File *supp_element;
int info;
if (F->taille == 0) {printf("file vide \n");   return -1; }
supp_element = F->debut;
F->debut = F->debut->suivant;
if  (F->taille == 1) F->fin=NULL;
info=supp_element->donnee;
free (supp_element);
F->taille--;
return info;
}

int File_vide(Var_File F) {
return F.debut==NULL;
}

void affiche(Var_File F) {
File *courant;  int i;
if (F.taille==0) printf("la file est vide \n");
else {     courant = F.debut;
        for(i=1; i<=F.taille; i++) {
                    printf("   %d\n", courant->donnee);
                                courant = courant->suivant;
         }
    }
}

void initialisation_File(Var_File *F) {
    F->taille=0; F->fin=NULL; F->debut=NULL;
}
```

```
Maximum(Var_File *F, int max){
   Var_File *T;
   int s;

        if (F->taille==0)  printf("la file est vide \n ");
        else
                while (!File_vide(*F)) {
                        s=file_pop(F);
                        if (s<max) file_push(T,s);
                 }
}

main(){
Var_File *F;
int val,max ;
F=(Var_File *) malloc (sizeof(Var_File));
initialisation_File(F);
while (1) {
    printf("taper un entier et pour sortir taper la valeur 0 :");
    scanf("%d",&val);
    if (val==0) break;
    file_push(F,val);
}
printf(" \n taper la valeur max :"); scanf("%d",&max);
Maximum(F,max) ;
printf(" \n File apres suppression des valeurs >= %d :\n",max);
affiche(*F);
system("pause");
}
```