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

TD1 MIPC
Structures de Données en Langage C
(2 séances)

Exercice 1 :

Un palindrome est une chaine de caractères qui se lit de la même manière de gauche à droite et de droite à gauche. En utilisant une pile écrire un programme qui permet de vérifier si une chaine de caractère est palindrome.

Exemple :

AZIZA est palindrome

Mohammed n'est palindrome

Exercice 2 :

Définir une structure pile à l'aide d'un tableau d'entiers de N éléments au maximum et un sommet. On considérera que N est une constante donnée.

1. Ecrire les fonctions suivantes :
- **pile * Créer_InitialiserPile()** ; qui crée et initialise une pile vide,
- **int pileVide(pile p)** ; qui retourne 1 si la pile est vide et 0 sinon,
- **int depiler(pile *p)** ; qui retourne le dernier élément après l'avoir retiré de la pile,
- **void empiler(pile *p , int elt)** ; qui empile l'élément elt,
- **int SommetPile(pile p)** ; qui retourne le sommet de la pile.

On se donne trois piles P1, P2 et P3. La pile P1 contient une suite de nombres entiers positifs.

2. Ecrire une fonction pour déplacer les entiers de P1 dans P2 de façon à avoir dans P2 tous les nombres pairs au-dessus des nombres impairs.

3. Ecrire une fonction pour copier dans P2 les nombres pairs contenus dans P1. Le contenu de P1 après exécution de l'algorithme doit être identique à celui avant exécution. Les nombres pairs doivent être dans P2 dans l'ordre ou ils apparaissent dans P1.

Exercice 3 :

On se donne une file d'entiers que l'on voudrait trier avec le plus grand élément en fin de file. On n'est autorisé à utiliser que la fonction **fileVide**, **emfiler**, **defiler** et les opérations suivantes:
- **defilerEnfiler** : Défile le premier élément de la première file et l'ajoute à la deuxième file.
- **comparer** : Retourne 1 si le premier élément de la première file est plus grand ou égal au premier élément de la deuxième file et 0 sinon.

1. Définir les fonction **fileVide** ,**emfiler**, **defiler , defilerEnfiler** et **comparer.**

2. Donner un algorithme de tri qui utilise seulement ces opérations et 3 files. La file f1 contiendra les entiers à trier, file f2 contiendra les entiers triés après exécution et la file f3 pourra servir de file auxiliaire.

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

Correction TD1 MIPC
Structures de Données en Langage C

Exercice 1:

```
#include<stdio.h>
#define max 50
char p[max]; // déclaration de la  pile
int sommet ;

initialisation() {  sommet=-1; }

void empiler(char x) {
          if (sommet>=max-1) printf("pile pleine");
         else { sommet++; p[sommet]=x; }
}

char depiler () {
char x;
 if (sommet<0)  { printf("pile vide"); return '\0'; }
 else   { x=p[sommet];  sommet--;       return x;}
 }

int pilevide() { return sommet<0;}

int pilepleine() {return sommet>=max-1;}

main() {
  char a,ch[max];
  int i,lg,valide=1;
  printf("une donner une chaine\n"); gets(ch);
  lg=strlen(ch);
  for(i=0;i<lg/2;i++) empiler(ch[i]);
  if(lg%2==0)  for(i=lg/2;((i<lg) && (valide==1));i++) { a=depiler();   if(ch[i]!=a)  valide=0; }
  else for(i=lg/2+1; ((i<lg) && (valide==1));i++) {a=depiler();  if(ch[i]!=a)  valide=0;}
  if (pilevide() && (valide==1)) printf("chaine palindrome  \n");
  else printf("chaine non palindrome \n");
  system("pause");
 }
```

Exercice 2:

```
#include<stdio.h>
#include<stdlib.h>
#define N 20

typedef struct{
      int t[N];
      int sommet;
      } pile;

void Initialiser_Pile(pile *p) {  p->sommet=-1;}

int pile_Vide(pile p) {  return (p.sommet>0) ;}

int depiler(pile *p) {
   int supp;
   if(pile_Vide(*p))  {printf("la pile  est vide\n"); return -1;}
   else { supp=p->t[p->sommet];  p->sommet--;return supp;}
}

void  empiler(pile *p,int elt) {
       if(!pile_Vide(*p)) printf("la pile  est pleine\n");
       else {  p->sommet++;   p->t[p->sommet]; }
}

int Sommet_pile(pile p) {  return (p.sommet) ;}

void afficher(pile p) {
    int i;
    for(i=0;i<p.sommet;i++)  printf("%d\n",p.t[i]);
}

void deplacer(pile p1,pile *p2) {
 pile *p3;
 int m,s;
          Initialiser_Pile(p3);
          while((!pile_Vide(p1))) {
                    m=depiler(&p1);
                    if(m%2==0) empiler(p3,m); else empiler(p2,m);
          }
          While (!pile_Vide(*p3)) { s=depiler(p3);  empiler(p2,s);  }
}

void copier(pile p1,pile *p2) {
 pile *p3;
 int m,s;
```

```
        Initialiser_Pile(p3);
        While (!pile_Vide(p1)) {
                m=depiler(&p1);
                if(m%2==0) empiler(p3,m);
        }
        while(p3->sommet>0){
                 s=depiler(p3);
                 empiler(p2,s);
                 }
}
```

Exercice 3 :

```
#include<stdlib.h>
#define N 20

typedef struct elmt {// Les structures de données à utiliser ;
      int p[N];
      int sommet;
      } pile;

 typedef struct {
      int F[N];
      int ar;
 } File;

File f1, f2, f3;  // Les variables globales ;

void InitialiserFile(File *F)  { // fonction initialiser une File ;
        F->ar=-1;
 }

int fileVide(File F) {  // fonction file est Vide ? ;
     return (F.ar==-1);
}

void enfiler(File *F, int val)  { // fonction enfiler ;
if (F->ar==N-1) printf("la file est pleine \n");
else { F->ar++; F->F[F->ar]=val; }
}

int defiler(File *Fi)   { // fonction defiler ;
 int x, t;
 int supp;
 if (fileVide(*Fi)) { printf("la file est vide \n"); return 0; }
```

```
 else {   supp=Fi->F[0];
           for (t=1; t<=Fi->ar;t++) Fi->F[t-1]= Fi->F[t] ;
           Fi->ar-- ;
          return supp;
      }
}
void defilerEnfiler(File *F1,File *F2) {
  enfiler(F2, defiler(F1));
}

void afficher(File Fi)  {   // fonction affichage ;
   int  courant=Fi.ar;
   while(courant!=-1) {  printf(" %d  ", Fi.F[courant]); courant --; }
   printf("\n");
}

void trier(File *F1,File *F2) { // fonction Trier ;
  int  m,   t;
 File *F3 ;
 InitialiserFile(F3)
  if (fileVide(*F1))  printf("la file est vide \n ") ;
  else
                while (!fileVide(*F1)) {
                    defilerEnfiler(F3, F1);
                    if (!fileVide(*F1)) {
                                    t=0;  m=F1->ar;
                                    while(t<=m) {
                                              if (comparer(F1, F3)==1) enfiler(F1,defiler(F1));
                                              else {enfiler(F1,defiler(F3)); enfiler(F3,defiler(F1));}
                                                   t++;
                                    }
                                   enfiler(F2, defiler(F3));
                }
           }
}

main(){              // fonction main ;
    int n;
   InitialiserFile(&f1);
   InitialiserFile(&f2);
   printf("entrer les elements de la file et taper un nombre négative pour terminer :\n");
   while(1){
           scanf("%d",&n);
           if(n<0) break; else enfiler(&f1,n);
```

```
   }
  printf("\n**********la file 'avant' le tri est : **********\n");
  afficher(f1) ;
  trier(&f1,&f2);
  printf("\n**********la file 'après' le tri est : **********\n");
  afficher(f2) ;
  system("pause");
}
```