Université Sultan Moulay Slimane **Année Universitaire 2019-2020**
Faculté des Sciences et Techniques
Béni Mellal
Département d'Informatique

Correction TD1 MIPC
Structures de Données en Langage C

Exercice 1 :

```
# include<stdio.h>
# include<string.h>
# define Max 50

typedef char element;
typedef element Pile[Max];
/* les variables globales*/
Pile p;
int Sommet;

initialisation() {Sommet=-1;}

int Pile_vide() { return Sommet <0;}

char pile_pop(void) {
element x;
if (Sommet<0) { printf(" Stack Underflow \n"); return ' '; }
else { x=p[Sommet] ; Sommet=Sommet-1; return x; }
}

int pile_push(element x) {
if (Sommet>=Max-1) {
        printf(" Stack overflow : dépassement de la capacité \n"); return 0; }
else { Sommet=Sommet+1; p[Sommet]=x; return 1;}
}

main() {
char expression[40] ;
char symbole,S,re;
int lon,i ;
printf("Entrer un mot "); gets(expression);
lon=strlen(expression);
initialisation();
if ((lon%2==1) && (expression[lon/2]=='c')) {
        i=0;
        while (i<(lon/2) {
                symbole=expression[i];
                if ((symbole=='a')||(symbole=='b')) pile_push(symbole);
                else break;
                i++;
        }
```

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

Exercice 2 :

```
#include<stdio.h>
#include<string.h>
#define Max 20
         typedef struct {
                        int Sommet;
                        int Tab[Max];
         } Pile;

   void initialiser(Pile *p) { p->Sommet=-1; }

   int Pile_vide(Pile p) { return (p.Sommet<0);}

   int Pile_pleine(Pile p) { return (p.Sommet>=Max-1);}

   int empiler(Pile *p,int x) {
   if (Pile_pleine(*p)) { printf(" Stack overflow : dépassement de la capacité"); return -1;}
   else { p->Sommet++; p->Tab[p->Sommet]=x; return 0;}
   }

   int depiler(Pile *p) {
        int x;
        if (Pile_vide(*p))  {printf(" Stack Underflow "); return -1;}
        else { x=p->Tab[p->Sommet];  p->Sommet--; return x;}
  }

  afficher(Pile p) {
     while ( ! Pile_vide(p))  printf("%d", depiler(&p));
     printf("\n Fin éléments Pile \n ");
  }

  lire (char tab[Max]) {
       printf("Donner un très grand nombre : ");  gets(tab);
  }
```

```
 Chargement (char tab[Max], Pile *p) {
            int i, l=strlen(tab);
            for(i=0; i<=l-1; i++) empiler(p,(tab[i]-48));
 }

Somme(Pile p1, Pile p2, Pile *S){
            int l1, l2, l, i, ret=0, som;
            l1=p1.Sommet; l2=p2.Sommet;  // je suppose que l1<=l2
            for(i=0;i<=l1;i++) {
                        som=depiler(&p1)+depiler(&p2)+ret;
                        ret=som/10;som=som%10;
                        empiler(S,som);
            }
            for(i=l1+1;i<=l2;i++) {
                         som=depiler(&p2)+ret;
                         ret=som/10;som=som%10;
                         empiler(S,som);
            }
            if (ret != 0)       empiler(S,ret);
}

main() {
   Pile p1,p2,Som;
   char tab1[Max],tab2[Max];
   int l1, l2;
   lire(tab1);  lire(tab2);
   initialiser(&p1); initialiser(&p2); initialiser(&Som);
   Chargement(tab1,&p1); Chargement(tab2,&p2);
   l1=strlen(tab1);  l2=strlen(tab2);
   if (l1<=l2) Somme(p1,p2,&Som); else  Somme(p2,p1,&Som);
   printf(" la somme est  :: \n");
   afficher(Som);
   printf(" ********************************** \n");
   system("pause");
}
```

Exercice 3 :

```
#include<stdio.h>
#include<string.h>
#define Max 20

          typedef struct {
                  int Sommet;
                  int Tab[Max];
          } Pile;

         typedef int File[Max];
         int ar, av;
         File F;
```

```
   void initialiser_pile(Pile *p) { p->Sommet=-1; }

   int pile_vide(Pile p) { return (p.Sommet<0);}

   int pile_pleine(Pile p) { return (p.Sommet>=Max-1);}

   int pile_push(Pile *p,int x) {
   if (pile_pleine(*p)) { printf(" Stack overflow : dépassement de la capacité"); return -1;}
   else { p->Sommet++; p->Tab[p->Sommet]=x; return 0;}
   }

   int pile_pop(Pile *p) {
        int x;
         if (pile_vide(*p))  {printf(" Stack Underflow "); return -1;}
         else { x=p->Tab[p->Sommet];  p->Sommet--; return x;}
  }

  afficher(Pile p) {
    while (!pile_vide(p))  printf("%d   ", pile_pop(&p));
    printf("\n Fin elements de la Pile \n ");
 }

// Implémentation à l'aide d'un tableau circulaire

void initialiser_file() { ar=0; av=0;}

int file_pleine() { return av==(ar+1)%Max;}

int file_vide() { return av==ar;}

void file_push(int donnee) {
if (file_pleine()) printf("la file est pleine \n");
else {ar=(ar+1)%Max; F[ar]=donnee;}
}

int file_pop () {
int t;
if (file_vide()) {printf("la file est vide \n"); return -1;}
else {av=(av+1)%Max; t=F[av]; return t;}
}

main(){
Pile P1, P2;
int i,x,l;
initialiser_pile(&P1);  initialiser_pile(&P2);  initialiser_file();
for(i=1; i<=10; i++) pile_push(&P1,i);

while (!pile_vide(P1)) {
          x=pile_pop(&P1);
          if (x%2==0) file_push(x); else pile_push(&P2,x);
}
```

```
while (!pile_vide(P2)) pile_push(&P1,pile_pop(&P2));

l=0;
while (!file_vide()) {l++; pile_push(&P1,file_pop());}
for(i=l; i>=1; i--) pile_push(&P2,pile_pop(&P1));

printf("\n  Pile 1  Nbres impairs \n "); afficher(P1);
printf("\n  Pile 2  Nbres pairs \n "); afficher(P2);

system("pause");
}
```