CHAPITRE 4 : LES PILES ET LES FILES

I. Introduction :

On a décrit une structure où on peut enlever n'importe quel élément. Cependant le coût de cette opération est linéaire en la taille de la structure. Souvent on ne veut pas enlever n'importe quel élément. Si on veut toujours enlever l'élément le plus ancien (c'est à dire le premier qui a été inséré dans la liste), on parlera d'une structure **FIFO** ("**First In First Out**").

Si on veut toujours enlever l'élément le plus récent (c'est à dire le dernier qui a été inséré dans la liste), on parlera d'une structure **LIFO** ("**Last In First Out**"). Les structures FIFO s'appellent également les **files** ("queue" en anglais). Les structures LIFO s'appellent également les **piles** ("stack" en anglais).

II. Les piles :


Pile de Livres

II-1. Définition :

Une pile est une structure de données particulière qui possède les propriétés suivantes :

- On ajoute un élément à la tête de la liste
- On enlève le dernier élément ajouté LIFO (Last In First Out).

Les opérations d'une pile sont si fréquentes qu'elles méritent des noms spéciaux.

II-2. Les opérations usuelles sur une pile :

Traditionnellement on associe à une pile un test qui regarde si la pile est vide. On a donc les opérations suivantes :

- Empiler() : ajout d'un élément au sommet de la pile.
- Depiler() : enlève un élément de la pile.
- PileVide() retourne vrai si la pile est vide

Empiler :	la Pile est vide:	Dépiler:

PILE* Empiler(PILE *tete , int v)
{
PILE *N;
N=(PILE *)malloc(sizeof(PILE));
N->val=v;
 	
N-> suivant =tete;
tete=N;
return (tete);
}
	int Pilevide(PILE * tete)
	{
	return (tete ==NULL);
	}
		PILE* Depiler(PILE *tete)
		{
		PILE *c;
		if(tete!=NULL)
		{
		c=tet ;
		tete = tete ->suivant;
		free(c) ;
		}
		return (tete);
		}

III. Les files :

III-1. Définition :

Un autre type de données abstrait basé sur le modèle de donnée liste est la ***file***. Il s'agit d'une forme restreinte de liste dans laquelle les éléments sont insérés à une extrémité, ***la queue***, et retirés à une autre extrémité, ***la tête.*** Le terme ***FIFO*** (***First-in-first-out***) est un synonyme de file. Une file d'attente à la caisse d'un magasin illustre bien cette notion de file. Les files sont utilisées en programmation pour gérer des objets qui sont en attente d'un traitement ultérieur. Dans une file, les éléments sont systématiquement ajoutés en queue et supprimés en tête.

III-2. Les opérations usuelles sur une file :

Traditionnellement on associe à une file un test qui regarde si la file est vide. On a donc les opérations suivantes :

- Enfiler( ) : ajout d'un élément à la fin d'une liste.
- Defiler( ) : enlève un élément de la file.
- FileVide( ) retourne vrai si la file est vide

Enfiler :

```
/* Enfiler : Fonction qui insère à la fin d'une file: */
/*  La fonction retourne  l'adresse  de la queue */

FILE* Enfiler(FILE *tete, FILE *queue, int v)
{
 FILE *N;
 N=(FILE *)malloc(sizeof(FILE));
 N->val=v;
  if(queue==NULL)   /* La file est vide*/
  {
        queue=N;
        tete=N;
         N-> suivant =NULL;
  }
else
{       N-> suivant =NULL;
        queue-> suivant =N;
        queue=N;
  }
return (queue);
}
```

la File est vide:

```
int  Filevide(FILE *tete)
{
return (tete ==NULL);

}
```

Défiler:

```
FILE* Defiler(FILE *tete)
{
        FILE  *c= tete;
        if(tete!=NULL)
        {
        c= tete;
        tete = tete ->suivant;
        free(c) ;
        }
return (tete);
}
```