Travaux Pratiques d'algorithmique n°8

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

File

Le but de ce TP est de manipuler une *file*. Cette notion de file doit être vue comme un type abstrait. Ainsi, toute opération sur un objet de type file ne doit pas laisser apparaître la structure sous-jacente.
Pour illustrer ceci, nous allons créer deux bibliothèques de fonctions implémentants les opérations de base sur les files.
Nous utiliserons ces bibliothèques pour implémenter une file d'attente à un péage d'autoroute.
Le tp se fera sur deux séances et devra être envoyé impérativement à la fin de la deuxième séance.

1 Représentation par un tableau

Dans cette première section, la file est codée par la structure suivante :

```
struct file
{
  int Tab[MAXFILE];
  int Tete;
  int Queue;
};
typedef struct file File;
```

▸ **Exercice 1.** *Écrire les fonctions :*

- *`void InitialiserFile(File *f)`,*
- *`int EstVideFile(File f)`,*
- *`int EnfilerFile(File *f, int val)`,*
- *`int DefilerFile(File *f, int *val)`,*
- *`int QueueFile(File f, int *val)`,*
- *`void ViderFile(File *f)`,*

permettant respectivement, d'initialiser une file, de tester si une file est vide, d'enfiler un entier sur une file, de défiler un entier d'une file, de retourner l'entier situé en queue d'une file, et de vider une file.
Certaines fonctions retournent 0 en cas de problème et 1 sinon.

▸ **Exercice 2.** ***(Affichage)*** *Écrire une fonction* `void AfficherFile(File f)` *permettant d'afficher le contenu de la file* `f`.

2 Application

▸ **Exercice 3.** ***(File d'attente)*** *Écrire une fonction* `void FiledAtente(void)` *qui implémente une file d'attente de voitures (représentées par des nombres compris entre 0 et 999) à un guichet d'autoroute. On vous demande d'implémenter les fonctionnalités de base :*

- *enfiler une voiture (*`e` *puis un nombre) ;*
- *defiler la voiture se trouvant au guichetr* `d`*) ;*
- *afficher la voiture se trouvant au guichet (*`u`*) ;*
- *afficher le contenu de la file (*`p`*) ;*
- *quitter (*`q`*).*

3 Représentation par une liste chaînée

Dans cette section, la file est codée par une structure chaînée :

```
struct cell_struct
{
  int                  Entier;
  struct cell_struct * Suiv;
};
typedef struct cell_struct Cellule;

struct file_struct
{
  Cellule * Tete;
  Cellule * Queue;
};
typedef  struct file_struct File;
```

▸ **Exercice 4.** *L'objectif de cet exercice est de réécrire toutes les fonctions de manipulation de la file des exercices 1 et 2.*

4 Application (suite)

▸ **Exercice 5.** *Si vous avez tenu compte des spécifications demandées pour la programmation de toutes ces fonctions, la file d'attente que vous avez conçue (cf. exercice 3) doit fonctionner sans aucune modification à l'aide de cette nouvelle représentation par liste chaînée. Tester si c'est bien le cas...*