

## TP1 Structures de données en langage C

### Parcours MIPC

#### Exercice 1 : *pile et élément avec priorité*

On suppose que tout élément est muni d'une **priorité strictement positive** représentée par un **entier** qui doit être précisé au moment de l'ajout de l'élément dans la **pile**. La fonction **retirer** doit **chercher l'élément le plus prioritaire**. On s'intéresse à une implémentation efficace des opérations (**ajouter** et **retirer**).

Chaque élément de la pile sera défini par la structure de donnée suivante :

```
typedef struct elt {  
    int  priorite ;    // priorite représente la priorité de l'élément de type entier  
    char donnee ;     // donnée porte une valeur de type char  
} element
```

- 1- Donner les structures de données et les variables globales à déclarer pour manipuler la pile.
- 2- Donner une implémentation des fonctions **ajouter** et **retirer** dans les deux cas suivants :
  - a- la fonction **ajouter** range les éléments dans un ordre quelconque. Ce qui implique que la fonction **retirer** doit parcourir la pile pour extraire les éléments jusqu'à elle arrive à l'élément prioritaire puis elle doit rendre les éléments retirés dans le même ordre avant l'extraction.
  - b- la fonction **ajouter** maintient les éléments par ordre de priorité croissante.
- 3- Ecrire la fonction **main** pour tester les différentes fonctions.

#### Exercice 2 :

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:

- **defilerEmfiler** : 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**, **defilerEmfiler** et **comparer**.
2. Ecrire la fonction qui permet de trier en utilisant 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.
3. Ecrire la fonction **main** pour tester les différentes fonctions.