Cours de programmation séquentielle

Files d'attente

1 Buts

- Implémentation de la structure de donnée de la file d'attente basée sur un tableau dynamique.
- Implémentation de la structure de donnée de la file d'attente basée sur une liste chaînée.
- Implémentation de fonctions de manipulation de files d'attente.
- Allocation dynamique de mémoire.
- Utilisation de `git` et de `make`.

2 Énoncé

Implémenter deux librairies de files d'attente que vous avez vues en cours d'algorithmique et structures de données:

- La première basée sur une structure de tableau dynamique.
- La seconde basée sur une structure de liste chaînée.

Bien entendu vous devez utiliser `git` et `make` pour la gestion et la compilation de votre projet.

2.1 Type des données dans les files

Le type, `T`, des éléments à stocker dans vos files d'attente seront connus à la compilation mais définis à l'aide d'un `typedef` [1] (un alias pour un type)

```
typedef int32_t T;
```

Ainsi les éléments de votre file d'attente dans le cas où elle est basée sur la liste chaînée seront de type (notez bien que `data` est de type `T`):

```
struct _element {
    T data;
    struct _element *next;
} element;
```

De cette façon vous pouvez facilement changer le type des données contenues dans vos files d'attente. Cela vous sera utile pour un prochain travail pratique.

1. Cela permet d'avoir une certaine "généricité" du code. Pour avoir une *vraie* généricité les éléments de la file d'attente devraient être de type `void*`. Si cela vous intéresse vous pouvez tenter de tout implémenter à l'aide de `void *` mais cela complique beaucoup le travail pratique.

3 Cahier des charges

Vous devez implémenter pour chaque file d'attente toutes les fonctions de manipulation de files que vous avez vues au cours d'algorithmique et structures de données.

Pour chacune des files implémenter un programme qui permet de tester le bon fonctionnement de votre librairie.

3.1 La file basée sur un tableau

Il s'agit ici de créer les fonctions suivantes:

- Créer une nouvelle file vide.
- Libérer le tableau, mettre la capacité à -1.
- Insérer un élément en début de file.
- Extraire un élément en tête de file et le retourner.
- Consulter l'élément en tête de file et le retourner (sans l'extraire).
- Consulter l'élément au début de file et le retourner.
- Tester si la file est vide.
- Compter le nombre d'éléments de la file.

3.2 La file basée sur une liste chaînée

Il s'agit ici de créer les fonctions suivantes:

- Créer une nouvelle file vide.
- Désallouer complètement la file.
- Insérer un élément en début de file.
- Extraire un élément en tête de la file et le retourner.
- Consulter l'élément en tête de file (sans l'extraire).
- Consulter l'élément en début de file.
- Tester si la file est vide.
- Compter le nombre d'éléments de la file.

3.3 Et avec des vecteurs

Adapter la librairie de vecteurs en deux dimensions que vous avez déjà implémentée en physique pour contenir uniquement des entiers (et non des `double`). Écrivez ensuite un programme permettant de tester que votre librairie de files d'attente fonctionne correctement avec ces vecteurs d'entiers.