Université Paris XII
IUT de Fontainebleau
Département informatique
Licence SIL
Structures de données abstraites
2004-2005

PARTIEL 1

Seuls sont autorisés, à titre de documents, les listings comprenant explicitement le nom de l'étudiant imprimé (et celui-ci seulement) et les notes manuscrites (pas de photocopie) comprenant le nom de l'étudiant sur chaque page.

Les deux exercices doivent être présentés dans l'ordre (prévoir deux pages par exercice).

Il est conseillé d'écrire les exercices en langage C++. L'écriture en Java complique les entrées-sorties et l'écriture en langage C allonge le programme (ce qui ne rapportera pas de point supplémentaire dans un cas comme dans l'autre).

Exercice 1.- (**Deque**)

Une **deque** *(pour l'anglais* Double-Ended QUEue*) est une structure de données dynamique linéaire pour laquelle on peut ajouter et retirer un élément à l'un ou l'autre bout.*

- 1^o) Définir la classe `item` des éléments d'une deque de caractères.

- 2^o) Définir la classe `deque` des deques de caractères, comprenant deux attributs `debut` et `fin` et les méthodes constructeur par défaut, `push_t()` (pour ajouter un élément à la fin), `pop_t()` (pour enlever et récupérer un élément à la fin), `push_b()` (pour ajouter un élément au début), `pop_b()` (pour enlever et récupérer un élément au début), `isEmpty()` (pour savoir si la deque est vide, et donc qu'on ne peut pas récupérer d'élément).

[*À implémenter comme liste doublement chaînée pour pouvoir faire référence aussi bien à l'élément précédent qu'à l'élément suivant.*]

- 3^o) Écrire un programme qui demande un certain nombre de lettres, la valeur sentinelle étant le chiffre '0' et qui affiche d'abord les lettres minuscules dans l'ordre inverse dans lequel elles ont été introduites puis les lettres majuscules dans l'ordre dans lequel elles ont été introduites.

Exercice 2.- (**File d'attente**)

Une **file d'attente** *est une structure de données dynamique qui stocke les données et les traite dans l'ordre d'arrivée (premier arrivé, premier servi). Seules deux opérations sont permises pour accéder à celle-ci :* **placer()** *ajoute un item en queue de la file d'attente et* **enlever()** *retire un item en tête de la file d'attente. Bien entendu* **vide()** *permet de tester si la file d'attente est vide.*

- 1^o) Définir la classe **item** des éléments d'une file d'attente d'entiers naturels.

- 2^o) Définir la classe **queue** des files d'attente d'entiers naturels.

- 3^o) Écrire un programme qui permet de saisir un certain nombre d'entiers naturels, la valeur sentinelle étant -1, puis un entier naturel n. Le programme affichera alors les entiers saisis depuis le n-ième jusqu'à la fin.

Les entiers saisis seront sauvegardés dans une file d'attente.