Département d'Informatique
DUT Informatique S2
Année 2011/2012
Algorithmique et programmation

Projet d'algorithmique et programmation n° 2

« Commande `sort` »

mai 2012

1 Objectif

L'objectif du projet est d'écrire un programme qui, de manière similaire à la commande standard `sort(1)`, trier des lignes de texte. Le programme devra :

(i) lire le texte sur son entrée standard `std::cin` ;

(ii) trier les lignes suivant l'ordre lexicographique ;

(iii) écrire les lignes triées sur sa sortie standard `std::cout`.

2 Contraintes

Le programme devra fonctionner pour un nombre arbitraire de lignes en entrée, dans la limite de la mémoire disponible. Si le programme doit avoir une limite sur la longueur maximale des lignes, celle-ci devra être au moins de 2048 caractères.

Le programme devra se comporter correctement dans les cas suivants (liste non exhaustive) :

- texte déjà trié ;
- texte composé de une ligne, deux lignes, ou plus ;
- texte comportant des lignes égales ;
- texte comportant des lignes préfixes d'autres lignes ;
- texte comportant des lignes vides ;
- cas particulier où la dernière ligne n'est pas terminée par un caractère `'\n'` ;
- texte composé d'un grand nombre de lignes ;
- (optionnel) texte comportant des lignes très longues.

3 Détails d'implémentation

Structure de données. Afin de pouvoir manipuler un nombre de lignes arbitrairement grand, les lignes devront être stockées dans une *liste chaînée*. Chaque maillon de la liste devra correspondre à une ligne de texte.

Chaînes de caractères. Le choix du type de chaînes de caractères utilisé est laissé libre, entre :

- les chaînes *à la C* (type `char*`) ; ou
- les chaînes *à la C++* (classe `std::string`).

Dans le premier cas, on pourra utiliser la fonction standard `strcmp(3)` pour effectuer la comparaison de chaînes de caractères. Il faut alors ajouter `#include <cstring>` au début du programme.

Algorithme de tri. L'algorithme de tri qui devra être utilisé est l'algorithme appelé *tri rapide* (*quicksort* en anglais). L'idée générale de cet algorithme récursif, pour trier une liste, est la suivante :

1. Choisir un élément de la liste : on appelle cet élément le *pivot*. L'enlever de la liste.
2. Partitionner la liste en deux sous-listes :
   - $liste_1$: contenant tous les éléments de la liste inférieurs au pivot ;
   - $liste_2$: contenant tous les éléments de la liste supérieurs ou égaux au pivot.
3. Trier récursivement les deux sous-listes $liste_1$ et $liste_2$.

4. Le résultat final est donné par la concaténation de
   - la sous-liste $liste_1$ triée ;
   - le pivot ;
   - la sous-liste $liste_2$ triée.

On remarquera que toutes ces manipulations peuvent se faire simplement en réagençant les maillons composant la liste initiale. À aucun moment il n'est nécéssaire de les dupliquer.

Libération de la mémoire. Toute la mémoire allouée dynamiquement devra être explicitement libérée avant de terminer le programme. Des outils comme `valgrind` peuvent être utilisés pour le vérifier.

4 Tests

Pour vérifier le résultat de votre programme, vous pourrez comparer sa sortie avec le résultat de la commande `sort(1)`. Les utilitaires `cmp(1)` et `diff(1)` permettent de faire une telle comparaison. Par exemple, si `entree` est un fichier contenant du texte, et `monsort` est le nom de votre programme :

```
$ ./monsort < entree > sortie_monsort
$ sort < entree > sortie_sort
$ cmp sortie_monsort sortie_sort
```

Par défaut, la commande `cmp` affiche un message si les deux fichiers (`sortie_monsort` et `sortie_sort` dans l'exemple) sont différents. Elle n'affiche rien si les fichiers sont identiques.

5 Contraintes diverses

Date importante. Le programme devra être rendu pour le **mercredi 6 juin 2012, 12h00** au plus tard.

Combien de personnes par projet ? Les projets sont à faire *individuellement*.

Quel sera le poids du projet dans la moyenne finale ? La note du projet ne comptera que si elle ne fait pas baisser la moyenne.

Comment rendre le projet ? Le projet devra être envoyé par mail à l'adresse suivante :
Arnaud Giersch <arnaud.giersch@iut-bm.univ-fcomte.fr>.
Un accusé de réception vous sera retourné. Si vous n'avez pas reçu d'accusé de réception le mercredi 6 juin à 18h00, il faudra renvoyer votre projet.

Que faut-il rendre ? Il est demandé de rendre deux fichiers : le code source du programme, et un rapport au format **PDF** (un rapport rendu dans tout autre format sera considéré comme nul).

Sous quelle forme doit se présenter le programme ? Le programme devra être composé d'un seul fichier source C++. Ce fichier devra pouvoir être compilé sans erreur. Il est *impératif* de mettre, en début de fichier, un commentaire rappelant le nom, le prénom et le groupe de l'étudiant, ainsi que le titre du projet.

Que faut-il mettre dans le rapport ? Le rapport devra contenir toutes les informations nécessaires à une bonne compréhension du travail rendu. Il devra également donner de manière claire les limites de votre programme (bugs, longueur maximale des lignes, etc.).

Quelles contraintes pour le code source ? Le code source devra être correctement indenté. Les lignes ne devront pas dépasser 80 caractères. Il sera porté une attention particulière à l'usage pertinent des commentaires, à l'usage pertinent des fonctions, au choix des noms de variables et de fonctions, et à la lisibilité du code en général.