Travail Pratique 1 : fréquencier

INF3105, Hiver 2015

But du travail pratique

Le but de ce travail pratique est de vous permettre de pratiquer les notions de C++ vues dans les deux premières séries d'acétates. Il consiste en la réalisation d'un fréquencier, programme qui affiche le nombre d'occurrences de chaque mot d'un texte. Ce type de programme est notamment utilisé en linguistique de corpus. Votre travail devra impérativement respecter les consignes spécifiques données dans cet énoncé.

Description

Vous devez écrire un programme, dont le fichier source se nommera **`frequencier.cpp`**, permettant d'afficher la fréquence des mots d'un texte quelconque. Le texte à analyser est contenu dans un fichier. Le programme lit ce dernier et affiche la liste des couples (mot,fréquence) sur la sortie standard. Il ne doit en aucun cas fixer de limites a priori quant à la taille des lignes à traiter, quant à leur nombre, ou quant à la taille des mots à traiter ou quant à leur nombre.

Exemple : supposons que le texte à analyser est contenu dans le fichier *toto.txt*, et que l'exécutable s'appelle `frequencier` (Attention : ne supposez pas que ce sera nécessairement le nom de l'exécutable). L'appel à `frequencier` se fera comme suit :

```
frequencier toto.txt
```

Tout autre appel à l'exécutable doit faire l'objet de messages d'erreur appropriés.

Un mot est une suite quelconque de lettres, majuscules ou minuscules, **non-accentuées**, et de chiffres. Tout autre caractère sera considéré comme séparateur de mots, et ne sera pas comptabilisé par votre application. Le résultat sera affiché par fréquence décroissante et par ordre alphabétique croissant de mot (selon le code ASCII), à raison d'un mot par ligne : la fréquence est affichée en premier, sur 10 colonnes, cadrée à droite, suivie d'un espace, suivie de l'affichage du mot cadré à gauche.

Des exemples d'exécution figurent sur la page Moodle du cours.Vous remarquerez entre autres que majuscules et minuscules sont considérées comme des lettres distinctes.

Démarche à suivre (impérativement)

Vous devrez utiliser une structure appropriée pour représenter un mot donné (i.e. la suite de caractères composant le mot ainsi que la fréquence associée). Vous utiliserez ensuite un vecteur pour représenter les mots contenus dans le texte à analyser et leur fréquence courante.

Une fois le texte à traiter parcouru, vous devrez procéder au tri des mots pour l'affichage. Vous devrez implémenter les tris nécessaires, mais vous veillerez à ne pas implémenter

de tris inutiles en revanche. L'algorithme de tri à utiliser est celui du **tri par insertion**, dont le pseudo-code suit :

```
// La numérotation du tableau à trier commence à 0
// Le tableau comporte N éléments
Pour i = 1 à N-1
     // Les éléments de 0 à i-1 sont ordonnés
     j = i;
     Tant que j > 0 et a[j] < a[j-1]
          echanger a[j] et a[j-1];
          j--;
```

Pour ce TP, vous ne devez pas recourir à d'autres structures de la STL que les vecteurs et les `string`, et vous ne devez pas non plus recourir aux algorithmes fournis par la STL. Lorsque cela est possible, vous devez privilégier les `string` aux chaînes de caractères de type C, et vous devez également privilégier les `vector` aux tableaux de type C. Vous ne créerez pas non plus de classes.

Remise

Le travail doit être réalisé seul ou en équipe d'au plus deux étudiants. N'oubliez pas de spécifier le nom des coéquipiers dans le code source et en page de garde de votre dossier papier.

La remise électronique doit être faite au plus tard le **23 février 2015 à 17h40 via Moodle**. Moodle bloque les remises électroniques après cette heure-là. Avant de procéder à la remise, veuillez renommer votre fichier :

codeEtudiant`_frequencier.cpp`

si vous travaillez seul, ou :

codeEtudiant1_codeEtudiant2`_frequencier.cpp`

si vous travaillez à 2. **Ce qui apparaît en gros caractères dans les noms précédents doit être remplacé par les codes appropriés**. Vous utiliserez Moodle pour la livraison de votre fichier renommé. Au cas où vous effectueriez plusieurs livraisons (avant la date limite), Moodle ne conserve que la dernière d'entre elles. Le code source remis doit pouvoir être compilé et exécuté SANS MODIFICATION sous Windows à l'aide de la version de *g*++ fournie avec CodeBlocks.

Le dossier papier doit être remis en main propre le **23 février 2015 à 18h00 (au début du cours)**. Il consiste en une impression du code source **précédée d'une page de garde**. **Votre dossier doit être broché**, mais il est inutile d'utiliser enveloppes ou pochettes. Si vous ne pouvez être présent au début du cours, vous devrez avoir remis votre dossier papier via la chute à travaux du sécrétariat du Département d'informatique **au plus tard à 16h00 le jour de la remise**. Dans ce cas, vous devez remettre votre travail dans une enveloppe mentionnant mon nom et le sigle du groupe-cours.

Les retards ne sont pas acceptés : c'est la note zéro qui sera attribuée à un travail non remis à temps.

Barème

Votre travail sera évalué selon le barème suivant :

Fonctionnalités	Le programme compile et fonctionne correctement. Ces points sont attribués par les tests de votre programme sur différents cas représentatifs, incluant la validation de la ligne de commande, les tests du programme sur des cas limites non problématiques (fichier vide, petit fichier …), et les cas problématiques lorsque applicable. Veuillez utiliser les options -Wextra et -Wall du compilateur pour corriger la présence d'éventuels avertissements du compilateur.	50 pts
Structure et lisibilité	Utilisation de structures de données appropriées. Test de l'ouverture des fichiers. Fermeture des fichiers ouverts précédemment dans le code.	10 pts
	Utilisation d'itérateurs dans au moins une partie du traitement des `vector`	10 pts
	Utilisation du mécanisme des exceptions et du canal d'erreur	10 pts
	Utilisation de structures de contrôles appropriées et bon découpage en fonctions. Passage de grosses structures de données par référence et non par valeur. Utilisation d'arguments qualifiés par `const` lorsque cela s'applique.	10 pts
	Clarté du code. En particulier : • indentation du code et présence de commentaires pertinents • utilisation de noms d'identificateurs significatifs et respectant les conventions énoncées en cours • le fait de ne pas inclure de bibliothèques standard non requises	10 pts
Total		