TP : Gestion de fichier

1. Les fichiers en C++

Généralité sur les flots

Les entrées et sorties en C++ se font toujours par l'intermédiaires de **flots** (ou flux), qu'on peut considérer comme des canaux
- recevant de l'information dans le cas d'un flot de sortie
- fournissant l'information dans le cas d'un flot d'entrée

Le flot de sortie standard est **cout**. Ce flot est connecté par défaut à l'écran.
Le flot d'entrée standard est **cin**. Ce flot est connecté par défaut au clavier.

Les opérateurs de manipulation des flots, qui permettent le transfert de l'information sont
- << qui permet l'écriture sur un flot de sortie
- >> qui permet la lecture sur un flot d'entrée

Ouverture et fermeture d'un fichier

Pour utiliser les fichiers en C++, il est nécessaire d'inclure le fichier d'en-tête **<fstream>.**
Ce fichier inclut lui-même <iostream>: donc lorsqu'on inclut fstream.h, il ne faut pas inclure iostream.h

Ouvrir en lecture

ifstream *nom_fichier_logique* ("*chemin_fichier_physique*");

entre parenthèses et entre guillemets, on indique le chemin physique du fichier, c'est-à-dire :
- le nom du fichier
- suivi de son extension
- et, si le fichier n'est pas contenu dans le même répertoire que l'exécutable, son chemin absolu complet (ex:"c:\travail\essai.txt")

Remarque : si le fichier est dans le même répertoire que le programme, seul le nom de ce fichier est utile.

ex: ifstream test ("essai.txt");

Ouvrir en écriture : création d'un nouveau fichier ou écriture par dessus le fichier s'il existe déjà

ofstream *nom_fichier_logique* ("*chemin_fichier_physique*");

ex: ofstream test ("essai.txt")

Ouvrir en mode ajout

ofstream <nom_fichier_logique> ("chemin_fichier_physique", **ios::in|ios::app**);

ex: ofstream test ("essai.txt", ios::in, ios::app);

Ouvrir en mise à jour

La mise à jour (modifications ou suppression d'une donnée) d'un fichier en C++ est très délicate à effectuer directement. Le plus simple est de charger tout le fichier en mémoire (mode lecture), dans un tableau d'enregistrements par exemple, de faire les modifications en mémoire, puis de réécrire toutes les données modifiées à la place de l'ancien fichier (mode écriture).

Fermeture d'un fichier

nom_fichier_logique.**close( );**
ex: test.close( );

Manipulation d'un fichier

Test de la fin d'un fichier
nom_fichier_logique.**eof( )**

lecture à partir d'un fichier

nom_fichier_logique >> *variable1* [>> *variable2*>> …];
ex: test >>nom>> prenom;

Signification : on lit à partir du fichier appelé test, les variables nom et prenom

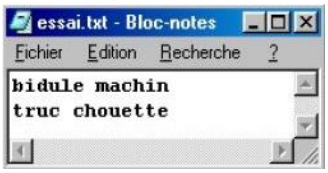
Rappelons que l'espace et le saut de ligne sont des séparateurs de lecture.

écriture sur un fichier

nom_fichier_logique << *expression1* [<<*expression2*<< …];

Il faut écrire soit même les différents séparateurs
(espaces ou sauts de ligne, ou tout autre caractère de séparation)

test<< "bidule"<<" "<<"machin"<<"\n"<<"truc"<<" "<<"chouette";
après cette instruction, le fichier nommé "essai.txt" va se présenter ainsi



Exercice 1

Ecrire 2 programmes (pas de classe personne):
- l'un qui permet de saisir et mémoriser dans un fichier nommé répertoire, le nom et le numéro de téléphone de n personnes
- l'autre qui permet d'afficher à l'écran le contenu du fichier répertoire, en sautant une ligne entre chaque personne

Exercice 2

Cet exercice va permettre la création et la consultation d'un ou plusieurs fichiers sur les matchs de la coupe du monde de football du groupe de la France uniquement (France, Angleterre, Suisse et Croatie).
{3,1}, // Suisse - Croatie
{2,0}, // France - Angleterre
{3,0}, // Suisse - Angleterre
{2,1}, // France - Croatie
{0,1}, // Angleterre - Croatie
{3,2} // Suisse - France

1. Il s'agira de créer deux fonctions :
- l'une permettant de saisir et d'enregistrer dans un fichier les informations concernant le résultat des matchs du groupe de la France
- l'autre permettant de consulter le contenu du fichier crée par la méthode précédente.

2. Prendre le source en ligne et le mettre sous forme de classe en incluant les 2 fonctions implémentées précédemment