Développement : Les fichiers

© tv <tvaira@free.fr> v.1.0

Ce sujet comprend 7 questions pour un total de 20 points.

A Questions de cours

Question 1 (3 points)
Quels sont les trois flux standards de tout programme ?

Question 2 (4 points)
Citer les quatre appels systèmes de l'API POSIX pour manipuler un fichier ?

Question 3 (4 points)
Citer les quatre fonctions de l'API Win32 pour manipuler un fichier ?

Question 4 (2 points)
À quel moment est réalisé l'association entre nom logique et nom physique ?

Question 5 (2 points)
À quel moment est vérifié les droits d'accès au fichier ?

Question 6 (2 points)
Qu'est-ce qu'un descripteur de fichier ?

B Exercices

On désire exporter des résultats vers un tableur par l'intermédiaire d'un fichier texte (au format CSV).

Le format CSV (*Comma-Separated Values*) est un format informatique ouvert représentant des données tabulaires sous forme de valeurs séparées par un délimiteur (initialement des virgules). Les séparateurs ne sont pas standardisés (virgules, points-virgules, etc...) et cela rend ce format peu pratique pour une utilisation autre que des échanges de données ponctuels. Il est toutefois très populaire parce qu'il est très facile à générer. Les fichiers CSV sont des fichiers texte essentiellement utilisés par des logiciels de type tableur (Microsoft Excel, LibreOffice/OpenOffice calc, ...). Chaque ligne du texte correspond à une ligne du tableau et les virgules correspondent aux séparations entre les colonnes. Les portions de texte séparées par une virgule correspondent ainsi aux contenus des cellules du tableau.

Le format d'enregistrement sera ici :
– retour à la ligne (`\n`) entre chaque enregistrement (mesure ;date ;heure)
– les trois valeurs (mesure ;date ;heure) sont séparés par des points-virgules (';').

Les données à exporter sont accessibles à partir des tableaux suivants :

```
#include <iostream>
#include <fstream>
using namespace std;

int main ()
{
    const int nbMesures = 6;
    double pt100[nbMesures] = {35.23, 35.10, 34.45, 35.02, 35.50, 35.53 };
    string date = "12-01-2016";
    string heure[nbMesures] = {"11:00", "11:05", "11:10", "11:15", "11:20", "
        11:25" };

    //...

    return 0;
}
```

En C++, on utilisera la classe `ofstream` pour écrire dans un fichier :
Exemple :

```
#include <fstream> // for std::ofstream

int main ()
{
  std::ofstream ofs("test.txt", std::ofstream::out);

  ofs << "hello\n" ; // write string

  int n = 5;
  ofs << n << std::endl; // write int
```

public member function

std::ofstream::ofstream

<fstream>

C++98 C++11

```
default (1)        ofstream();
initialization (2) explicit ofstream (const char* filename, ios_base::openmode mode = ios_base::out);
```

Construct object

Constructs an ofstream object:

(1) default constructor

Constructs an ofstream object that is not associated with any file.
Internally, its ostream base constructor is passed a pointer to a newly constructed filebuf object (the *internal file stream buffer*).

(2) initialization constructor

Constructs an ofstream object, initially associated with the file identified by its first argument (*filename*), open with the mode specified by *mode*.
Internally, its ostream base constructor is passed a pointer to a newly constructed filebuf object (the *internal file stream buffer*). Then, filebuf::open is called with *filename* and *mode* as arguments.
If the file cannot be opened, the stream's failbit flag is set.

(3) copy constructor (deleted)

Deleted (no copy constructor).

(4) move constructor

Acquires the contents of *x*.
First, the function move-constructs both its base ostream class from *x* and a filebuf object from *x*'s internal filebuf object, and then associates them by calling member set_rdbuf.
x is left in an unspecified but valid state.

The internal filebuf object has at least the same duration as the ofstream object.

Parameters

filename

A string representing the name of the file to be opened.
Specifics about its format and validity depend on the library implementation and running environment.

mode

Flags describing the requested input/output mode for the file.
This is an object of the bitmask member type openmode that consists of a combination of the following member constants:

member constant	stands for	access
in	**in**put	File open for reading: the *internal stream buffer* supports input operations.
out *	**out**put	File open for writing: the *internal stream buffer* supports output operations.
binary	**binary**	Operations are performed in binary mode rather than text.
ate	**at e**nd	The *output position* starts at the end of the file.
app	**app**end	All output operations happen at the end of the file, appending to its existing contents.
trunc	**trunc**ate	Any contents that existed in the file before it is open are discarded.

These flags can be combined with the bitwise OR operator (|).
* out is always set for ofstream objects (even if explicitly not set in argument *mode*).
Note that even though ofstream is an output stream, its internal filebuf object may be set to also support input operations.

```
  ofs.close();

  return 0;
}
```

Question 7 (3 points)
Ecrire le code C++ qui exporte les valeurs du tableau pt100 (capteur de températures) dans le fichier texte "export.txt". Celui-ci sera créé dans le répertoire courant.