CM – Programmation C / C++

Chapitre 2 : Fichiers en C++

I- Définition d'un fichier :

Un fichier est un ensemble de données stockées sur un support. Dans le cadre d'un ordinateur, c'est généralement dans son disque dur. Un fichier a un nom, terminé par un point et une extension de fichier, comme par exemple : `coursc++ch5.docx`, `wordpad.exe`, `fichier.txt`, `tp.h`, `main.cpp`, etc. Les fichiers sont regroupés en répertoires, eux-mêmes pouvant être regroupés en d'autres répertoires et formant un chemin d'accès au fichier (par exemple : `C:\Users\cazes\Documents\Enseignement\CPP`).

En C++, un fichier sera représenté par un « objet » fichier qui est une variable de type **fstream**.

II- Ouverture d'un fichier :

1) *Principe d'ouverture d'un fichier :*

Voici un exemple de code permettant d'ouvrir un fichier «Liste.txt» quelconque :

```
#include <iostream>
#include <fstream>

using namespace std;


int main()
{
    fstream f;
    f.open("Liste.txt", ios::out);

    instructions;

    return 0;
}
```

Pour ouvrir un fichier, il faut d'abord inclure la bibliothèque <fstream> dans les directives. On initialise ensuite une variable f de type **fstream** qui sert de nom logique au fichier dans le code. On ouvre ensuite le fichier par la commande f.open et en entrant son nom entre guillemets en argument. Le terme ios::out dans la parenthèse signifie qu'il s'agit d'un fichier de sortie, c'est-à-dire que l'on va écrire à l'intérieur. Pour éviter de mettre ce terme, on peut utiliser le type **ofstream**.

Lorsque l'on ouvre un fichier, s'il n'existe pas, il est créé, et s'il existe déjà, on le remplace, ce qui fait perdre toutes les données du fichier précédent.

2) *Ecriture et lecture dans un fichier :*

Pour écrire dans un fichier, on doit initialiser une variable et ouvrir le fichier de la manière suivante :

```
fstream f;
f.open("Liste.txt", ios::out);
```

On peut aussi écrire avec le type **ofstream** :

```
ofstream f;
f.open("Liste.txt");
```

Pour lire dans un fichier, on doit faire la même initialisation, mais on ajoute `ios::in` dans la parenthèse :

```
fstream f;
f.open("Liste.txt", ios::in);
```

On peut aussi écrire avec le type **ifstream** :

```
ifstream f;
f.open("Liste.txt");
```

Pour lire et écrire simultanément dans un fichier, on écrit :

```
fstream f;
f.open("Liste.txt", ios::out | ios::in);
```

Il est possible d'ouvrir un fichier en même que l'on déclare la variable de type **fstream** :

```
fstream f("Liste.txt", ios::in);
```

Cette écriture est aussi utilisable avec `ios::out` et `ios::app`. De même, il est aussi possible d'ouvrir un fichier en déclarant la variable avec les types **ifstream** et **ofstream** :

```
ofstream f("Liste.txt");

ifstream f("Liste.txt");
```

3) Classes de flux :

Dans leur fonctionnement, l'écriture et la lecture dans un fichier ressemblent beaucoup à l'affichage à l'écran et à la lecture au clavier. En prenant le cas de lecture et d'écriture d'un fichier quelconque (appelé «toto.txt» ici), on peut comparer respectivement avec l'affichage à l'écran et la lecture au clavier :

```
ofstream f("toto.txt");
f << "bla bla" << endl;
cout << "bla bla" << endl;
```

Dans ce cas, on ouvre le fichier en écriture tout en déclarant la variable puis on écrit le texte à l'intérieur, comme on afficherait un texte à l'écran. La variable `f` de type **ofstream** et **cout** sont tous les deux des objets de type **ostream** : ce sont des flux de sortie (ils font partir les informations contenues dans le programme vers un autre endroit extérieur, un fichier ou l'invité de commandes affiché à l'écran).

```
ifstream f("toto.txt");
string a;
f >> a;
cin >> a;
```

Dans ce cas, on ouvre le fichier en lecture tout en déclarant la variable puis on déclare une variable de type **`string`** dont la variable du fichier va affecter une valeur se trouvant dans le fichier, comme on affecterait une variable en entrant la valeur grâce au clavier. La variable `f` de type **`ifstream`** et **`cin`** sont tous les deux des objets de type **`istream`** : ce sont des flux d'entrée (ils récupèrent des informations externes, venant d'un fichier ou d'une entrée au clavier, au programme pour les lui affecter).

4) Utilisation des opérateurs de flux :

Afin d'écrire dans un fichier, on utilise l'opérateur `<<`, tandis que pour lire dans un fichier, on utilise l'opérateur `>>`.

Exemple – Création d'un fichier et écriture des chiffres de 0 à 5 :

```
#include <iostream>
#include <fstream>

using namespace std;


int main()
{
    fstream f;
    f.open("Liste.txt", ios::out);

    for(int i=0;i<6;i++)
        f << i << endl;

    f.close();

    return 0;
}
```

Le programme a créé un fichier et y écrire les chiffres de 0 à 5 à chaque ligne.

Exemple – Lecture du fichier créé :

```
#include <iostream>
#include <fstream>

using namespace std;


int main()
{
    fstream f;
    f.open("Liste.txt", ios::in);

    int A;
    f >> A;

    f.close();

    return 0;
}
```

Le programme affecte à la variable écrite une valeur entière qu'il lit dans le fichier.

Exemple – Ajout d'une ligne à un fichier existant :

```
#include <iostream>
#include <fstream>

using namespace std;


int main()
{
    fstream f;
    f.open("Liste.txt", ios::out | ios::app);

    f << "J'ajoute une ligne au fichier" << endl;

    f.close();

    return 0;
}
```

Le programme ouvre le fichier `Liste.txt` qui a été créé précédemment et ajoute une phrase supplémentaire dans le fichier.

5) Fonctions utiles du type fstream :

Il existe deux fonctions utiles dans l'utilisation du type **`fstream`** :

- `f.good()` qui renvoie **`false`** si on ne peut plus lire dans le fichier.
- `f.eol()` qui renvoie **`true`** si on est à la fin du fichier géré par la variable.