Programming techniques

Week 10 - Binary File

04/2024

Text file

- To open a text file:

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

int main () {
   ofstream fout;
   fout.open ("FileToWrite.txt");
   // do something on the file
   fout.close();
   return 0;
}
```

Working with files

- In C++, we use the following streams
  - fstream: to open for reading and writing
  - ifstream: default is for reading (ios::in)
  - ofstream: default is for writing (ios::out)
- Syntax:

```
f.open(filename, mode);
```

The modes for file

- Modes:
  - `ios::in` open to read
  - `ios::out` open to write
  - `ios::binary` open a binary file (default: text)
  - `ios::ate` starting from the end of file
  - `ios::app` open to append (write mode only)
  - `ios::trunc` delete the old file and overwrite
- Close file when finishes:

```
f.close();
```

Status of a file

- To check the status of a file
  - **good()**: check if the stream is ready for reading or writing?
  - **bad()**: if the reading or writing was fail, this flag is turned on
  - **fail()**: similar to bad(), but it also checks the format, i.e. we need to read an integer number but facing a character..
  - **eof()**: the file cursor is at the end of the file

The get and put cursor of a stream

- In the **ifstream**, the **get** cursor is at the position of the next element to be read
- In the **ofstream**, the **put** cursor is at the position to write the next element
- Some operations on these 2 cursors:
  - **tellg()**: tell the current position of the **get** cursor
  - **tellp()**: tell the current position of the **put** cursor
  - **seekg(position)**: move the **get** cursor to the **position**
  - **seekp(position)**: move the **put** cursor to the **position**

get/put file cursor

- **seekg(offset, direction)**
- **seekp(offset, direction)**

Move the cursor offset steps based on the direction.

- Direction:
  - **ios::beg**: from the beginning of the stream
  - **ios::cur**: from the current position
  - **ios::end**: from the end of the stream

Binary file

- Open a binary file
  - Similar to a text file but with the mode

    ios::binary
- Read and write to a binary file:
  - **write(mem_block, size)**
  - **read(mem_block, size)**

  mem_block: a pointer to an array of bytes

  size: the number of bytes to read/write

Example

```
int a[]={1, 2, 3, 4};
int n=sizeof(a)/sizeof(a[0]);
ofstream fout;
fout.open("Output.bin", ios::binary);
if (fout.is_open()) {
   fout.write((char*)&n, sizeof(n));
   fout.write((char*)a, sizeof(a));
}
fout.close();
```

Home reading & implementation

- Read and understand the format of a bitmap file
  - Document provided on Moodle
  - Read from different sources from the internet
- With the help from the TAs, students need to implement a program to read and display a BMP file