Lab Sheet 8
Understanding the Concept of Console and File Input/Output

Console Input/Output

Within console we can perform unformatted and formatted input/output. For unformatted input/output stream functions like `put()`, `get()`, `getline()`, `write()`, `read()` etc are used. For formatted input/output the stream objects `cin` and `cout` are used along with `ios` functions and flags and manipulators.

The `ios` functions that can be used for formatting are

- `width()`
- `fill()`
- `precision()`
- `setf()`
- `unsetf()`
- `flags()` etc

However to use the `setf()`, `unsetf()` and `flags()` functions one should know the flags available in `ios` class.

File Handling

There are three classes for handling files.

- `ifstream` - for handling input files
- `ofstream` - for handling output files
- `fstream` - for handling input as well as output files.

In all three classes, passing a filename as the first parameter in the constructor itself can open a file.

e.g `ifstream infile("test.txt")` opens the file `test.txt` in the input mode.

The constructors for all these classes are defined in the header file `<fstream>`, which are as follows

```
ifstream( const char *path, int mode=ios::in)

ofstream( const char *path, int mode=ios::out)

fstream( const char *path, int mode=ios::in|ios::out)
```

where `path` specifies the file to be opened, `mode` specifies the mode in which the file is to be opened.

File opening can also be done explicitly by calling the member function `open()` of the file stream classes. The `open()` function has similar prototype as the constructors.

After opening, the file contents can be written or read by using the stream operators with the file objects as

```
ofstream ofile("test.txt");

ofile<<"C++ lab class";
```

This statement writes "`C++ lab class`" in the file "`test.txt`"

Reading and Writing A class Object

The Binary input and output functions read( ) and write are designed to handle the entire structure of an object as a single unit, using the computer's internal representation of data. The function write copies a class object from memory byte by byte with no conversion.

Binary output and input functions take the following form

```
ipfile.read(reinterpret_cast<char*>(&obj),sizeof(obj));

opfile.write(reinterpret_cast<char*>(&obj),sizeof(obj));
```

Example

```
#include<iostream>
#include<fstream>
#include<iomanip>
using namespace std;
class demofile
{
   private:
     int a;
     int b;
   public:
     demofile(){}
     demofile(int x,int y){a=x;b=y;}
     void display()
     { cout<<"a= "<<a<<endl<<"b= "<<b<<endl;}
};
```

```
int main()
{
   demofile de(10,20);
   clrscr();
   fstream file;
   file.open("demo.txt",ios::in|ios::out);
   file.write(reinterpret_cast<char*>(&de),sizeof(de));
   file.seekg(0);
   file.read(reinterpret_cast<char*>(&de),sizeof(de));
   de.display();
   file.close();
   return 0;
}
```

Exercises

1. Write a program to demonstrate the use of different ios flags and functions to format the output. Create a program to generate the bill invoice of a department store by using different formatting.
2. Write a program to create a userdefined manipulator that will format the output by setting the width, precision and fill character at the same time by passing arguments.

3. Write a program to overload stream operators to read complex number and display the complex number in a+ib format.
4. Write a program that stores the information about students (name, student id,department and address) in a structure and then transfers the information to a file in your directory. Finally, retrieve the information from your file and print in the proper format on your output screen.
5. Write a program for transaction processing that write and read object randomly to and from a random access file so that user can add, update, delete and display the account information (accountnumber, lastname, firstname, totalbalance).