National Chiao Tung University
Department of Electrical and Computer Engineering

Laboratory Manual 03
March 19, 2018

UEE1303 S18: Object-Oriented Programming

Basic Class & File I/O

What you will learn from Lab 3

In this laboratory session you will:

1. Learn how to write a basic class
2. learn the usage of file I/O.

LAB 3-1: BASIC CLASS

✓ We rewrite the structure Point2D, defined in program lab3-1-1, as a class object.

```
// lab3-1-1.cpp
#include <iostream>
using namespace std;
class Point2D
{
    void assignPoint2D(int n1, int n2, double v);
    void displayPoint2D();
    int x;
    int y;
    double value;
};
void Point2D::assignPoint2D(int n1, int n2, double v)
{
    x = n1;
    y = n2;
    value = v;
}
void Point2D::displayPoint2D()
{
    cout << "(" << x << "," << y << ") = ";
    cout << value << endl;
}
int main()
{
    Point2D ptArray[10];
    for (int i = 0; i < 10; i++)
    {
        ptArray[i].assignPoint2D(i, i+2, i*10);
        ptArray[i].displayPoint2D();
    }
    return 0;
}
```

➢ Please fix the compiler error in this example.

➢ If you do not specific the member access modifiers, the compiler will take as private member.

- ✓ We rewrite the above program and modify the class Point2D with member access modifiers.

```
// lab3-1-2.cpp
#include <iostream>
using namespace std;
class Point2D
{
    public:
    void assignPoint2D(int n1, int n2, double v);
    void displayPoint2D();
    private:
    int x;
    int y;
    double value;
};
void Point2D::assignPoint2D(int n1, int n2, double v)
{
    x = n1;
    y = n2;
    value = v;
}
void Point2D::displayPoint2D()
{
    cout << "(" << x << "," << y << ") = ";
    cout << value << endl;
}
int main()
{
    Point2D ptArray[10];
    for (int i = 0; i < 10; i++)
    {
        ptArray[i].assignPoint2D(i, i+2, i*10);
        ptArray[i].displayPoint2D();
    }
    return 0;
}
```

LAB 3-2: MEMBER FUNCTION

- ✓ To access private data members, we can add get (accessor) and set (mutator) functions as public member functions.

```
// lab3-2.cpp
#include <iostream>
using namespace std;
class Point2D
{
public:
    void setCoord(int n1, int n2);
    void setValue(double v);
    int getCoordX();
    int getCoordY();
    double getValue();
private:
    int x;
    int y;
    double value;
};
// Please implement the definitions of five additional member functions.
int main()
{
    Point2D a;
    a.setCoord(1,3);
    cout << "a(x,y) = " << a.getCoordX() << " "    << a.getCoordY() << endl;
    Point2D *b = new Point2D;
    b->setValue(5);
    cout << "value of b is " << b->getValue() << endl;

    return 0;
}
```

LAB 3-3: FILE INPUT STREAM

✓ Program lab3-3 shows an example of using file input stream to read data from the text file lab3-3.txt.

✓ The below shows the context of file lab3-3.txt.

```
1 4.5 file
```

✓ Program lab3-3 is listed below. To use ifstream to read text file, <fstream> library is required to be included.

```
// file: lab3-3.cpp
#include <fstream>
#include <iostream>
using namespace std;

int main()
{
    ifstream in_file;
    in_file.open("lab3-3.txt");

    int vali;
    double vald;
    char valc[50];

    in_file >> vali >> vald >> valc;

    cout << "Integer number in file: " << vali << endl;
    cout << "Floating number in file: " << vald << endl;
```

```
        cout << "Character in file: " << valc << endl;

        in_file.close();
        return 0;
}
```

- in_file is declared as an input stream variable. open() is one of member function of in_file to open the text file and close() is used to close the text file.
- Use the extraction operator >> to take input from the file

LAB 3-4: FILE OUTPUT STREAM

- Program lab3-4 gives an example to output the results to file lab3-4.txt. Compare with ifstream to read file, an output stream ofstream is used to write to the file

```
// file: lab3-4.cpp
#include <fstream>
#include <iostream>
using namespace std;

int main()
{
        char *filename;
        cout << “Enter the filename:”;
        cin >> filename;

        ofstream out_file(filename);

        int n;
        cout << "Enter an number:";
        cin >> n;
        for (int idx = 0; idx < n; idx++)
        {
                out_file << idx << endl;
        }

        cout << "Write file lab10-2.txt done!" << endl;
        out_file.close();

        return 0;
}
```

- Below shows the result of executing program lab3-4.

```
> ./lab3-4
Enter the filename: lab3-4.txt↲
Enter an number: 5↲
Write file lab3-4.txt done!
> cat lab3-4.txt
0
1
2
```

```
3
4
```

LAB 3-5: FILE INPUT STREAM: IFSTREAM

✓ Program lab3-5 provides an example of using for file input stream to read information from the file lab3-5.txt.

✓ Here below shows the content of file lab10-3.txt.

```
10 20 30 40 50 60 70 80 90 100
```

✓ Program lab3-5 asks the user to input the filename to open and check if the file can be opened successfully.

```
//File: lab3-5.cpp
#include <fstream>
#include <iostream>
#include <string>
using namespace std;

int main()
{
	string filename;
	cout << "Enter the filename:";
	cin >> filename;

	ifstream in_file;
	in_file.open(filename.c_str());
	if (in_file.fail())
	{
		cout << "File " << filename << " does not exist !!" << endl;
		exit(1);
	}

	int num;
	int count = 0;

	while (in_file >> num)
	{
		cout << count << ": " << num << endl;
		++count;
	}

	in_file.close();
	return 0;
}
```

➢ Since the member function open takes a c-string variable as its argument, you can use the string member function c_str() to convert the data type from string to c-string.

➢ The statement while (in_file >> num) returns true if the read operation succeeds and returns false when the program attempts to read beyond the end of the file.

- ✓ Program lab3-6 gives an example to read the text file lab3-6.txt.
- ✓ Here below shows the content of file lab3-6.txt.

```
    When your program takes input from a file,
it is said to be reading from the file; when
your program sends output to a file, it is said
to be writing to the file.
```

- ✓ Program lab3-6 uses a different method to open the file.

```
// file: lab3-6.cpp
#include <fstream>
#include <iostream>
#include <string>
using namespace std;

int main(int argc, char *argv[])
{
    char *filename = argv[1];

    ifstream in_file(filename);
    if (in_file.fail())
    {
        cout << "File " << filename << " does not exist !!" << endl;
        exit(1);
    }

    string textline;
    int count = 0;
    while (getline(in_file,textline,'\n'))
    {
        cout << count << ": " << textline << endl;
        ++count;
    }

    in_file.close();
    return 0;
}
```

- ➢ Use ./lab3-6 lab3-6.txt to examine the results of this program
- ➢ The statement getline(in_file,textline,’\n’) returns true if the read operation succeeds and returns false when the program attempts to read beyond the end of the file.

Exercise 3-1 (COMPLEX NUMBER)

- ✓ Create a Complex class to perform complex number arithmetic and write a program to test your class. The class provides four complex operations: addition, subtraction, multiplication and division. The sample output is shown as follows

```
(1.0, 7.0) + (9.0, 2.0) = (10.0, 9.0)
(1.0, 7.0) - (9.0, 2.0) = (-8.0, 5.0)
(1.0, 7.0) * (9.0, 2.0) = (-5.0, 65.0)
(1.0, 7.0) / (9.0, 2.0) = (0.3, 0.7)
(10.0, 7.0) - (9.0, -1.0) = (1.0, 8.0)
```

✓ The main structure of the program is like as

```
// Complex.h
#ifndef COMPLEX_H
#define COMPLEX_H

// Write class definition for Complex


#endif
```

```
// Complex.cpp
#include <iostream>
#include "Complex.h"
using namespace std;
// Member-function definitions for class Complex.
```

```
// ex3-1.cpp
#include <iostream>
#include "Complex.h"
using namespace std;

int main()
{
    Complex a, b, c; // create three Complex objects

    a.assign(1.0,7.0);
    b.assign(9.0,2.0);
    a.printComplex(); // output object a
    cout << " + ";
    b.printComplex(); // output object b
    cout << " = ";
// invoke add function and assign to object c
    c = a.add(b);
    c.printComplex(); // output object c
    cout << endl;

    a.printComplex(); // output object a
    cout << " - ";
    b.printComplex(); // output object b
    cout << " = ";
    c = a.subtract(b); // invoke subtract function
    c.printComplex(); // output object c
```

```
        cout << endl;

        a.printComplex(); // output object a
        cout << " * ";
        b.printComplex(); // output object b
        cout << " = ";
        c = a.multiply(b); // invoke subtract function
        c.printComplex(); // output object c
        cout << endl;

        a.printComplex(); // output object a
        cout << " / ";
        b.printComplex(); // output object b
        cout << " = ";
        c = a.division(b); // invoke subtract function
        c.printComplex(); // output object c
        cout << endl;

        a.assignReal(10.0); // reset object a
        b.assignImage(-1.0); // reset object b
        a.printComplex(); // output object a
        cout << " - ";
        b.printComplex(); // output object b
        cout << " = ";
// invoke subtract function and assign to object c
        c = a.subtract( b );
        c.printComplex(); // output object c
        cout << endl;

        return 0;
}
```

Exercise 3-2:

✓ Please write a program to extend lab3-5. The new program can read an input file and print the data on the screen. The result which has counts the number of words in each sentence and displays words one-by-one.

➢ Here below shows the content of file ex3-2.txt

```
I am Charles.
I am a handsome boy.
I love C++ programing.
```

➢ The comment shows as follow:

```
> ./ex3-2 ex3-2.txt
line 1 has 3 words, "I", "am", "Charles"
line 2 has 5 words, "I", "am ", "a", "handsome", "boy"
line 3 has 4 words, "I", "love", "C++", "programing"
```

➢ HINT: This program can find the number of space in a sentence and includes two string functions:

A. please check the function **find**,
http://www.cplusplus.com/reference/string/string/find/

B. please check the function **substr**,
http://www.cplusplus.com/reference/string/string/substr/

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string str = "I am a good student";
    int position1, position2 = -1, cnt = 0;
    while(1) {
        position1 = str.find(" ",position2+1);
        cout << str.substr(position2+1, position1-position2) << endl;
        if(position1==-1) break;
        position2 = position1;
        cnt++;
    }
    cout << "number of space : " << cnt << endl;
    return 0;
}
```