

UEE1303(1602) S12: Object-Oriented Programming

C/C++ Overview



What you will learn from Lab 1

In this laboratory, you will learn the basic concept of C/C++ programming.

TASK 1-1 NAMESPACES

- ✓ Please execute the program lab1-1. Please try to identify the scope of variable defined in namespace Complex.

```
// lab1-1.h
namespace Complex{
    typedef struct{
        double real;
        double image;
    }Cplex;

    const double pi = 3.1416;
    void showComplex(const Cplex &m);
}
```

```
// lab1-1.cpp
#include <iostream>
#include "lab1-1.h"
namespace Complex{
    void showComplex(const Cplex &m)
    {
        std::cout << m.real;
        if (m.image < 0)
            std::cout << m.image << "i" << std::endl;
        else
            std::cout << "+" << m.image << "i" << std::endl;
    }
}
```

```
// lab1-1-main.cpp
#include <iostream>
#include "lab1-1.h"

int main()
{
    Complex::Cplex n;
    n.real = 1 * pi;
```

```
n.image = -0.5;  
Complex::showComplex(n);  
return 0;  
}
```

- Please modify the compiler error.

- ✓ Please modify lab1-1-main.cpp as following and execute the program again.

```
// lab1-1-main.cpp  
#include <iostream>  
#include "lab1-1.h"  
  
using namespace Complex;  
  
int main()  
{  
    Cplex n;  
    n.real = 1 * pi;  
    n.image = -0.5;  
    showComplex(n);  
    return 0;  
}
```

- Can you differentiate the above two programs?

TASK 1-2 SCOPE OPERATOR(::)

- ✓ Please compile and execute the program lab1-2.

```
// lab1-2.cpp  
#include <iostream>  
  
const int n = 10000;  
  
int main()  
{  
    int n = 10;  
    std::cout << n << " " << ::n << std::endl;  
    return 0;  
}
```

- Scope operator is usually used to qualify the member defined in specific namespace. For example, scope operator in `std::cout` is used to qualify the member function `cout` under namespace `std`.
- Scope operator can also indicate the member function under global namespace, as `::n` is in this example

TASK 1-3 INLINE FUNCTIONS

- ✓ The following two examples are used to illustrate the difference between inline functions and define macro.

```
// lab1-3-1.cpp
#include <iostream>
#define Area(x,y) ((x)*(y))

int main()
{
    double n = Area(3,5.1);
    std::cout << "Area(3,5.1) = " << n << std::endl;
    return 0;
}
```

```
// lab1-3-2.cpp
#include <iostream>
inline int Area(int x,int y) {return x*y;}

int main()
{
    double n = Area(3,5.1);
    std::cout << "Area(3,5.1) = " << n << std::endl;
    return 0;
}
```

- Although define macro results in the answer as you expect, inline functions follow all the protocols of type safety enforced on normal functions. While you compile the lab1-3-2.cpp file using inline functions, compiler shows the warning message about error type conversion but it does not so as you compile lab1-3-1.cpp.

TASK 1-4 DIRECTIVES

- ✓ Please compile and execute the lab1-4 to understand the usage of ifdef/ifndef directive.

```
// lab1-4-1.cpp
#include <iostream>
#ifndef PI
#define PI 3.14159
#endif
int main()
{
    std::cout << PI << std::endl;

    return 0;
}
```

```
// lab1-4-2.cpp
#include <iostream>

#define PI 3.1416
#ifndef PI
#define PI 3.14159
#endif

int main()
{
    std::cout << PI << std::endl;
    return 0;
}
```

- ✓ Please compare the results above two programs and explain the reason.

TASK 1-5 EXERCISES

1. COMPLEX ARITHMETIC

- ✓ Please write a C++ program to operate the arithmetic of complex number. You have to read two complex numbers from “complex.txt.” and output the arithmetical result of the two complex numbers to “result.txt.” The example files content are as follow.

The command-line usage of the program is shown below

```
> ./pg01 complex.txt result.txt
```

The required format for a sample input file “ex1-1a.in” as “complex.txt” is shown as follows.

```
1.5+6i
-2-10i
```

In the “complex.txt” file, the first line and the second line indicates the complex number A and B, respectively. The representation of complex number is $a+bi$, where a means the real part and b means the image part. If a is equal to zero, it can be written as $0+bi$ instead of bi . For the same reason, it should be written as $a+0i$ while the complex number has no image part.

Your result should be written to user-specified output file. The sample output for “ex1-1a.in” is file “ex1-1a.out” with the content shown as below.

```
-0.500-4.000i    // A + B
3.500+16.000i    // A - B
57.000-27.000i   // A * B
-0.606+0.029i    // A / B
```

✓ Requirement

In this exercise, you need to extract information from the given text file. To process a text file, you have to observe the format of this text file first. For example, each line indicates a complex number, there is no space in the line, and the numbers are separated by '+', '-' and 'i'.

➤ Main function:

```
// ex1-1.cpp
int main(int argc, char *argv[])
{
    Cplex a, b;    // use struct named Cplex under namespace Complex
    ReadTextFile(argv[1], a, b); // process text file
    Cplex results[4];           // store the results of diff. operation
    results[0] = ComplexOperation(a, b, '+');
    results[1] = ComplexOperation(a, b, '-');
    results[2] = ComplexOperation(a, b, '*');
    results[3] = ComplexOperation(a, b, '/');
    PrintComplex(argv[2], results); // print the results on file.

    return 0;
}
```

You have to finish this exercise by above structure and write three functions: ReadTextFile(), ComplexOperation() and PrintComplex().

✓ Extra Test case

You can use extra test case to test your program.

The file is "ti01b.in".

```
1.5+6i
1.5-6i
```

The output should be “ti01b.out”

```
3.000+0.000i
0.000+12.000i
38.250+0.000i
-0.882+0.471i
```

2. DIFFERENTIATION OF POLYNOMIALS

- ✓ Given an arbitrary polynomial with a degree n

$$f(x) = c_n x^n + c_{n-1} x^{n-1} + \dots + c_1 x^1 + c_0, \quad n \text{ and } c_i \in \mathbb{Z}, i \in [0, n]$$

Please write a program to read a file containing a polynomial and find the differentiation of the polynomials. Write the result to the output file. In this example, n is less than 1000. The command-line usage of the program is shown below

```
> ./ex02 input_file.in ouput_file.out
```

The required format for a sample input file “01.in” is shown as follows.

```
2X^2+X+1
```

Note that the representation of polynomial may not be in descending order. It can be $X+1+2X^2$ in above example.

Your result should be written to user-specified output file. The sample output for “ex02a.in” is file “ex02a.out” with the content shown as below.

```
4X+1
```

- ✓ Requirement

The main function of your program is specific as follows,

➤ Data structure and main function:

```
// ex1-2.cpp
const int MAX = 1000;
typedef struct
```

```
{
    char coeff[10];
    int power;
}Items;
typedef struct
{
    Items items[MAX];
    int num_items;
}Poly;

int main(int argc, char *argv[])
{
    Poly eq;
    ReadTextFile(argv[1], eq); // process text file
    DifferentiationPoly(eq);    // differentiation of polynomials
    PrintComplex(argv[2], eq); // print the results on file.
    return 0;
}
```