Lab Manual

CS112L – Object Oriented Programming Lab

Lab No: **07**

Topic: **Separation of interface and implementation**

Class: **BSGM**

Semester: **II**

Session: **Spring, 2022**

Instructor: Ms. **Saira Qamar**

Lab Date: **March 22th, 2022**

Lab Time: **11:40hrs – 2:30hrs**

Air University Islamabad

FACULTY OF COMPUTING & ARTIFICIAL INTELLIGENCE

Instructions

Submission: Use proper naming convention for your submission file. Name the submission file as **LabNO_ROLLNUM (e.g. Lab01_00000).** Submit the file on Google Classroom within the deadline. Failure to submit according to the above format would result in deduction of 10% marks. Submissions on the email will not be accepted.

Plagiarism: Plagiarism cases will be dealt with strictly. If found plagiarized, both the involved parties will be awarded zero marks in the assignment, all of the remaining assignments, or even an F grade in the course. Copying from the internet is the easiest way to get caught!

Deadline: The deadlines to submit the assignment are hard. Late submission with marks deduction will be accepted according to the course policy shared by the instructor. Correct and timely submission of the assignment is the responsibility of every student; hence no relaxation will be given to anyone.

Comments: Comment your code properly. Bonus marks (maximum 10%) will be awarded to well comment code. Write your name and roll number (as a block comment) at the beginning of the solution to each problem.

Tip: For timely completion of the assignment, start as early as possible. Furthermore, work smartly - as some of the problems can be solved using smarter logic.

1. Note: Follow the given instructions to the letter, failing to do so will result in a zero.

Objectives

In this lab, you will learn:

- Separation of interface and implementation

Concepts

1. Introduction:

An Interface defines a set of method **signatures**, while a Class defines the **structure and behavior of its instances.** When it comes to solving real life problems, only interface of an object should be visible to outside world and actual implementation needs to be hidden from outside world.

The **benefit** of using this approach is that our object interface to outside word becomes independent from inside implementation of that interface. This is achieved through the concepts of **encapsulation** and **information hiding**.

2. Real Life example of separation of interface and implementation:

Driver has a standard **interface** to drive a car and using that **interface** he can drive any car regardless of its model or type whateverengine type it has or whatever type of fuel it is using.

In C++, **public member functions** exposed by a class are called **interface**.

3. Benefits of separating interface and implementation:

- Separate **interface** and **implementation** allows us to **easily change implementation without changing interface** and makes our software flexible, extensible, portable and modular.
- User is **only concerned about ways of accessing data** (interface).
- User has **no concern** about the **internal representation and implementation** ofthe class.

4. Implementation:

In C++, generally we can relate the concept of interface of a class to its header (.h) file and implementation of a class to its (.cpp) file. However it is not complete separation of interface and implementation.

- Usually functions are defined in implementation file (.cpp) while the class definition is given in header file (.h).

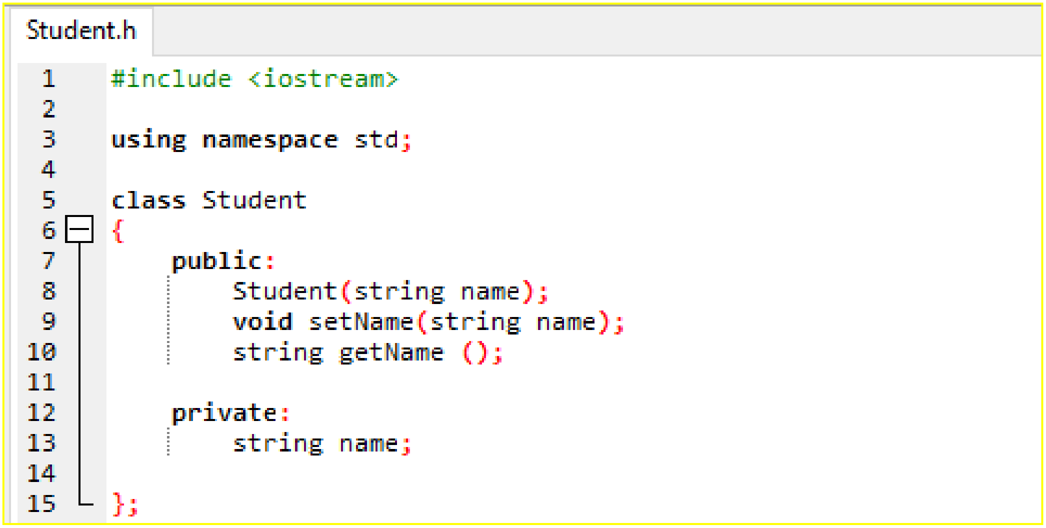
Demo

The following **example** illustrates the separation of **interface** and **implementation:**

Header Files:

Class declarations are stored in a separate file. A file that contains a class declaration is called header file. The name of the class is usually the same as the name of the class, with a .h extension.

- Write the following code in **Student.h file**:



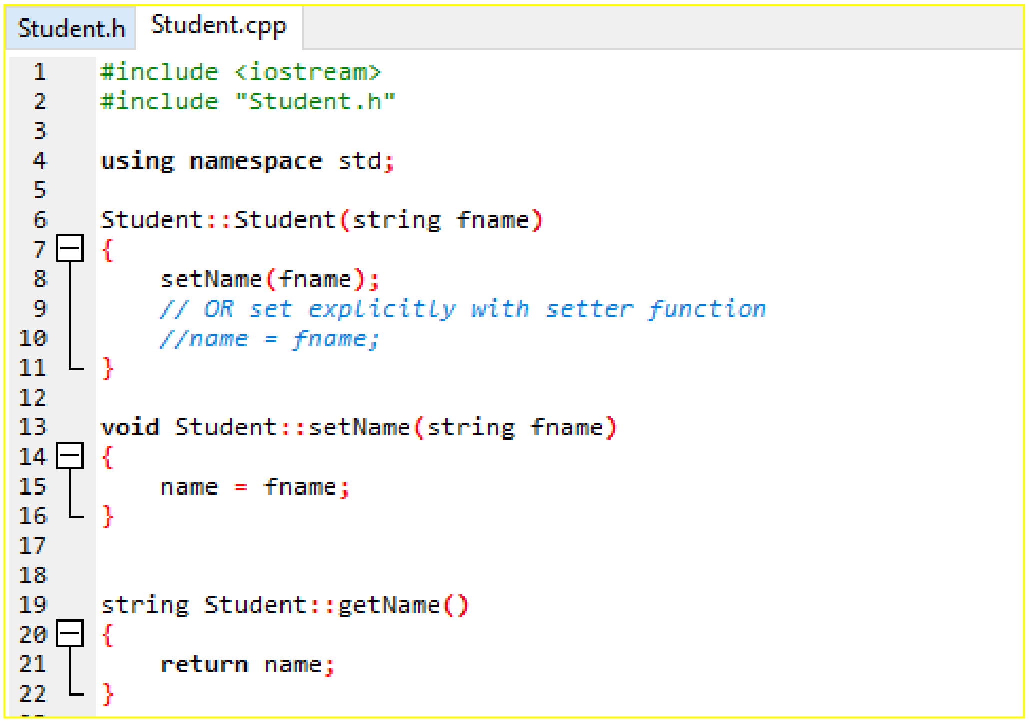
Student.h

```
#include <iostream>

using namespace std;

class Student
{
    public:
        Student(string name);
        void setName(string name);
        string getName ();

    private:
        string name;

};
```

Implementations Files:

The member function definitions for a class are stored in a separate .cpp file, which is called the class implementation file. The file usually has the same name as the class, with the .cpp extension.

- Write the following code in **Student.cpp file**:

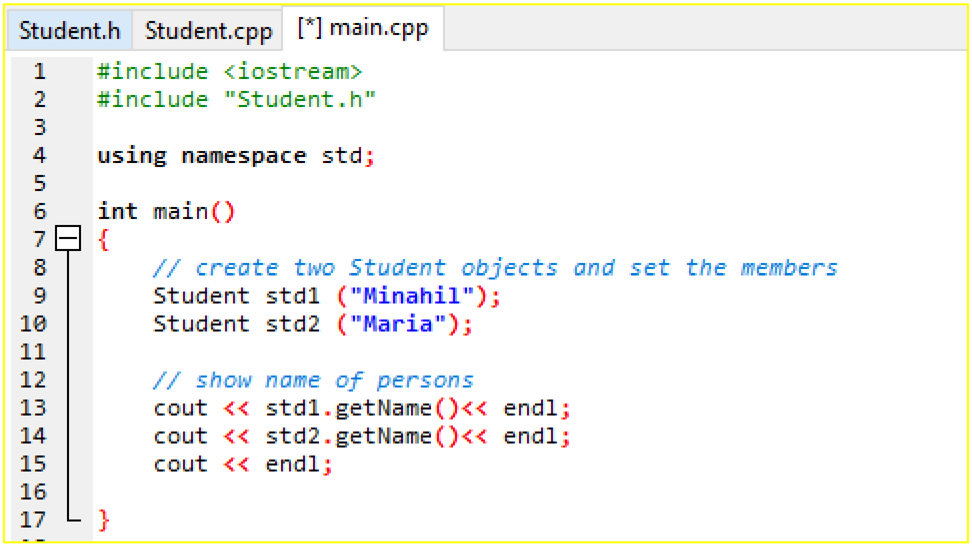


```cpp
#include <iostream>
#include "Student.h"

using namespace std;

Student::Student(string fname)
{
    setName(fname);
    // OR set explicitly with setter function
    //name = fname;
}

void Student::setName(string fname)
{
    name = fname;
}


string Student::getName()
{
    return name;
}
```

Implementations Files:

Client code, is the one that includes the main function. This file should be stored by the name main.cpp.

- Write the following code in **main.cpp file**:



```cpp
#include <iostream>
#include "Student.h"

using namespace std;

int main()
{
    // create two Student objects and set the members
    Student std1 ("Minahil");
    Student std2 ("Maria");

    // show name of persons
    cout << std1.getName()<< endl;
    cout << std2.getName()<< endl;
    cout << endl;

}
```

Run and compile the file from main.cpp.

Lab Tasks

1. Create a class called Employee that includes three pieces of information as data members—a first name (type string), a last name (type string) and a monthly salary (type int). Your class should have a constructor that initializes the three data members. Provide a set and a get function for each data member. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again.