

Lab Manual for Object Oriented Programming (LAB-02)

Introduction to Classes and Objects

Table of Contents

1.	Introduction	15
2.	Activity Time boxing	15
3.	Objective of the Experiment	15
4.	Concept Map	16
4.1	Object	16
4.2	Data Members	16
4.3	Member Functions	16
4.4	Constant Member Functions	17
4.5	Class	17
4.6	Encapsulation	17
5.	Home Work Before Lab	18
5.1	Problem Solution Modelling	18
5.2	Practices from home	18
6.	Procedure & Tools	18
6.1	Tools	18
6.2	Setting-up Visual Studio 2008	18
6.3	Walkthrough Task	18
7.	Practice Tasks	20
7.1	Practice Task 1	20
7.2	Practice Task 2	20
7.3	Practice Task 3	20
7.4	Outcomes	20
7.5	Testing	20
8.	Evaluation Task (Unseen)	21
9.	Evaluation Criteria	21
10.	Further Reading	21
10.1	Books	21
10.2	Slides	21

Lab 02: Introduction to Classes and Objects

1. Introduction

The programming that you have understood in your computer programming course, allows you to design a program by assembling a sequence of instructions. This is the way in which traditional programming works, but now there is clear paradigm shift towards an object based approach. The new object oriented approach allows you to model your program in a way in which objects, their relationships and the functions that they operate are clearly visible to the programmer.

Perhaps the greatest advantage of this approach is that it gives an extended modular concept in which the interface is made public while the implementation details are kept hidden. This allows the program to grow in size without becoming too cumbersome. Previously, when a program grew beyond a certain size it became almost impossible to understand and extend especially when the programmer had forgotten the minor concepts constructing the program. Here, it should be understood and acknowledged that programs are not disposable entities they have a life beyond what is normally expected. Hence designing a program that can be easily extended, modified and interfaced with other systems is a very important characteristic of any well written program.

This lab has been designed to give in-depth knowledge of how to make classes, objects and their interrelations. The concept map provides all the crucial details that are required for the completion of this lab.

Relevant Lecture Material

- **Lectures:** 3, 4
- **Textbook:** Object-Oriented Programming Using C++, Fourth edition, Robert Lafore
 - **Pages:** 195-201

2. Activity Time boxing

Table 1: Activity Time Boxing

Task No.	Activity Name	Activity time	Total Time
5.1	Evaluation of Design	15 mins	15 mins
6.2	Setting-up Visual Studio	5 mins	5 mins
6.3	Walkthrough Task	30 mins	30 mins
7	Practice tasks	25 + 25 + 35 (mins)	85 mins
8	Evaluation Task	45 min	45 mins
		Total Time	180 Minutes

3. Objective of the Experiment

After completing this lab the student should be able to:

- Clearly understand the purpose and benefits that OOP has to offer.
- Understand the concept of a class and objects.
- Develop a basic class with fundamental data members.
- Develop a basic class with a number of member functions.
- Use a constant member function.
- Separate the implementation section of a function.
- Use the class objects and member functions to provide and extract data from an object.
- Experiment with classes and objects.

4. Concept Map

4.1 Object

In OOP an object is a very much like a real world object. An object can be defined as a collection of state and behaviour. For example, consider the example of a cat. A cat is has both a state and behaviour. The state of a cat is its attributes namely colour, breed, gender, age, weight, height, etc. Whereas the behaviour of a cat is its sound, eating, sleeping, yawning, walk, etc. Hence a cat can be completely identified by its unique characteristics and behaviours.

In programming an object is a collection of variables and functions that together have a unique purpose of identifying a distinctive entity.

4.2 Data Members

Again referring to the concept of an object and the example of a cat. The cat had a number of characteristics or attributes that can be used to identify a cat. In programming these attributes can be programmed by using regular variables that are called data members. A class can be composed of a number of data members of various types. The data members of a class are private by default i.e. they are not directly accessible outside the class.

Here it is important to point out that often people casually refer to these as variables, which is a wrong terminology. These should only be called data members or class variables.

4.3 Member Functions

Again referring to the concept of an object and the example of a cat. The cat had a number of behaviours or things that a cat does. In programming these behaviours can be programmed by using functions that are called member functions. A class can be composed of a number of member functions of various types. Overloading of member functions is also permitted in C++. The implementation section of member functions can be separated whereby the body of the function is created outside the class but the function prototype has to exist in the body of the class. The implementation section is separated by writing the return type followed by class name. This is further followed by scope resolution operator `::` and then the remaining function definition.

```
using namespace std;
class myclass
{
    int datamember;

    int memberfun(int);           // Function prototype
};

int myclass :: memberfun (int x) // Note the scope resolution operator
{
    ...                          // Function body
}

void main( )
{
}
```

4.4 Constant Member Functions

A constant member function is just like a conventional function except it is a read only function. This means that the function cannot modify anything related to the object. But the member function can still perform all reading activities related to the object. Such type of functions are generally created when the programmer wishes to only read/ display the data members on the screen. Given below is the syntax for creating these constant member functions.

```
void addtwo ( ) const    //Note the keyword const
{
    cout<<"The sum is= "<<s;
}

```

4.5 Class

A class is a collection of data members and member functions. A class is used to define an abstract data type. This abstract data type is used in the construction of an object which is used to access all the data members and member functions. A class has to be created before it can be used. Provided below are the syntax for creating a class.

```
using namespace std;
class myclass
{
    int datamember;

    void memberfun (int)
        {
            ...
        }
}; //Semicolon is necessary

void main()
{
}

```

```
using namespace std;
class myclass;

void main()
{
}

class myclass
{
    int datamember;

    void memberfun (int)
        {
            ...
        }
}; //Semicolon is necessary

```

4.6 Encapsulation

Encapsulation is a very important design goal because of which OOP is preferred over conventional programming. Encapsulation is a quality because of which data and function are stored in a single unit commonly known as a class. This unit/ bundle ensures that the data is not openly accessible to the outer world. The access is provided by the functions that are part of the unit/bundle. This means that the functions work like doors in a room. You can prevent thieves from breaking in. Only those people can enter who come through the door.

5. Home Work Before Lab

5.1 Problem Solution Modelling

Design the following problem by listing down the data members and the member functions. **You are required to bring this task with you and submit to the lab instructor.**

5.1.1 Problem description

List down the data members and member functions of a class that will identify a vehicle. You are allowed to suppose the data members and the member functions of the class. Also create a single member function that will display the entire class data members. You will be graded on the quality and clarity of your design.

5.2 Practices from home

5.2.1 Task-1

Identify the data members and member functions for a pizza class. The class should have all the relevant attributes and qualities required for a pizza object. Try to be imaginative in your design.

6. Procedure & Tools

6.1 Tools

Visual Studio 2008.

6.2 Setting-up Visual Studio 2008

[Expected time = 5 mins]

Setup Visual Studio and make a project named “cylinder”.

6.3 Walkthrough Task

[Expected time = 30 mins]

Write a program that creates a class called cylinder. The data members of the class are radius and height of the cylinder is provided. Write two functions that set the respective values and then write a single constant function that will read the values.

6.3.1 Writing Code

In the source file created in the project “cylinder” write the following C++ code:

```
using namespace std;
class cylinder
{
    float radius;
    float height;
public:
    void SetRadius(float);           //Function Prototypes
    void SetHeight();
    void display() const           //Const function
    {
        cout<<"The radius is "<<radius;
        cout<<"\n\nThe height is "<<height<<"\n";
    }
};
void cylinder::SetRadius(float r)  //Seperate implementation Sections
{
    radius=r;
}
void cylinder::SetHeight()
{
    cout<<"Enter the height ";
    cin>>height;
}
int main()
{
    cylinder obj;
    obj.SetRadius(3.5);
    obj.SetHeight();
    obj.display();
    system("pause");
    return 0;
}
```

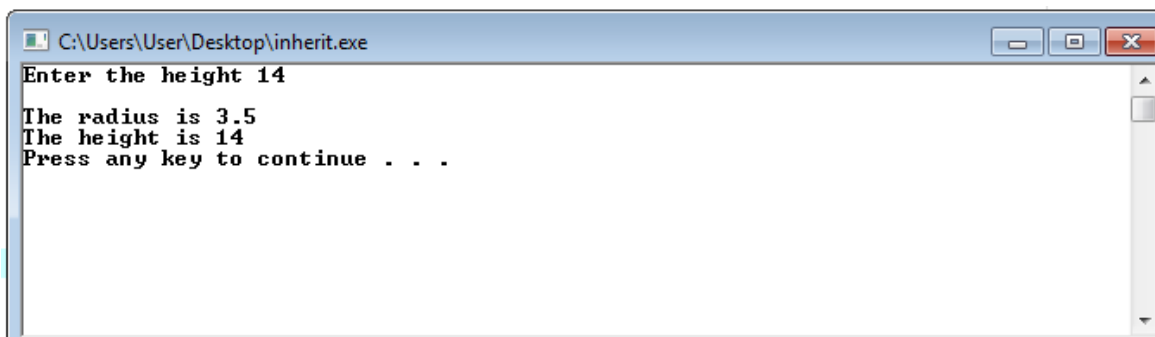
Figure 1: Class for creating a cylinder with its relevant data members

6.3.2 Compilation

After writing the code, compile your code according to the guidelines mentioned. Remove any errors and warnings that are present in your code.

6.3.3 Executing the Program

A sample output after running the program is shown below. Also run the code with other possible inputs.



```
C:\Users\User\Desktop\inherit.exe
Enter the height 14
The radius is 3.5
The height is 14
Press any key to continue . . .
```

Figure 2: Final output of cylinder program.

7. Practice Tasks

This section will provide more practice exercises which you need to finish during the lab. You need to finish the tasks in the required time. When you finish them, put these tasks in the following folder:

\\dataserver\assignments\$\OOP\Lab02

7.1 Practice Task 1

[Expected time = 25 mins]

Write a C++ program that creates a class called laptop. The data members of the class are brand (string), model (string), serial (int), colour (string), price (float), processor speed (float), RAM (int), screen size(float).

Create member function that will set the individual values. Since the RAM can be upgraded therefore create a function that allows you to upgrade the RAM only. In the end, create a function that will display all the data members.

7.2 Practice Task 2

[Expected time = 25 mins]

Write a class called rectangle. Your task is to store the length and width of the rectangle. Write a member function called increment that will add 1 to the value of length and width. Also write a function that will compute the area of the rectangle. Finally write a constant function that will display the length, width and area of the rectangle.

Demonstrate the use of the object in the main function. Make sure that the function names are meaningful.

7.3 Practice Task 3

[Expected time = 35 mins]

Write a program that creates a class called number. Your class will have two data members namely num (float) and result (int). To find the factorial of the entered number you will need to design three functions as follows:

- Function to determine if a number is a whole number or not
- Function to determine if the number is positive or not
- Function to find the actual factorial
- Function to display the number and its factorial

Remember that to find the factorial the number must of positive and a whole number. So if any of these conditions are not met then you cannot determine the factorial.

7.4 Outcomes

After completing this lab, students will be able to design a basic class with data members and member functions.

7.5 Testing

Test Cases for Practice Task-1

Sample Inputs	Sample Outputs
Set the following values Brand = DELL Model = 1565D Serial = 123456 Colour = Silver Price = 64500.5 Processor = 2.8 RAM = 2 Screen = 15.6	Brand = DELL Model = 1565D Serial = 123456 Colour = Silver Price = 64500.5 Processor = 2.8 RAM = 3 Screen = 15.6
Then Modify the RAM to 3	

Test Cases for Practice Task-2

Sample Inputs	Sample Outputs
Length: 4 width: 10	Length = 5 Width = 11 Area = 55
call the increment function	Check the use of constant function for display

Test Cases for Practice Task-3

Sample Inputs	Sample Outputs
Number: -5	Invalid Input
Number: 5.5	Invalid Input
Number: 5	Number = 5 Factorial = 120

Table 2: Confirmation of practice tasks T1, T2 and T3

Practice Tasks	Confirmation
T1	
T2	
T3	

8. Evaluation Task (Unseen)**[Expected time = 45 Minutes]**

The lab instructor will assign you an unseen task depending upon the progress of the students.

9. Evaluation Criteria

The evaluation criteria for this lab will be based on the completion of the following tasks. Each task is assigned marks which will be evaluated by the instructor in the lab depending on the accomplishment of the assigned tasks.

Table 3: Evaluation of the Lab

Sr. No.	Task No	Description	Marks
1	4.1	Problem Modelling	20
2	6	Procedures and Tools	5
3	7.1	Practice task 1 with Testing	10
4	7.2	Practice task 2 with Testing	10
5	7.3	Practice task 3 with Testing	10
6	8.1	Evaluation Tasks (Unseen)	15
7		Good Programming Practices	10
Total Marks			80

10. Further Reading**10.1 Books**

- Object-Oriented Programming Using C++, Fourth edition, Joyce Farrell

10.2 Slides

The slides and reading material can be accessed from the folder of the class instructor available at [\\dataserver\jinnah\\$](\\dataserver\jinnah$)