**Lab#1**

**Objective:** To do hands-on practice on class and Objects ( OOP features).

**Theory:**

The main purpose of C++ programming is to add object orientation to the C programming language and classes are the central feature of C++ that supports object-oriented programming and are often called user-defined types.

A class is used to specify the form of an object and it combines data representation and methods for manipulating that data into one neat package. The data and functions within a class are called members of the class.

**C++ Class Definitions**

When you define a class, you define a blueprint for a data type. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

A class definition starts with the keyword class followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations. For example, we defined the Box data type using the keyword class as follows:

class Box {

public:

double length; // Length of a box

double breadth; // Breadth of a box

double height; // Height of a box

};

The keyword public determines the access attributes of the members of the class that follow it. A public member can be accessed from outside the class anywhere within the scope of the class object. You can also specify the members of a class as private or protected which we will discuss in a sub-section.

**Define C++ Objects**

A class provides the blueprints for objects, so basically an object is created from a class. We declare objects of a class with exactly the same sort of declaration that we declare variables of basic types. Following statements declare two objects of class Box:

Box Box1; // Declare Box1 of type Box

Box Box2; // Declare Box2 of type Box

Both of the objects Box1 and Box2 will have their own copy of data members.

**Accessing the Data Members**

The public data members of objects of a class can be accessed using the direct member access operator (.).

**Lab Tasks:**

* Write below given code, compile it and run it.
* Write output of code in below given box.
* Write another program(C++ CODE) with class and objects.
* Compile, run and write the output..

**Program 1 (Code):**

#include <iostream>

using namespace std;

class Box {

public:

double length; // Length of a box

double breadth; // Breadth of a box

double height; // Height of a box

};

int main( ) {

Box Box1; // Declare Box1 of type Box

Box Box2; // Declare Box2 of type Box

double volume = 0.0; // Store the volume of a box here

// box 1 specification

Box1.height = 5.0;

Box1.length = 6.0;

Box1.breadth = 7.0;

// box 2 specification

Box2.height = 10.0;

Box2.length = 12.0;

Box2.breadth = 13.0;

// volume of box 1

volume = Box1.height \* Box1.length \* Box1.breadth;

cout<< "Volume of Box1 : " << volume <<endl;

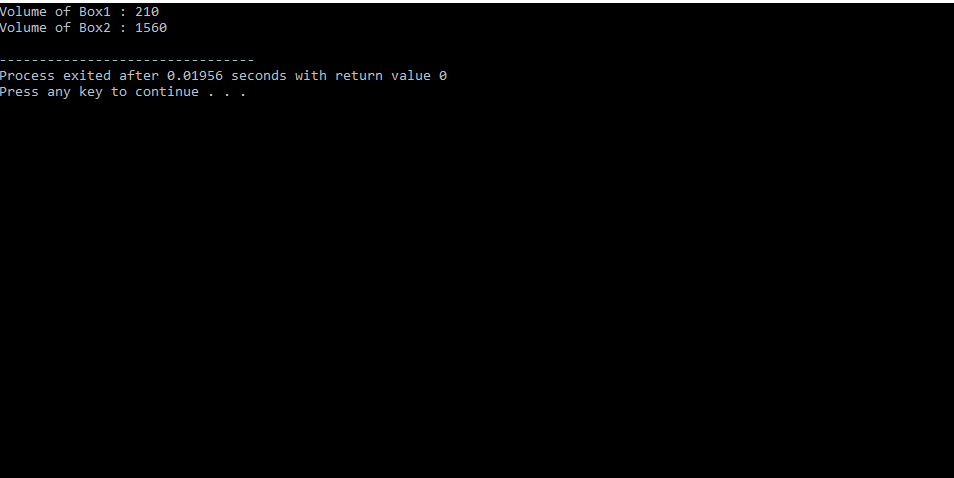
// volume of box 2

volume = Box2.height \* Box2.length \* Box2.breadth;

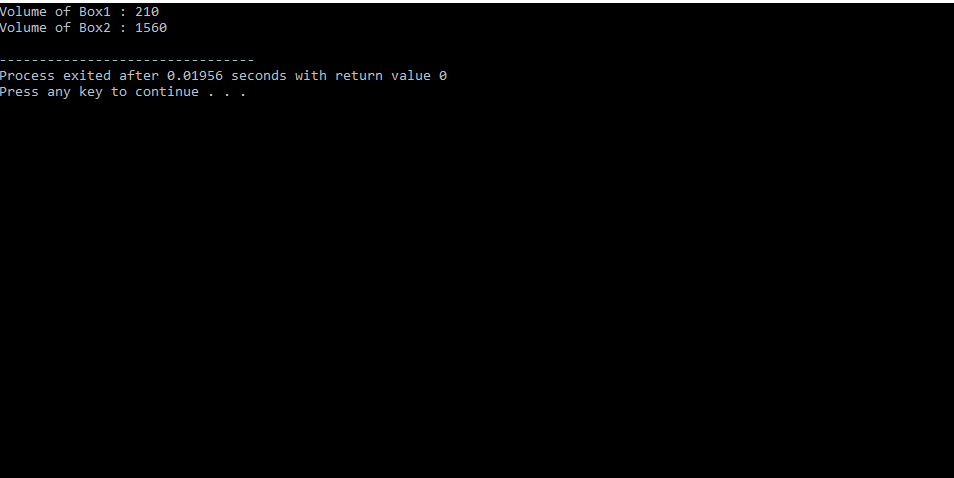
cout<< "Volume of Box2 : " << volume <<endl;

return 0;

}



**OUTPUT:**



**2nd programme of (OOP) to demonstrate the class and objects:**

**Which calculate sum and average of three numbers using class and objects.**

#include<iostream>

using namespace std;

class Marks{

private:

int a,b,c;

public:

void in(){

cout<<"enter three numbers"<<endl;

cin>>a>>b>>c;

}

int sum(){

return a+b+c;

}

int avg(){

return(a+b+c)/3;

}

};

int main()

{

Marks m;

float a;

int s;

m.in();

s=m.sum ();

a=m.avg ();

cout<<"sum is"<<" "<<s<<endl;

cout<<"avg is"<<" "<<a<<endl;

return (0);

}

**OUTPUT:**

