Assignment 1

What is a Class?

A class is a blueprint for the object.
We can think of a class as a sketch (prototype) of a house. It contains all the details about the floors, doors, windows, etc. Based on these descriptions we build the house. House is the object.

To Create a Class

A class is defined in C++ using the keyword **class** followed by the name of the class.

The body of the class is defined inside the curly brackets and terminated by a semicolon at the end.

```
class className {
  // some data
  // some functions
};
```

Example

```
class Room {
   public:
      double length;
      double breadth;
      double height;

      double calculateArea(){
         return length * breadth;
      }

      double calculateVolume(){
         return length * breadth * height;
      }

};
```

Here, we defined a class named Room.

The variables *length*, *breadth*, and *height* declared inside the class are known as **data members**. And, the functions calculateArea() and calculateVolume() are known as **member functions** of a class.

Objects of a class

When a class is defined, only the specification for the object is defined; no memory or storage is allocated. A class is an abstract entity.

Syntax to Define Object in C++

```
className objectVariableName;
```

We can create objects of Room class (defined in the above example) as follows:

```
// sample function
void sampleFunction() {
// create objects
   Room room1, room2;
}

int main(){
   // create objects
   Room room3, room4;
}
```

Here, two objects *room1* and *room2* of the Room class are created in sampleFunction(). Similarly, the objects *room3* and *room4* are created in main().

As we can see, we can create objects of a class in any function of the program. We can also create objects of a class within the class itself, or in other classes.

Also, we can create as many objects as we want from a single class.

C++ Access Data Members and Member Functions

We can access the data members and member functions of a class by using a . (dot) operator. For example,

```
room2.calculateArea();
```

This will call the calculateArea() function inside the Room class for object *room2*.

Similarly, the data members can be accessed as:

```
room1.length = 5.5;
```

In this case, it initializes the *length* variable of *room1* to 5.5.

```
#include <iostream>
using namespace std;


// create a class
class Room {


  public:
   double length;
   double breadth;
   double height;


   double calculateArea() {
      return length * breadth;
   }
```

```
    double calculateVolume() {
        return length * breadth * height;
    }
};

int main() {// create object of Room class
    Room room1;

    // assign values to data members
    room1.length = 42.5;
          room1.breadth = 30.8;
    room1.height = 19.2;

    // calculate and display the area and volume of the room
    cout << "Area of Room =  " << room1.calculateArea() << endl;
    cout << "Volume of Room =  " << room1.calculateVolume() << endl;

    return 0;
}
```

Out is the following

```
Area of Room =  1309
Volume of Room = 25132.8
```

Do the following exercise.

a. Make the following attributes private and make modifications to the program

```
double length;
double breadth;
double height;
```

Hint : - Create set_length(),set_breadth(), set_beight() functions and call them from the driver main().

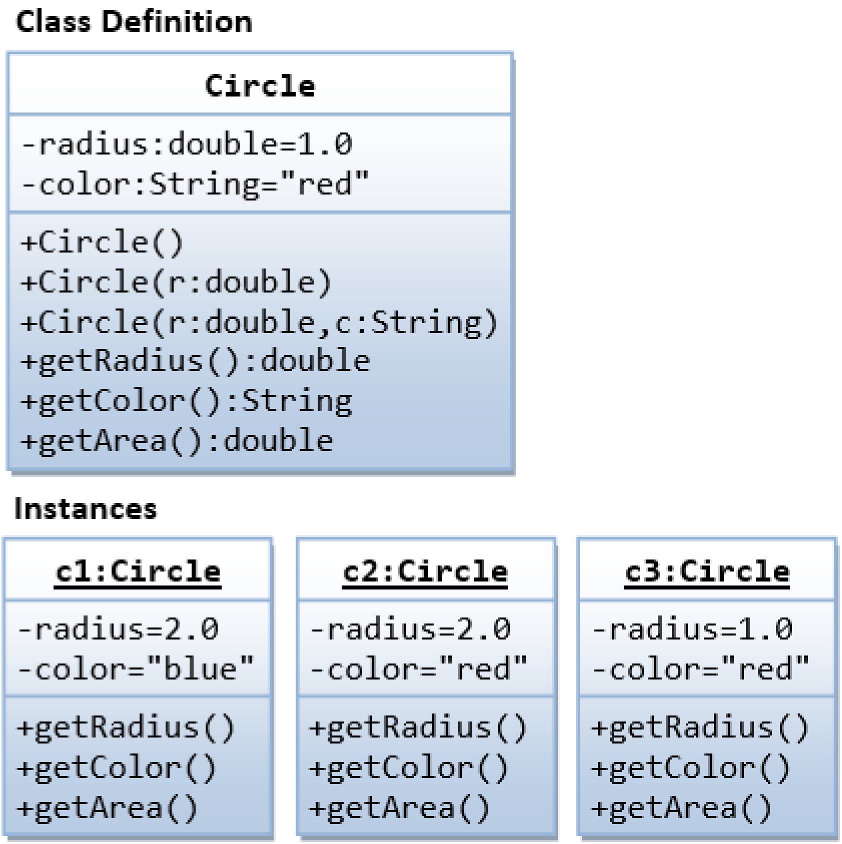
b. Add constructor to following functions. See the sample code and make modifications.

- Create a default constructor
- Create a parameterized constructor. Create an object in driver class which takes the following values.

  ```
  double length  = 10.8;
  double breadth  = 8.6;
  double height = 15.5;
  ```

- Create a copy constructor. This takes the object created in parameterized constructor as in input.

1. Create a class circle with the following data members and member functions. Create the following instances of circle and display their values assigned to the data members.



Class Definition

Circle

-radius:double=1.0
-color:String="red"

+Circle()
+Circle(r:double)
+Circle(r:double,c:String)
+getRadius():double
+getColor():String
+getArea():double

Instances

c1:Circle

-radius=2.0
-color="blue"

+getRadius()
+getColor()
+getArea()

c2:Circle

-radius=2.0
-color="red"

+getRadius()
+getColor()
+getArea()

c3:Circle

-radius=1.0
-color="red"

+getRadius()
+getColor()
+getArea()