Basics Of OOPs

Class

- Classes are the most important feature of C++ that leads to Object Oriented Programming. Class is a user defined data type, which holds its own data members and member functions, which can be accessed and used by creating instance of that class.
- Class in C++ are similar to structures in C, the only difference being, class defaults to private access control, where as structure defaults to public.
- 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.

```
class Box {

   public:             // access attribute of the members

      double length;   // Length of a box
      double breadth;  // Breadth of a box
      double height;   // Height of a box
};
```

Object

- 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.

```
Box Box1;          // Declare Box1 of type Box
Box Box2;          // Declare Box2 of type Box
```

Accessing the Data Members

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

Example Program

```
#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
};
```

Contd..

```
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;
```

Contd..

```
 // 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;
```

Contd..

```
// volume of box 2
   volume = Box2.height * Box2.length * Box2.breadth;
   cout << "Volume of Box2 : " << volume <<endl;
```

- Output of the this program will be

  Volume of Box1 : 210

  Volume of Box2 : 1560

Member Functions

- A member function of a class is a function that has its definition or its prototype within the class definition like any other variable.

```
class Box {

    public:

       double length;         // Length of a box

       double breadth;        // Breadth of a box

       double height;         // Height of a box

       double getVolume(void); // Returns box volume

};
```

Contd..

```
double Box::getVolume(void) {
       return length * breadth * height;
}
```

- A member function will be called using a dot operator (.)

```
Box myBox;           // Create an object
myBox.getVolume();   // Call member function for the object
```

Access Modifiers

- The access restriction to the class members is specified by the labeled public, private, and protected sections within the class body.
- A public member is accessible from anywhere outside the class but within a program.
- A private member variable or function cannot be accessed, or even viewed from outside the class. Only the class can access private members.

Example

```
class Box {
   public:
      double length;
      void setWidth( double wid );
      double getWidth( void );
   private:
      double width;
};
// Member functions definitions
double Box::getWidth(void) {
   return width ;
}
void Box::setWidth( double wid ) {
   width = wid;
}
```

Contd..

```
// Main function for the program
int main( ) {
   Box box;

   // set box length without member function
   box.length = 10.0;                    // OK: because length is public
   cout << "Length of box : " << box.length <<endl;

   // set box width without member function
   // box.width = 10.0;                      // Error: because width is private
   box.setWidth(10.0);                       // Use member function to set it.
   cout << "Width of box : " << box.getWidth() <<endl;

   return 0;
}
```

Constructor

- Constructor is a special member function of a class that is executed whenever we create new objects of that class.

```
class Box {
   Public:
            Box(int, int, int);   // This is the constructor
};
Box::Box(int height, int width, int length) {
   // Sets the height, width and length of the Box
}
int main( ) {
   Box Box1(10,15,20);
   display(Box1);
   return 0;
}
```

Destructor

- A destructor is a special member function of a class that is executed whenever an object of it's class goes out of scope or whenever the delete expression is applied to a pointer to the object of that class.

```
class Box {
   Public:
            Box(int, int, int);   // This is the constructor
            ~Box();               // This is the destructor
};
Box::~Box() {
   cout << "Object is being deleted" << endl;
}
```