LAB NO. 10

INHERITANCE IN C++

Lab Objectives

Following are the lab objectives:

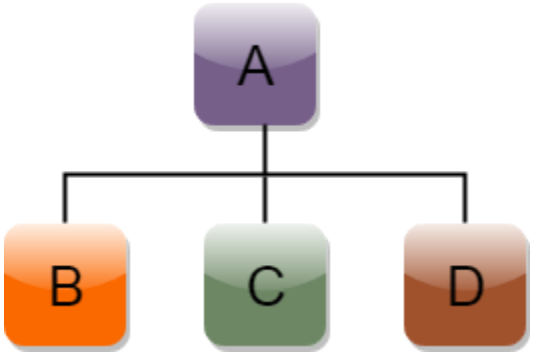
1. Inheritance in C++
2. Constructors and destructors calling order

Instructions

- This is individual Lab work/task.
- Complete this lab work within lab timing.
- Discussion with peers is not allowed.
- Copy paste from Internet will give you **negative marks**.
- Lab work is divided into small tasks, complete all tasks sequentially.

C++ Hierarchical Inheritance

Hierarchical inheritance is defined as the process of deriving more than one class from a base class. Example: Computer, Civil, Mechanical, Electrical are derived from Engineer. Natural Language and Programming Language derived from Language.



Syntax of Hierarchical inheritance:

```
class A
{
   // body of the class A.
}
class B : public A
{
   // body of class B.
}
class C : public A
{
   // body of class C.
}
class D : public A
{
   // body of class D.
}
```

Let's see a simple example:

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

```
class Shape                // Declaration of base class.
{
   public:
   int a;
   int b;
   void get_data(int n,int m)
   {
      a= n;
      b = m;
   }
};
class Rectangle : public Shape  // inheriting Shape class
{
   public:
   int rect_area()
   {
      int result = a*b;
      return result;
   }
};
class Triangle : public Shape    // inheriting Shape class
{
   public:
   int triangle_area()
   {
      float result = 0.5*a*b;
      return result;
   }
};
int main()
{
   Rectangle r;
   Triangle t;
   int length,breadth,base,height;
```

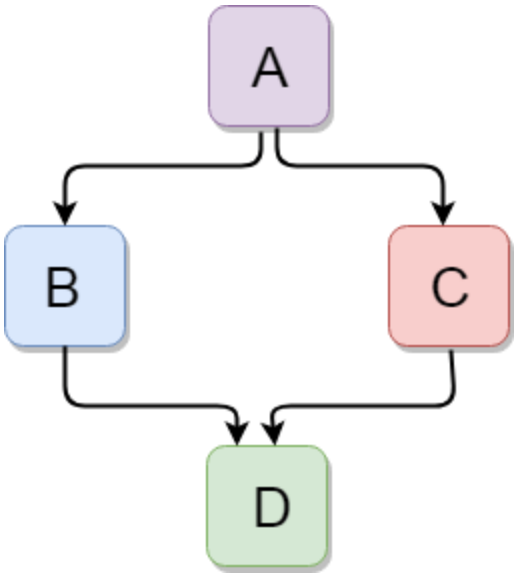
```
    cout << "Enter the length and breadth of a rectangle: " <<endl;
    cin>>length>>breadth;
    r.get_data(length,breadth);
    int m = r.rect_area();
    cout << "Area of the rectangle is : " <<m<<endl;
    cout << "Enter the base and height of the triangle: " <<endl;
    cin>>base>>height;
    t.get_data(base,height);
    float n = t.triangle_area();
    cout <<"Area of the triangle is : "  << n<<endl;
    return 0;
}
```

Output:

```
Enter the length and breadth of a rectangle:
23
20
Area of the rectangle is : 460
Enter the base and height of the triangle:
2
5
Area of the triangle is : 5
```

C++ Hybrid Inheritance

Hybrid inheritance is a combination of more than one type of inheritance. Example of hybrid inheritance is combination of multilevel and hierarchical inheritance.



Let's see a simple example:

```
#include <iostream>
using namespace std;
class A
{
   protected:
   int a;
   public:
   void get_a()
   {
     std::cout << "Enter the value of 'a' : " << std::endl;
     cin>>a;
   }
};

class B : public A
{
   protected:
   int b;
   public:
   void get_b()
```

```
    {
        std::cout << "Enter the value of 'b' : " << std::endl;
       cin>>b;
    }
};
class C
{
    protected:
    int c;
    public:
    void get_c()
    {
        std::cout << "Enter the value of c is : " << std::endl;
        cin>>c;
    }
};

class D : public B, public C
{
    protected:
    int d;

    public:
    void mul()
    {
         get_a();
         get_b();
         get_c();
         std::cout << "Multiplication of a,b,c is : " <<a*b*c<< std::endl;
    }
};
int main()
{
    D d;
```

```
    d.mul();
    return 0;
}
```

Output:

```
Enter the value of 'a' :
10
Enter the value of 'b' :
20
Enter the value of c is :
30
Multiplication of a,b,c is : 6000
```

Order of Constructor/ Destructor Call in Inheritance C++

Whenever we create an object of a class, the default constructor of that class is invoked automatically to initialize the members of the class.

If we inherit a class from another class and create an object of the derived class, it is clear that the default constructor of the derived class will be invoked but before that the default constructor of all of the base classes will be invoke, i.e the order of invokation is that the base class's default constructor will be invoked first and then the derived class's default constructor will be invoked.

Why the base class's constructor is called on creating an object of derived class?

To understand this you will have to recall your knowledge on inheritance. What happens when a class is inherited from other? The data members and member functions of base class comes automatically in derived class based on the access specifier but the definition of these members exists in base class only. So when we create an object of derived class, all of the members of derived class must be initialized but the inherited members in derived class can only be initialized by the base class's constructor as the definition of these members exists in base class only. This is why the **constructor of base class is called first to initialize all the inherited members**.

```
// C++ program to show the order of constructor call
// in single inheritance
#include <iostream>
using namespace std;
// base class
class Parent
{
    public:
    // base class constructor
```

```
    Parent()
    {
        cout << "Inside base class" << endl;
    }
};
// sub class
class Child : public Parent
{
    public:
    //sub class constructor
    Child()
    {
        cout << "Inside sub class" << endl;
    }
};
// main function
int main() {
    // creating object of sub class
    Child obj;
    return 0;
}
```

Output:

```
Inside base class
Inside sub class
```

Order of constructor call for Multiple Inheritance

For multiple inheritance order of constructor call is, the base class's constructors are called in the order of inheritance and then the derived class's constructor.

```
// C++ program to show the order of constructor calls
// in Multiple Inheritance
#include <iostream>
using namespace std;
// first base class
class Parent1
```

```
{
    public:
    // first base class's Constructor
    Parent1()
    {
        cout << "Inside first base class" << endl;
    }
};
// second base class
class Parent2
{
    public:
    // second base class's Constructor
    Parent2()
    {
        cout << "Inside second base class" << endl;
    }
};
// child class inherits Parent1 and Parent2
class Child : public Parent1, public Parent2
{
    public:
    // child class's Constructor
    Child()
    {
        cout << "Inside child class" << endl;
    }
};
// main function
int main() {
    // creating object of class Child
    Child obj1;
    return 0;
```

}

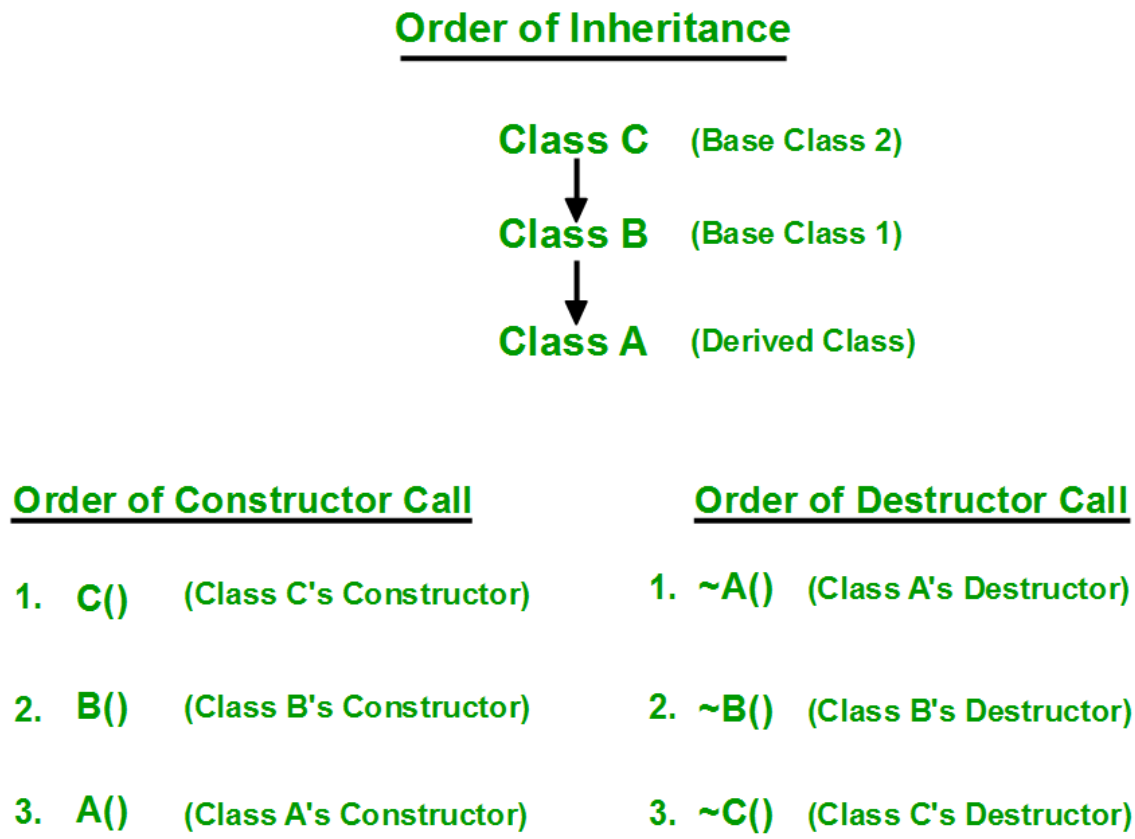
Output:

```
Inside first base class

Inside second base class

Inside child class
```

Order of constructor and Destructor call for a given order of Inheritance



How to call the parameterized constructor of base class in derived class constructor?

To call the parameterized constructor of base class when derived class's parameterized constructor is called, you have to explicitly specify the base class's parameterized constructor in derived class as shown in below program:

```
#include<iostream>

using namespace std;

class parent

{

   int x;

   public:

   // parameterized constructor
```

```
    parent(int i)
    {
        x = i;
        cout << "Parent class Parameterized Constructor\n";
    }
};
class child: public parent
{
    int y;
    public:

    // parameterized constructor
    child(int j) : parent(j)        //Explicitly calling
    {
        y = j;
        cout << "Child class Parameterized Constructor\n";
    }
};
int main()
{
    child c(10);
    return 0;
}
```

Output

Parent class Parameterized Constructor

Child class Parameterized Constructor

Constructor call in multiple inheritance constructors

class C: public A, public B;

Constructors are called upon the order in which they are inherited

First class A constructors are executed followed by class B constructors, then class C constructors

Lab Tasks

Q1). Imagine a publishing company that markets both book and audio-cassette versions of its works. Create a class publication that stores the title and price of a publication. from this class derive two classes:

1. book, which adds a page count and
2. tape, which adds a playing time in minutes.
3. each of these three classes should have getdata() function to get its data from the user at the keyboard and a putdata() function to display its data.

Write a main() program to test the book and tape class by creating instances of them, asking the user to fill in their data with getdata() and then displaying the data with putdata().

Q2). Write a class Person that has attributes of id, name and address. It has a constructor to initialize, a member function to input and a member function to display data members. Create another class Student that inherits Person class. It has additional attributes of rollnumber and marks. It also has member function to input and display its data members.

Q3). Write a base class Computer that contains data members of wordsize(in bits), memorysize (in megabytes), storagesize (in megabytes) and speed (in megahertz). Derive a Laptop class that is a kind of computer but also specifies the object's length, width, height, and weight. Member functions for both classes should include a default constructor, a constructor to inialize all components and a function to display data members.

Home Activity

Q1). Write a program having a base class Student with data members rollno, name and Class define a member functions getdata() to input values and another function putdata() to display all values. A class Test is derived from class Student with data members T1marks, T2marks, T3marks, Sessional1, Sessional2, Assignment and Final. Also make a function getmarks() to enter marks for all variables except Final and also make a function putmarks() to display result. Make a function Finalresult() to calculate value for final variable using other marks. Then display the student result along with student data.

Q2). Write a program that declares two classes. The parent class is called Simple that has two data members num1 and num2 to store two numbers. It also has four member functions.

- The add() function adds two numbers and displays the result.
- The sub() function subtracts two numbers and displays the result.
- The mul() function multiplies two numbers and displays the result.
- The div() function divides two numbers and displays the result.

The child class is called Complex that overrides all four functions. Each function in the child class checks the value of data members. It calls the corresponding member function in the parent class if the values are greater than 0. Otherwise it displays error message.