Week 9: Lecture 17

Type of Inheritance in C++

Types of Inheritance

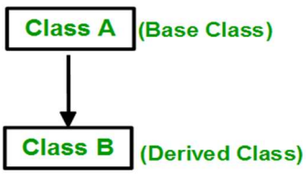
Single Inheritance: It is the inheritance hierarchy wherein one derived class inherits from one base class.

Multiple Inheritance: It is the inheritance hierarchy wherein one derived class inherits from multiple base class(es)

Hierarchical Inheritance: It is the inheritance hierarchy wherein multiple subclasses inherit from one base class.

Multilevel Inheritance: It is the inheritance hierarchy wherein subclass acts as a base class for other classes.

1. **Single Inheritance**: In single inheritance, a class is allowed to inherit from only one class. i.e. one sub class is inherited by one base class only.



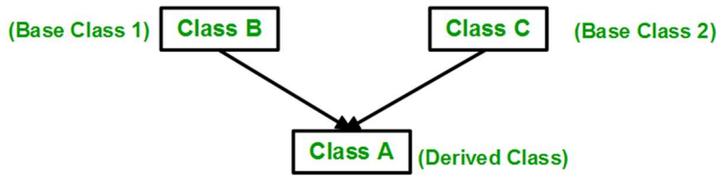
Example:

```
// C++ program to explain
// Single inheritance
#include <iostream>
using namespace std;

// base class
class Vehicle {
  public:
    Vehicle()
    {
      cout << "This is a Vehicle" << endl;
    }
};
// sub class derived from two base classes
class Car: public Vehicle{
};
// main function
int main()
{
    // creating object of sub class will
    // invoke the constructor of base classes
    Car obj;
    return 0;
}
```

Output:

```
This is a vehicle
```

2. **Multiple Inheritance:** Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes. i.e one **sub class** is inherited from more than one **base classes**.



Example:

```
class derivedclass_name : access_mode base_class1, access_mode
base_class2, ....
```

```
{
  //body of subclass
};
```

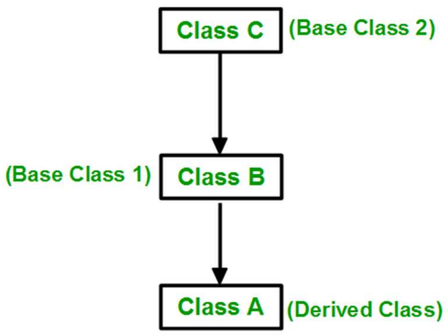
Here, the number of base classes will be separated by a comma (', ') and access mode for every base class must be specified.

```
// C++ program to explain
// multiple inheritance
#include <iostream>
using namespace std;
// first base class
class Vehicle {
  public:
    Vehicle()
    {
      cout << "This is a Vehicle" << endl;
    }
};
// second base class
class FourWheeler {
  public:
    FourWheeler()
    {
      cout << "This is a 4 wheeler Vehicle" << endl;
    }
};
// sub class derived from two base classes
class Car: public Vehicle, public FourWheeler {
};
// main function
int main()
{
    // creating object of sub class will
    // invoke the constructor of base classes
    Car obj;
    return 0;
}
```

Output:

```
This is a Vehicle
This is a 4 wheeler Vehicle
```

1. **Multilevel Inheritance**: In this type of inheritance, a derived class is created from another derived class.

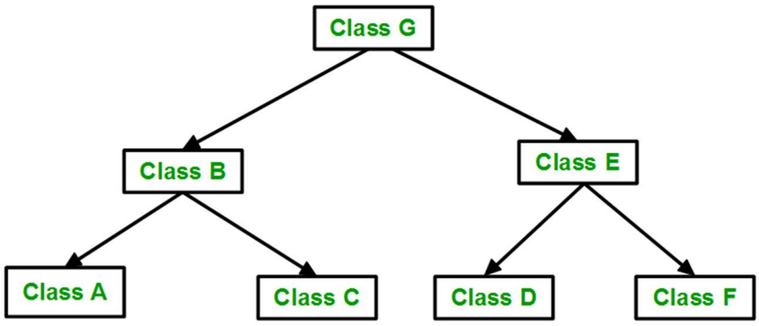



```
// C++ program to implement
// Multilevel Inheritance
#include <iostream>
using namespace std;

// base class
class Vehicle
{
  public:
    Vehicle()
    {
      cout << "This is a Vehicle" << endl;
    }
};
class fourWheeler: public Vehicle
{  public:
    fourWheeler()
    {
      cout<<"Objects with 4 wheels are vehicles"<<endl;
    }
};
// sub class derived from two base classes
class Car: public fourWheeler{
   public:
     car()
     {
       cout<<"Car has 4 Wheels"<<endl;
     }
};

// main function
int main()
{
    //creating object of sub class will
    //invoke the constructor of base classes
    Car obj;
    return 0;
}
```

Output:

```
This is a Vehicle

Objects with 4 wheels are vehicles

Car has 4 Wheels
```

1. **Hierarchical Inheritance**: In this type of inheritance, more than one sub class is inherited from a single base class. i.e. more than one derived class is created from a single base class.



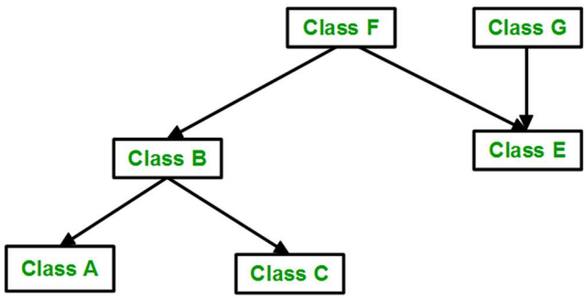

```
// C++ program to implement
// Hierarchical Inheritance
#include <iostream>
using namespace std;

// base class
class Vehicle
{
  public:
    Vehicle()
    {
      cout << "This is a Vehicle" << endl;
    }
};

// first sub class
class Car: public Vehicle
{

};

// second sub class
class Bus: public Vehicle
{

};

// main function
int main()
{
    // creating object of sub class will
    // invoke the constructor of base class
    Car obj1;
    Bus obj2;
    return 0;
}
```

Output:

```
This is a Vehicle
This is a Vehicle
```

2. **Hybrid (Virtual) Inheritance**: Hybrid Inheritance is implemented by combining more than one type of inheritance. For example: Combining Hierarchical inheritance and Multiple Inheritance.

   Below image shows the combination of hierarchical and multiple inheritance:




```
// C++ program for Hybrid Inheritance
#include <iostream>
using namespace std;
// base class
class Vehicle
{
  public:
    Vehicle()
    {
      cout << "This is a Vehicle" << endl;
    }
};
//base class
class Fare
{
    public:
    Fare()
    {
        cout<<"Fare of Vehicle\n";
    }
};
// first sub class
class Car: public Vehicle
{
};
// second sub class
class Bus: public Vehicle, public Fare
{
};
// main function
int main()
{
    // creating object of sub class will
    // invoke the constructor of base class
    Bus obj2;
    return 0;
}
```

Output:

```
This is a Vehicle
Fare of Vehicle
```