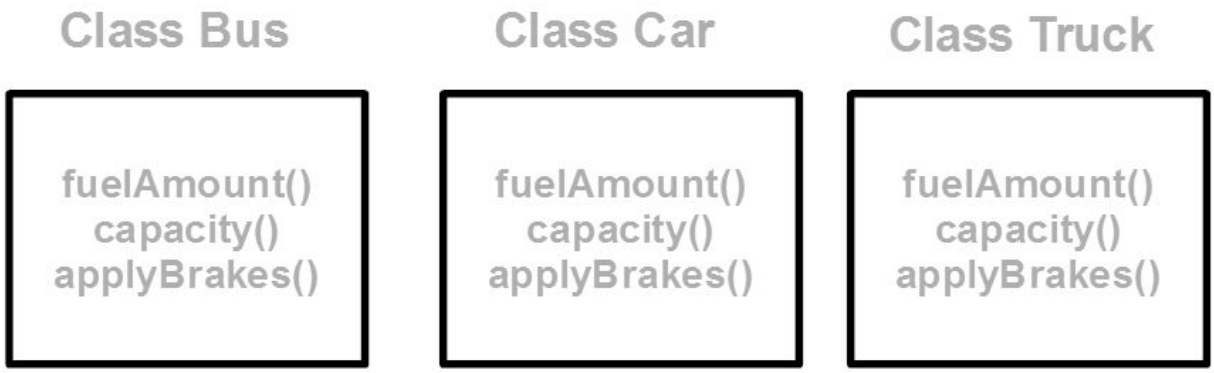
Inheritance in C++

The capability of a class to derive properties and characteristics from another class is called Inheritance. Inheritance is one of the most important feature of Object Oriented Programming.
Sub Class: The class that inherits properties from another class is called Sub class or Derived Class.

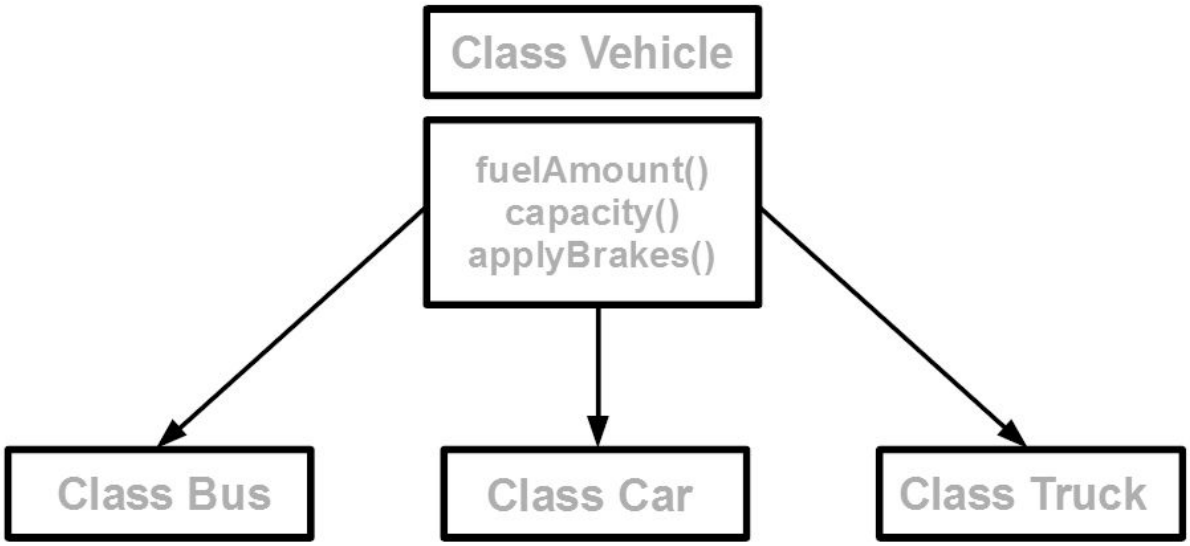
Super Class:
The class whose properties are inherited by sub class is called Base Class or Super class.

Why and when to use inheritance?

Consider a group of vehicles. You need to create classes for Bus, Car and Truck. The methods fuelAmount(), capacity(), applyBrakes() will be same for all of the three classes. If we create these classes avoiding inheritance then we have to write all of these functions in each of the three classes as shown in below figure:



You can clearly see that above process results in duplication of same code 3 times. This increases the chances of error and data redundancy. To avoid this type of situation, inheritance is used. If we create a class Vehicle and write these three functions in it and inherit the rest of the classes from the vehicle class, then we can simply avoid the duplication of data and increase re-usability. Look at the below diagram in which the three classes are inherited from vehicle class:



Using inheritance, we have to write the functions only one time instead of three times as we have inherited rest of the three classes from base class(Vehicle).
Implementing inheritance in C++: For creating a sub-class which is inherited from the base class we have to follow the below syntax.

Syntax:

```
class subclass_name : access_mode base_class_name
{
  //body of subclass
};
```

Here, subclass_name is the name of the sub class, access_mode is the mode in which you want to inherit this sub class for example: public, private etc. and base_class_name is the name of the base class from which you want to inherit the sub class.
Note: A derived class doesn't inherit access to private data members. However, it does inherit a full parent object, which contains any private members which that class declares.

In the above program the 'Child' class is publicly inherited from the 'Parent' class so the public data members of the class 'Parent' will also be inherited by the class 'Child'.

Modes of Inheritance

Public mode: If we derive a sub class from a public base class. Then the public member of the base class will become public in the derived class and protected members of the base class will become protected in derived class.

Protected mode: If we derive a sub class from a Protected base class. Then both public member and protected members of the base class will become protected in derived class.

Private mode: If we derive a sub class from a Private base class. Then both public member and protected members of the base class will become Private in derived class.

Note : The private members in the base class cannot be directly accessed in the derived class, while protected members can be directly accessed. For example, Classes B, C and D all contain the variables x, y and z in below example. It is just question of access.

The below table summarizes the above three modes and shows the access specifier of the members of base class in the sub class when derived in public, protected and private modes:

Base class member access specifier	Type of Inheritence		
	Public	Protected	Private
Public	Public	Protected	Private
Protected	Protected	Protected	Private
Private	Not accessible (Hidden)	Not accessible (Hidden)	Not accessible (Hidden)

In C++, the class which inherits the members of another class is called derived class and the class whose members are inherited is called base class. The derived class is the specialized class for the base class.

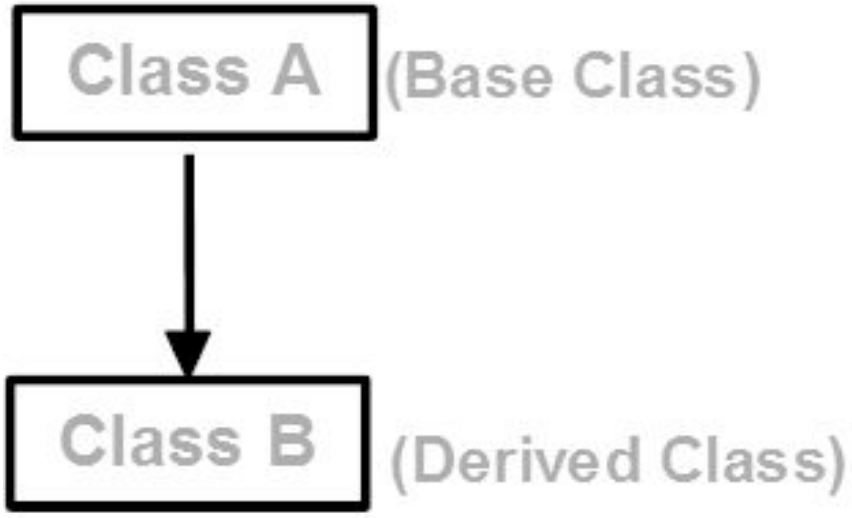
Advantage of C++ Inheritance

Code reusability: Now you can reuse the members of your parent class. So, there is no need to define the member again. So less code is required in the class.

Types of Inheritance in C++

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.



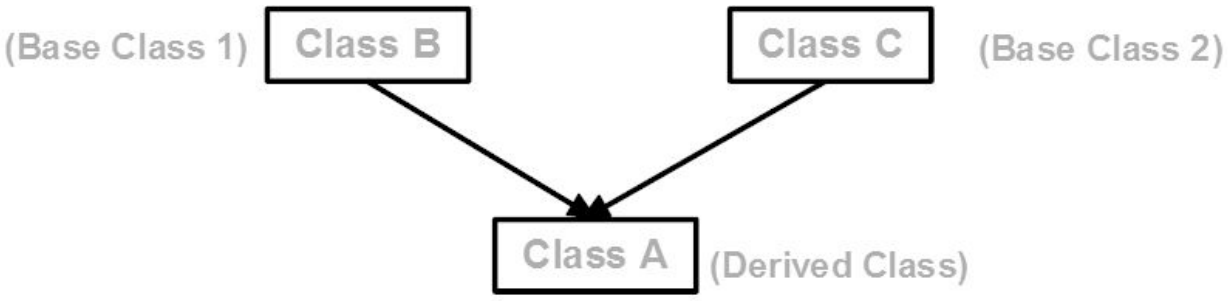
Syntax:

```
class subclass_name : access_mode base_class
{
  //body of subclass
};
```

Single Inheritance in C++ Programming: The class which inherits the properties of another class is called Derived or Child or Sub class and the class whose properties are inherited is called Base or Parent or Super class. When a single class is derived from a single parent class, it is called Single inheritance.

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.



Syntax:

class subclass_name : access_mode base_class1, access_mode base_class2, ...

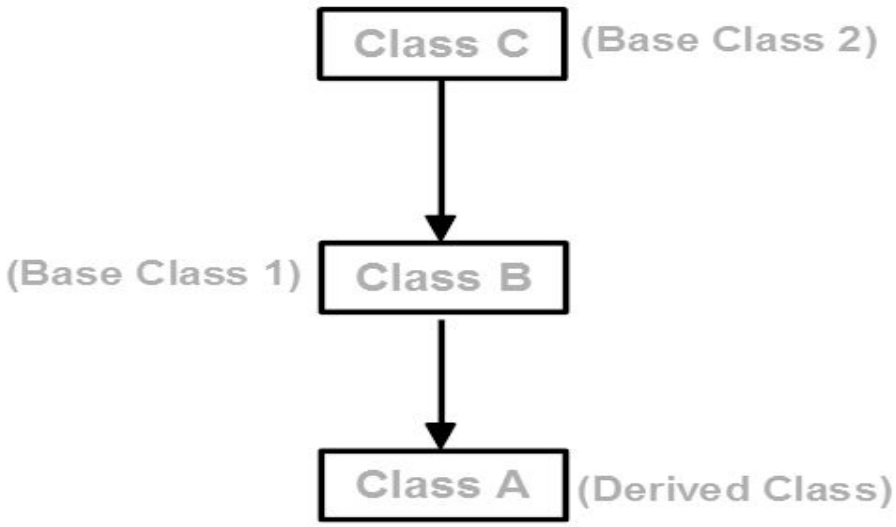
{

```
  //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.

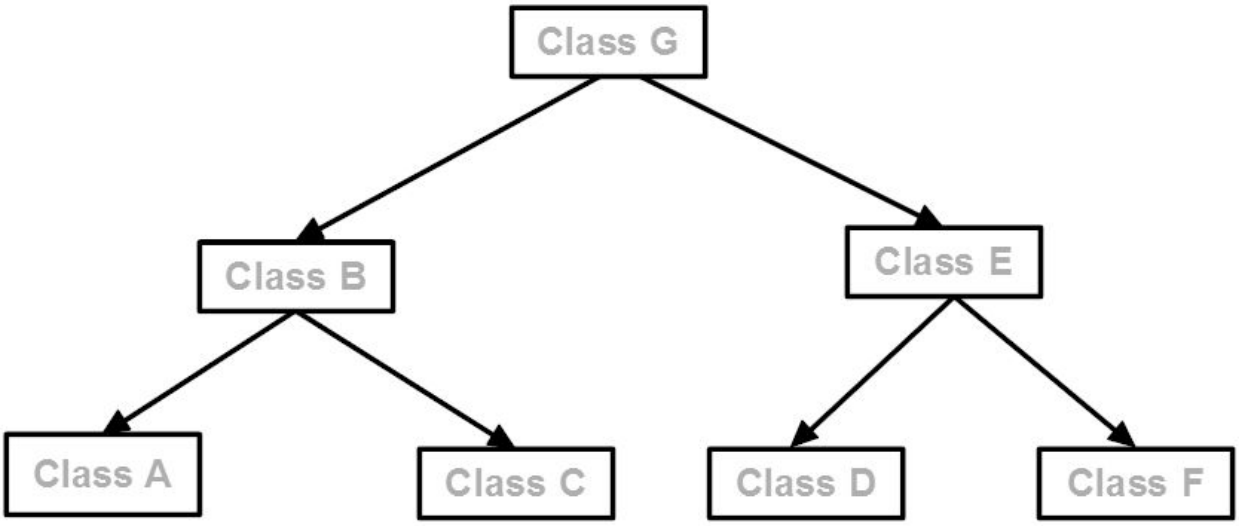
Multilevel Inheritance:

In this type of inheritance, a derived class is created from another derived class.



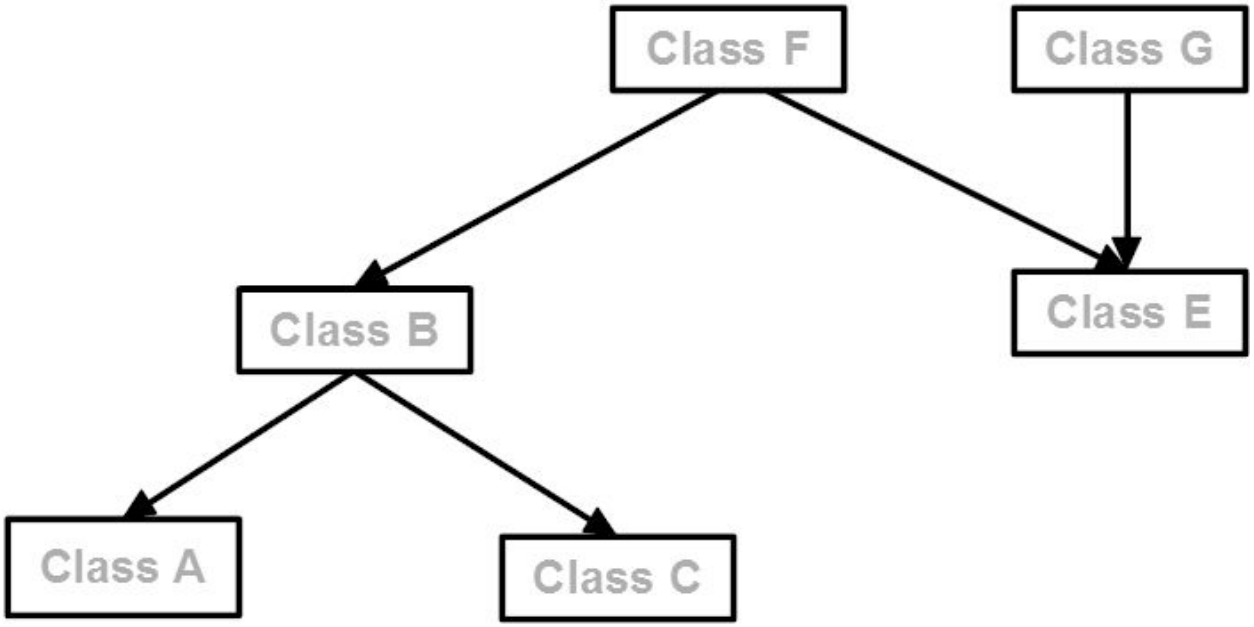
Hierarchical Inheritance:

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.



Hybrid (Virtual) Inheritance:

Below image shows the combination of hierarchical and multiple inheritance:



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

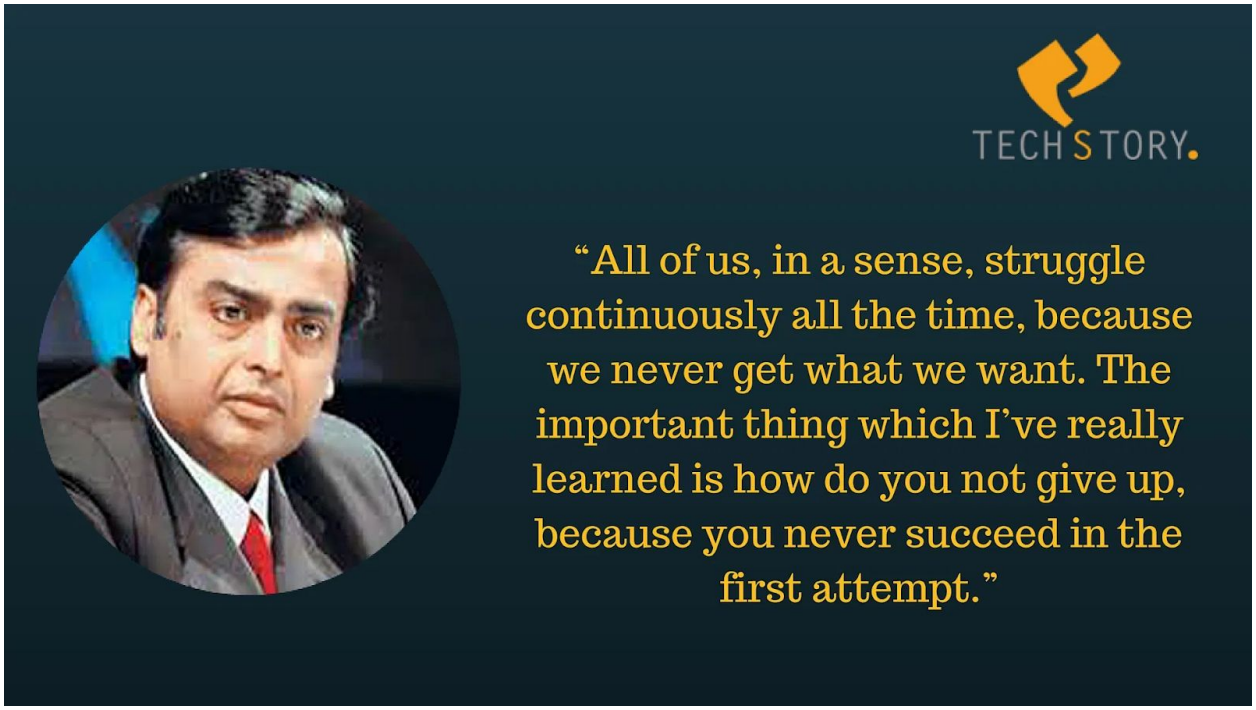
EXTERNAL RESOURCES :

https://www.tutorialspoint.com/cplusplus/cpp_inheritance.htm

https://www.javatpoint.com/cpp-inheritance

https://www.programiz.com/cpp-programming/inheritance

https://www.learncpp.com/cpp-tutorial/112-basic-inheritance-in-c/



NeXt LIVE : POLYMORPHISM