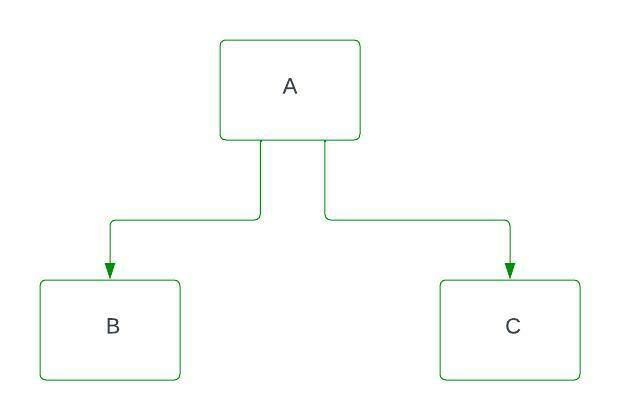
# **Hierarchical Inheritance**

Inheritance is a feature of Object-Oriented-programming in which a derived class (child class) inherits the property (data member and member functions) of the Base class (parent class). For example, a child inherits the traits of their parents.

In Hierarchical inheritance, more than one sub-class inherits the property of a single base class. There is one base class and multiple derived classes. Several other classes inherit the derived classes as well. Hierarchical structures thus form a tree-like structure. It is similar to that, mango and apple both are fruits; both inherit the property of fruit. Fruit will be the Base class, and mango and apple are sub-classes.

The below diagram shows, Class A is a Base class, B is a subclass inherited from class A, and C is a subclass it also inherits from class A.



**Syntax:**

Class A

{

............

};

Class B: access\_specifier A

{

.........

};

Class C: access\_specifier A

{

.............

};

// C++ program for Hierarchical Inheritance

#include<iostream>

using namespace std;

class A //superclass A

{

public:

void show\_A() {

cout<<"class A"<<endl;

}

};

class B : public A //subclass B

{

public:

void show\_B() {

cout<<"class B"<<endl;

}

};

class C : public A //subclass C

{

public:

void show\_C() {

cout<<"class C"<<endl;

}

};

int main() {

B b; // b is object of class B

cout<<"calling from B: "<<endl;

b.show\_B();

b.show\_A();

C c; // c is object of class C

cout<<"calling from C: "<<endl;

c.show\_C();

c.show\_A();

return 0;

}

**Output**

calling from B:

class B

class A

calling from C:

class C

class A