Jaydeep Shah
radhey04ec@gmail.com
<Self learning notes>

TUTORIAL 9 : INHERITANCE

It is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:

derived class (child) - the class that inherits from another class

base class (parent) - the class being inherited from

Inheritance means:
Sharing the information,
Reuse old code / logic

To inherit from a class, use the : symbol.

Inheritance is way to use code again and again instead of creation / code reuse.

How use?

Class “Child class name” : access_specifier “parent class name”

Example:

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


//Parent class
class car
{

public:
   string brand;

   void honk_sound()
   {
      cout << "\n Tun Tun ";
```

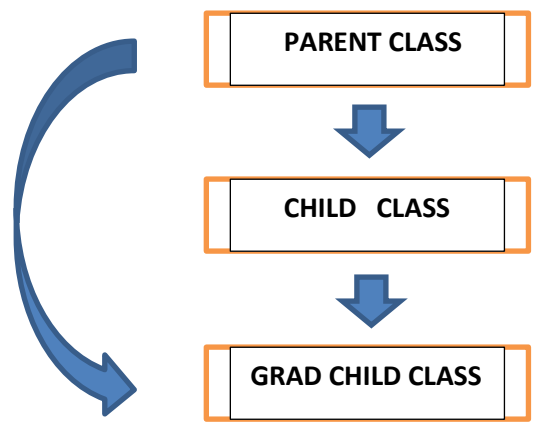
```
    }

};

//child class  (child class name : access_specifier parent class name)
class vehicle : public car{
public:
string model_name = "TATA";  //Default
};


int main()
{
  vehicle a1;   //a1 is child class object but possible to access all parent class object
  cout<< "Model default = "<<a1.model_name;
  a1.honk_sound();


   return 0;
}
```

class vehicle : public car : This statement said that class vehicle is derived from parent class car,and we are going to use parent class member/method in this derived class as a public.

Multilevel Inheritance:

When class derived from another class, but another class also not original class, that was also derived from some other class.

My Grand child → Derived from Child class → Parent Class



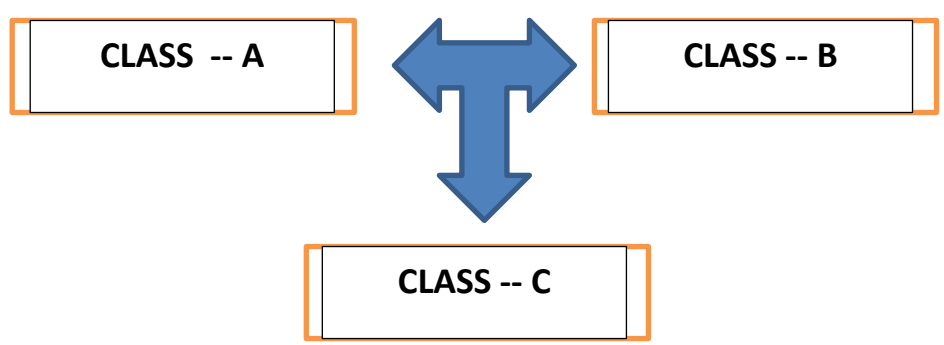
Example:

```
#include<iostream>

using namespace std;

//Parent class

class parent_class

{

public:

   void parent_fun()

   {

      cout << "Parent Function call";

   }

};


//Child class

class child : public parent_class

{
```

```
    public:
};

//Grandchild class
class grand_child : public child
{
   public:
};

int main()
{
//Create object of grand child class
grand_child gc;
gc.parent_fun();  //call function of parent class function from grand child
    return 0;
}
```

As you can see in above example you can call parent class function from grand_child class.

Multilevel Inheritance with more than one base class:



Example:

```
#include<iostream>


using namespace std;


//Class (1)
class class_one
{
public:
   void class_one_func()
   {
      cout << "Class one function \n";
   }
};


//Class (2)
class class_two
{
public:
   void class_two_func()
   {
      cout << "Class two function \n";
   }

};
```

```
//Derived class
class derived_clas : public class_one,public class_two
{

public:
   void derived_class_func()
   {

      cout <<"Derived class function \n";
   }

};
int main()
{

//Create object
derived_clas derived_obj;
//Func 1 call-
derived_obj.derived_class_func();
//Func 2 call -
derived_obj.class_two_func();
//Func 3 call -
derived_obj.class_one_func();
   return 0;
}
```

OUTPUT:

Derived class function

Class two function

Class one function

Access specifier:

Until now, we have only used public (members of a class are accessible from outside the class) and private (members can only be accessed within the class). **The third specifier, protected, is similar to private, but it can also be accessed in the inherited class**:

```
// Base class
class Employee {
  protected: // Protected access specifier
    int salary;
};
// Derived class
class Programmer: public Employee {
  public:
    int bonus;
    void setSalary(int s) {
      salary = s;
    }
    int getSalary() {
      return salary;
    }
};

int main() {
```

```
  Programmer myObj;
  myObj.setSalary(50000);
  myObj.bonus = 15000;
  cout << "Salary: " << myObj.getSalary() << "\n";
  cout << "Bonus: " << myObj.bonus << "\n";
  return 0;
}
```

Note:

Private member not accessible from other class

Protected member is accessible from other class using “**GET/SET Method**”