Inheritance

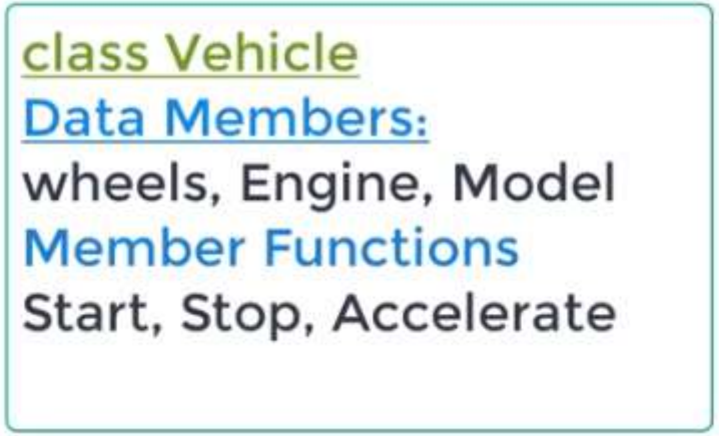
Inheritance

A programming technique that is used to build a new class using existing class.

Inheritance is one of the key features of Object-oriented programming in C++. It allows us to create a new class (derived class) from an existing class (base class).

The derived class inherits the features from the base class and can have additional features of its own.

Example



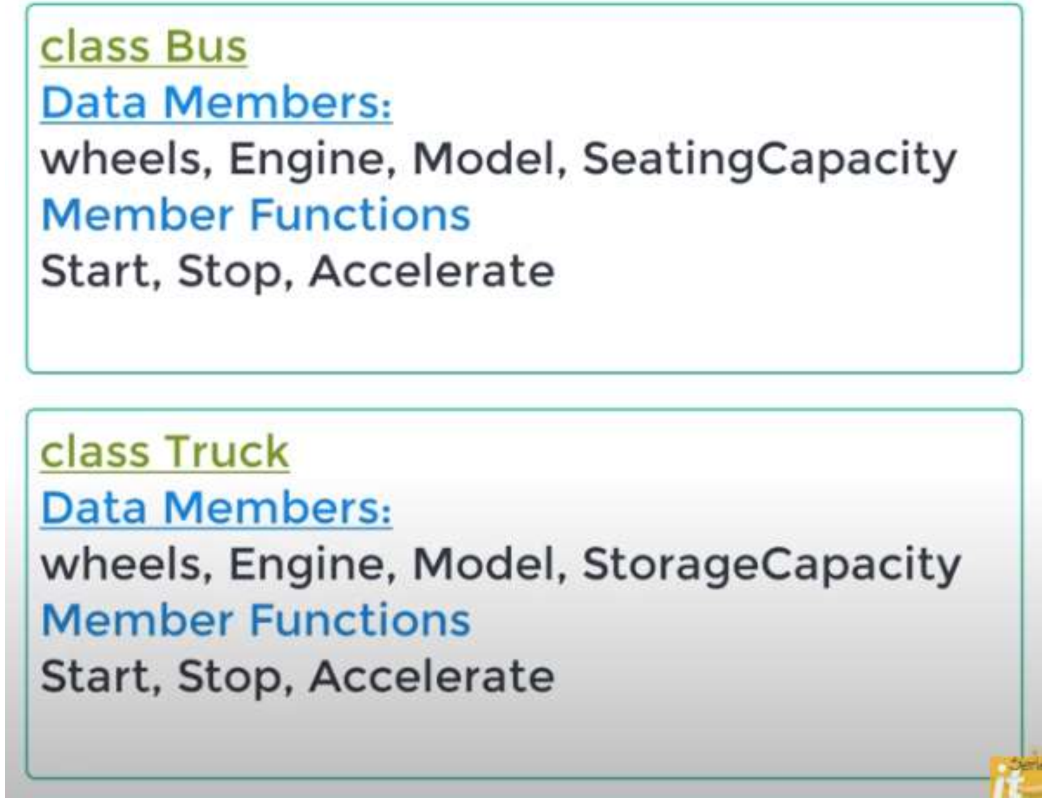
class Vehicle

Data Members:

wheels, Engine, Model

Member Functions

Start, Stop, Accelerate



class Bus

Data Members:

wheels, Engine, Model, SeatingCapacity

Member Functions

Start, Stop, Accelerate

class Truck

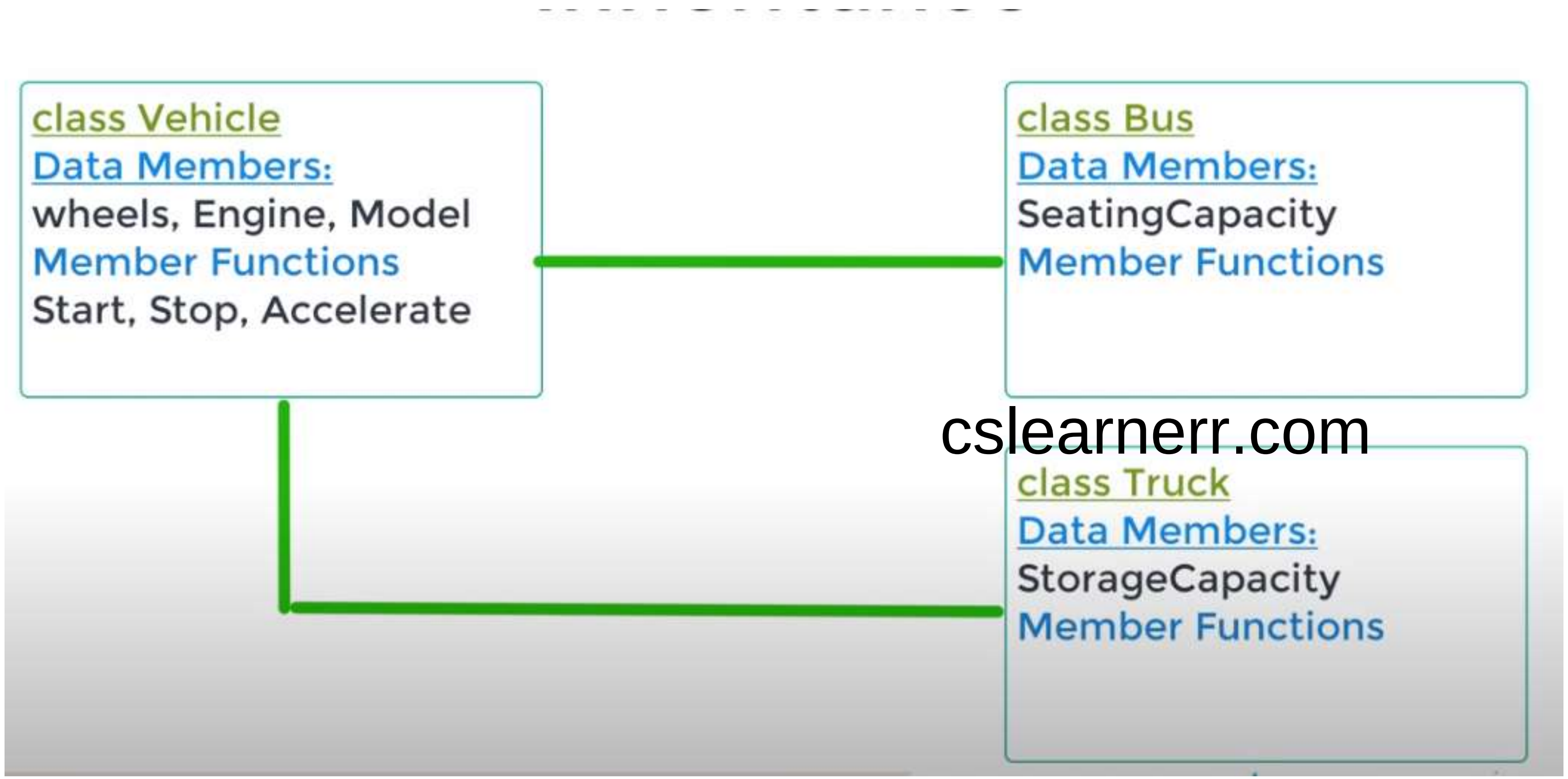
Data Members:

wheels, Engine, Model, StorageCapacity

Member Functions

Start, Stop, Accelerate

Example



class Vehicle

Data Members:

wheels, Engine, Model

Member Functions

Start, Stop, Accelerate

class Bus

Data Members:

SeatingCapacity

Member Functions

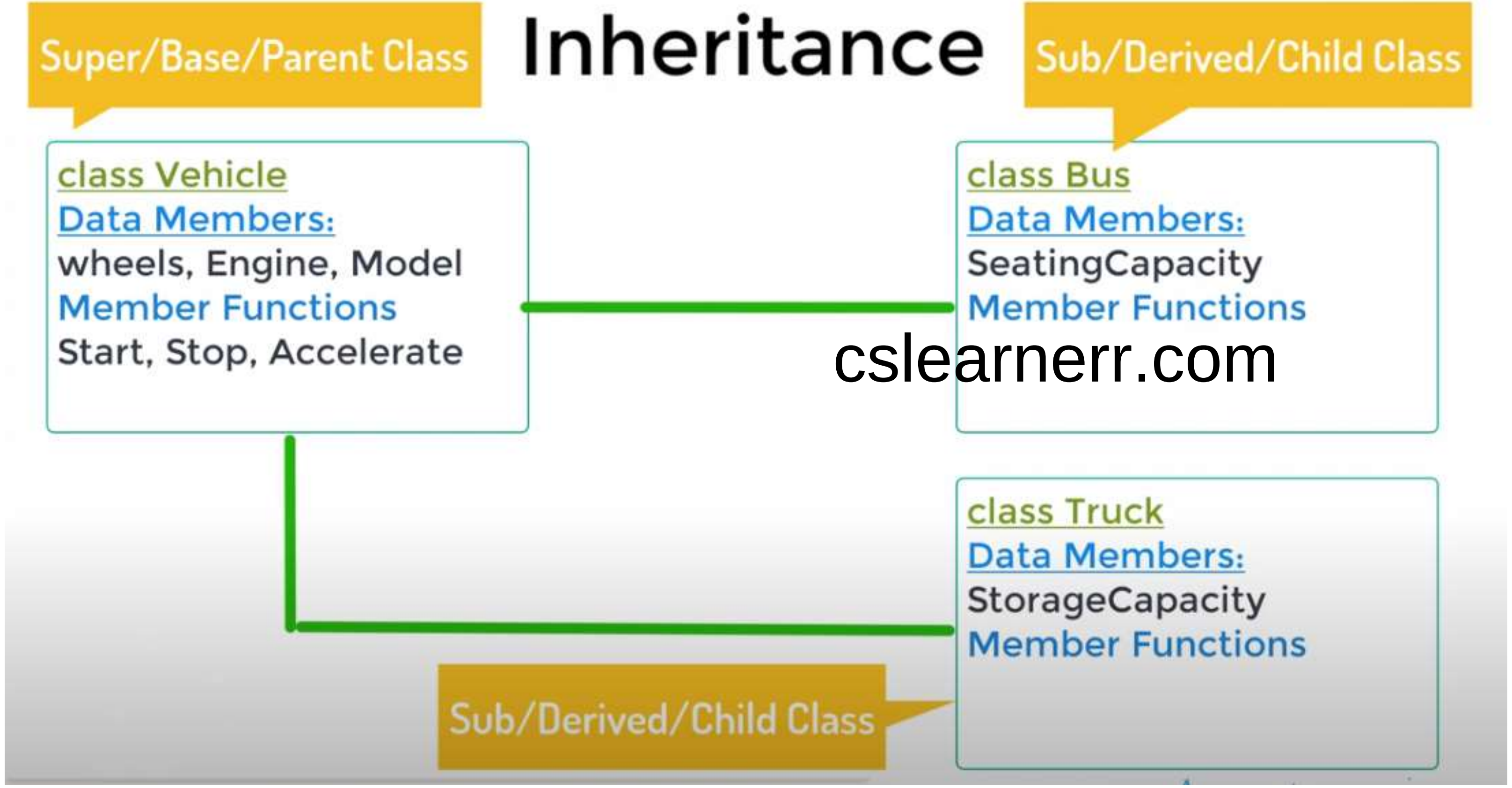
class Truck

Data Members:

StorageCapacity

Member Functions

Example



Super/Base/Parent Class

Inheritance

Sub/Derived/Child Class

class Vehicle
Data Members:
wheels, Engine, Model
Member Functions
Start, Stop, Accelerate

class Bus
Data Members:
SeatingCapacity
Member Functions

class Truck
Data Members:
StorageCapacity
Member Functions

Sub/Derived/Child Class

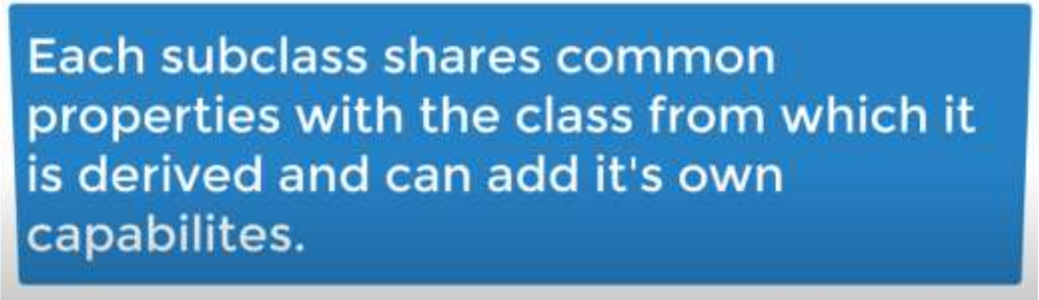
Inheritance

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

To inherit a class , use the : symbol



Each subclass shares common properties with the class from which it is derived and can add it's own capabilites.

Categories of Inheritance

Single Inheritance :

- A child class is derived from single parent class.

Multiple Inheritance:

- A child class is derived from multiple parent class.

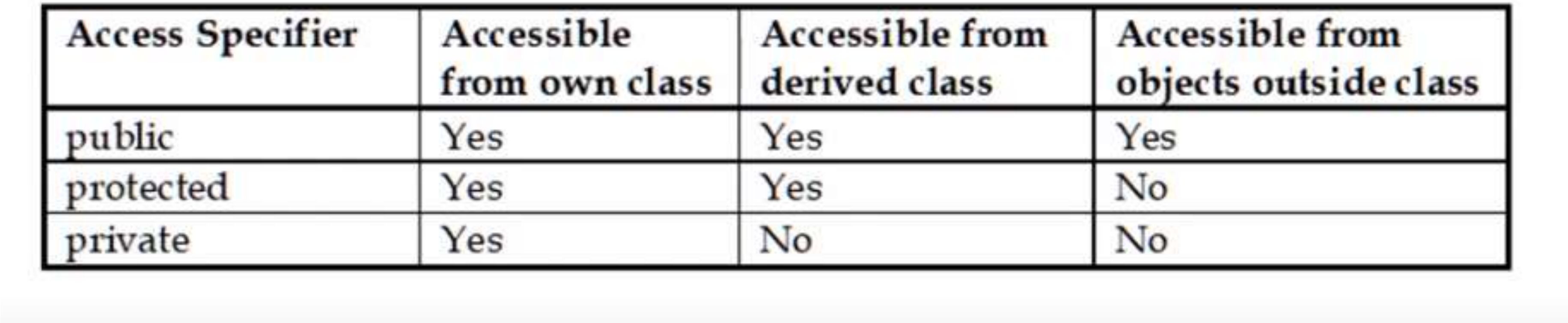
Access specifier

Public : The public data member of a class is accessible from anywhere in the program .

Private: the private data member of the class is only accessible in the class in which they are declared.

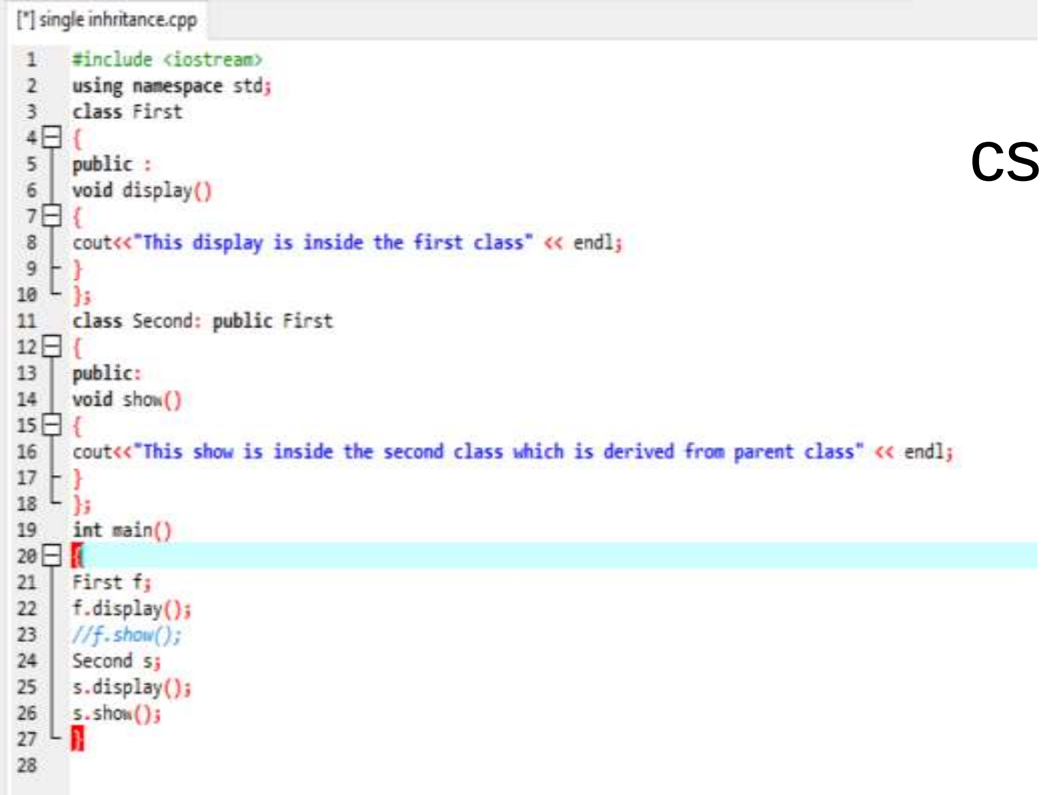
Protected: Data member declared with private access specifier cannot be accessed in a child class. Here to overcome this issue protected access specifiers ae used. Protected data members can be accessed in child classes as well.

Hierarchy

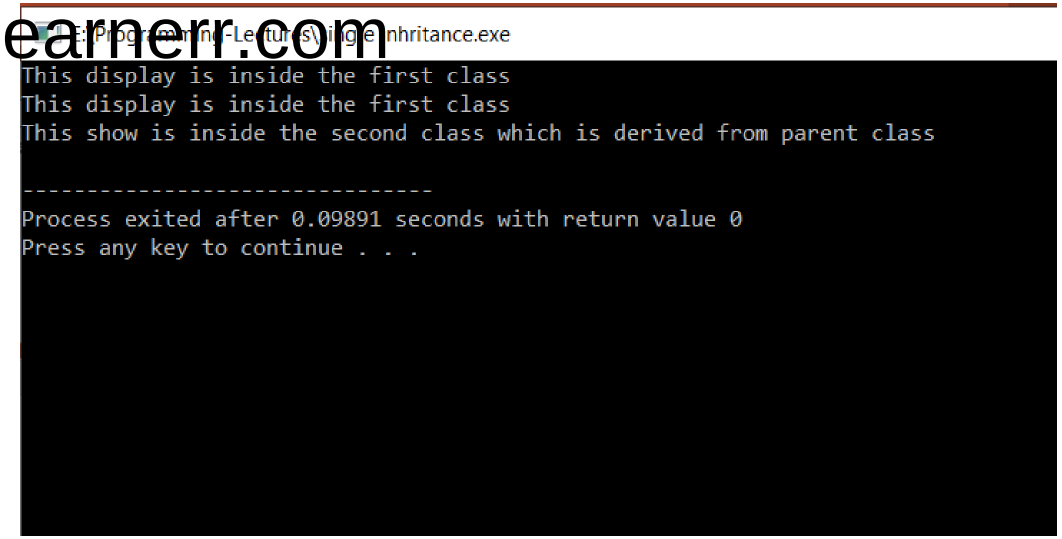


Access Specifier	Accessible from own class	Accessible from derived class	Accessible from objects outside class
public	Yes	Yes	Yes
protected	Yes	Yes	No
private	Yes	No	No

Single Inheritance

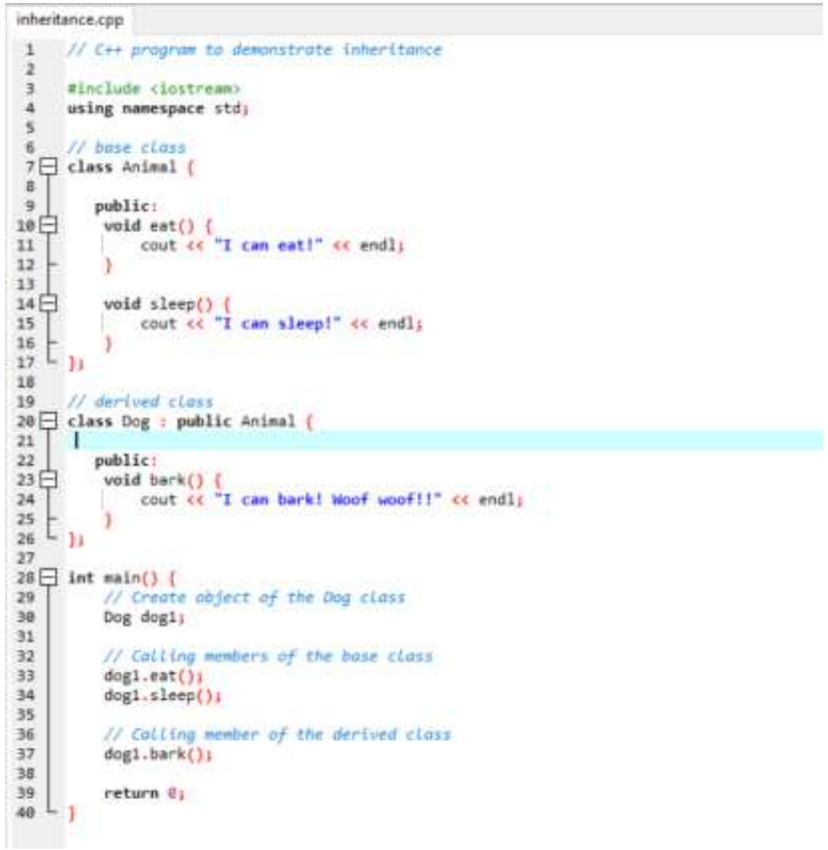


```
#include <iostream>
using namespace std;
class First
{
public :
void display()
{
cout<<"This display is inside the first class" << endl;
}
};
class Second: public First
{
public:
void show()
{
cout<<"This show is inside the second class which is derived from parent class" << endl;
}
};
int main()
{
First f;
f.display();
//f.show();
Second s;
s.display();
s.show();
}
```

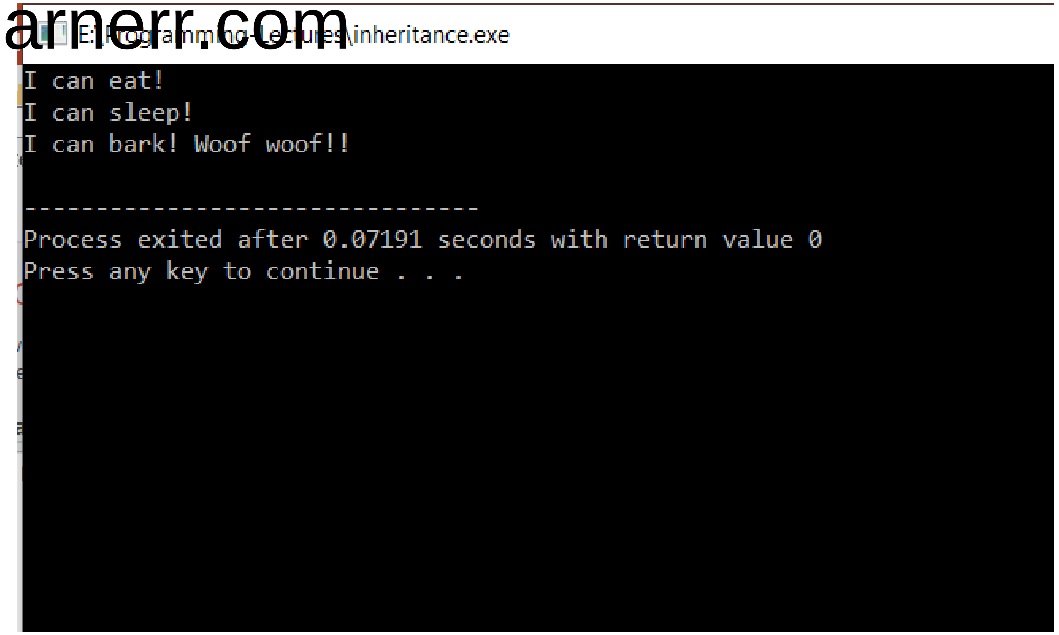


```
This display is inside the first class
This display is inside the first class
This show is inside the second class which is derived from parent class

--------------------------------
Process exited after 0.09891 seconds with return value 0
Press any key to continue . . .
```

Example



```cpp
// C++ program to demonstrate inheritance

#include <iostream>
using namespace std;

// base class
class Animal {

   public:
    void eat() {
        cout << "I can eat!" << endl;
    }

    void sleep() {
        cout << "I can sleep!" << endl;
    }
};

// derived class
class Dog : public Animal {

   public:
    void bark() {
        cout << "I can bark! Woof woof!!" << endl;
    }
};

int main() {
    // Create object of the Dog class
    Dog dog1;

    // Calling members of the base class
    dog1.eat();
    dog1.sleep();

    // Calling member of the derived class
    dog1.bark();

    return 0;
}
```



```
I can eat!
I can sleep!
I can bark! Woof woof!!

--------------------------------
Process exited after 0.07191 seconds with return value 0
Press any key to continue . . .
```