Calculate inheritance in C

One of the most important concepts in object-oriented programming is Inheritance. Inheritance allows us to define a class in terms of another class, which makes it easier to create and maintain an application. This also provides an opportunity to reuse code features and faster execution time.

One of the most important concepts in object-oriented programming is Inheritance. Inheritance allows us to define a class in terms of another class, which makes it easier to create and maintain an application. This also provides an opportunity to reuse code features and faster execution time.

When creating a class, instead of writing all data members and new member functions, programmers may inherit the members of an existing class. This existing class is called **Base Class - the base** class, and the new class is treated as **Derived Class - the inheritance class.**

The idea of ??inheritance calculates the **IS-A** relationship (Being One). For example, mammal **IS A** animal, **IS-A** mammal dog, so the **IS-A** animal is animal, and .

Base Class and Inheritance (Derived Class) in C

A class can be inherited from more than one other class, that is, it can inherit data and functions from multiple base classes or interfaces.

The syntax for creating inheritance classes in C # is:

```
class   { . } class   :   { . }
```

Considering a base class Shape and inheritance Rectangle later: create 3 classes named **Shape, HinhChuat and TestCsharp** in turn:

The **Shape** class is the base class

```
dùng System;

QTMCsharp namespace
{
class Shape
{

protected int chieu_rong;
protected int chieu_cao;
public void setChieuRong (int w)
{
chieu_rong = w;
```

```
}
public void setChieuCao (int h)
{
chieu_cao = h;
}
}
}
```

The **HinhChuNhat** class is the inheritance class

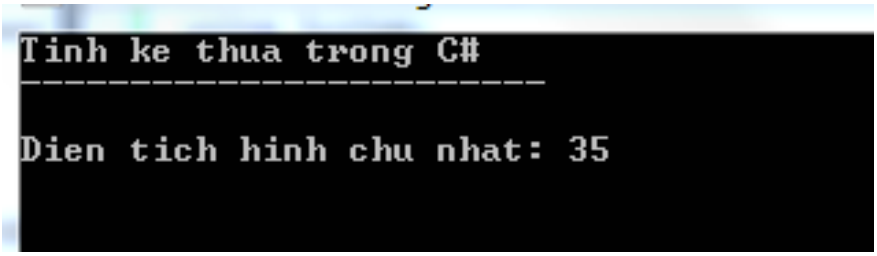
```
dùng System;

QTMCsharp namespace
{
class HinhChuNhat: Shape
{
public int TinhTich ()
{
return (chieu_cao * chieu_rong);
}
}
}
```

The **TestCsharp** class contains the main () method to manipulate the HinhChat object

```
dùng System;
QTMCsharp namespace
{
public class TestCsharp
{
public static void Main (string [] args)
{
Console.WriteLine ("Crash in C #");
Console.WriteLine ("------------------------ n");

// create doi tuong HinhChuNhat
HinhChuNhat hcn = new HinhChuNhat ();

hcn.setChieuRong (5);
hcn.setChieuCao (7);

// Print your phone book.
Console.WriteLine ("Standard configuration: {0}", hcn.tinhDienTich ());

Console.ReadKey ();
}
}
}
```

If you do not use the **Console.ReadKey ()** command; then the program will run and finish (so fast that you can not see the results). This command allows us to see the results more clearly.

Compiling and running the above C # program will produce the following results:



Initialize Base Class in C

The derived class (Derived Class) in C # inherits member variables and member methods from the base class. Therefore, the parent class object should be created before the subclass is created. You can provide directives to initialize the subclass in the member initialization list.

The following example program illustrates how to initialize Base Class in C #: create 3 classes named **HinhChuNhat, ChiPhiXayDung, TestCsharp** as follows:

The **HinhChuNhat** class is the base class

```
dùng System;

QTMCsharp namespace
{
class HinhChuNhat
{
// members of the staff
protected double chieu_dai;
protected double chieu_rong;
// constructor
public HinhChuNhat (double l, double w)
{
chieu_dai = l;
chieu_rong = w;
}
// phuong thuc
public double planet ()
{
return chieu_dai * chieu_rong;
}

public void Display ()
{
Console.WriteLine ("Chatter: {0}", chieu_dai);
Console.WriteLine ("Junk: {0}", chieu_rong);
Console.WriteLine ("Connection: {0}", TinhDienTich ());
}
}
}
```

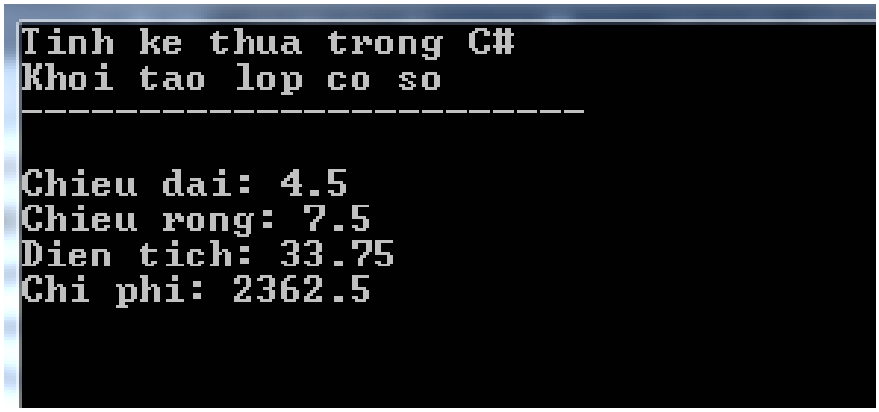
Class **ChiPhiXay** Inherits class **HinhChuNhat**

```
dùng System;

QTMCsharp namespace
{
class ChiPhiXayDung: HinhChuNhat
{
private double cost;
public ChiPhiXayDung (double l, double w): base (l, w)
{}
public double planetChiPhi ()
{
double chi_phi;
chi_phi = tinhDienTich () * 70;
return chi_phi;
}
public void hienThiThongTin ()
{
base.Display ();
Console.WriteLine ("Cost: {0}", crystalChiPhi ());
}
}
}
```

The **TestCsharp** class contains the **main** () method to manipulate the **ChiPhiXayDung** object

```
dùng System;
QTMCsharp namespace
{
public class TestCsharp
{
public static void Main (string [] args)
{
Console.WriteLine ("Crash in C #");
Console.WriteLine ("Download and install");
Console.WriteLine ("------------------------ n");
// tao doi tuong ChiPhiXayDung
ChiPhiXayung t = new ChiPhiXayDung (4.5, 7.5);
t.hienThiThongTin ();
Console.ReadLine ();

Console.ReadKey ();
}
}
}
```

Compiling and running the above C # program will produce the following results:



Multiple inheritance in C

C # does not support multiple inheritance. However, you can use Interface to implement multiple inheritance. The following example illustrates how to use Interface to deploy multiple inheritance in C #: we create 2 classes named Shape, HinhChatat, TestCsharp and an interface named ChiPhiSon as follows:

The **Shape** class is the base class

```
dùng System;

QTMCsharp namespace
{
class Shape
{

protected int chieu_rong;
protected int chieu_cao;
public void setChieuRong (int w)
{
chieu_rong = w;
}
public void setChieuCao (int h)
{
chieu_cao = h;
}
}
}
interface ChiPhiSon

dùng System;

QTMCsharp namespace
{
public interface ChiPhiSon
{
int tinhChiPhi (int dien_tich);
}
}
```

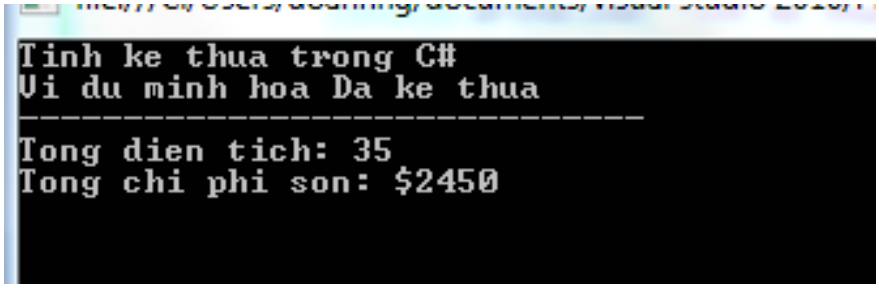
The class **HinhChuNhat** is the class that inherits the **Shape** class and the **ChiPhiSon interface**

```
dùng System;

QTMCsharp namespace
{
class HinhChuat: Shape, ChiPhiSon
{
public int TinhTich ()
{
return (chieu_rong * chieu_cao);
}
public int tinhChiPhi (int dien_tich)
{
return dien_tich * 70;
}
}
}
```

The **TestCsharp** class contains the main () method to manipulate the **HinhChat** object

```
dùng System;
QTMCsharp namespace
{
public class TestCsharp
{
public static void Main (string [] args)
{
Console.WriteLine ("Crash in C #");
Console.WriteLine ("Da ke loser");
Console.WriteLine ("------------------------------");
// create doi tuong HinhChuNhat
HinhChuNhat hcn = new HinhChuNhat ();
int dien_tich;
hcn.setChieuRong (5);
hcn.setChieuCao (7);
dien_tich = hcn.tinhDienTich ();

// print and print.
Console.WriteLine ("Tongue: {0}", hcn.tinhDienTich ());
Console.WriteLine ("Tong read: 0", hcn.tinhChiPhi (dien_tich));
Console.ReadLine ();

Console.ReadKey ();
}
}
}
```

Compiling and running the above C # program will produce the following results:



Follow tutorialspoint

Previous lesson: Class (Class) in C #

Next article: Polymorphism in C #

You finished reading the article "**Calculate inheritance in C #**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.