例:程序class\_test2

// Copyright 2016.刘珅珅

// author：刘珅珅

// 类和方法

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace class\_test2

{

class MyClass

{

private int alpha;

private int beta;

public MyClass(int i, int j)

{

alpha = i;

beta = j;

}

// MyClass类型的引用作为参数

public bool SameAs(MyClass obj)

{

if ((obj.alpha == alpha) && (obj.beta == beta))

{

return true;

}

return false;

}

public void Copy(MyClass obj)

{

alpha = obj.alpha;

beta = obj.beta;

}

public void Show()

{

Console.WriteLine("alpha: {0}, beta: {1}", alpha, beta);

}

}

class ClassTest

{

static void Main(string[] args)

{

MyClass obj1 = new MyClass(4, 5);

MyClass obj2 = new MyClass(6, 7);

Console.Write("obj1 : ");

obj1.Show();

Console.Write("obj2 : ");

obj2.Show();

obj1.Copy(obj2);

if (obj1.SameAs(obj2))

{

Console.WriteLine("obj1 and obj2 have the same values");

}

else

{

Console.WriteLine("obj1 and obj2 have different values");

}

}

}

}

C#中传递参数的方式主要有两种：与C++类似

1. 按值传递

将实参的值复制给形参，在函数中改变形参的值对实参没有影响。

1. 引用传递（类似于C++中的指针传递）

将实参的引用传递给形参。引用本身是通过值传递完成的。因此，将建立引用的副本。在函数中改变形参不会影响实参（参考C++，在函数中改变指针本身，不会影响实参指针的指向），然而，改变形参所指向的对象会影响到实参所指向的对象（这一点也同C++中的指针传递类似）。

例:程序class\_test3

// Copyright 2016.刘珅珅

// author：刘珅珅

// 值传递与引用传递

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace class\_test3

{

class MyClass

{

private int a, b;

public MyClass(int i, int j)

{

a = i;

b = j;

}

public void Change(MyClass obj)

{

obj.a = obj.a + obj.b;

obj.b = -obj.b;

}

public void Show()

{

Console.WriteLine(" a is " + a + " b is " + b);

}

}

class ClassTest

{

static void Main(string[] args)

{

MyClass obj = new MyClass(15, 20);

Console.Write("Before change: ");

obj.Show(); // 15 20

obj.Change(obj);

Console.Write("After change: ");

obj.Show(); // 35 -20

}

}

}