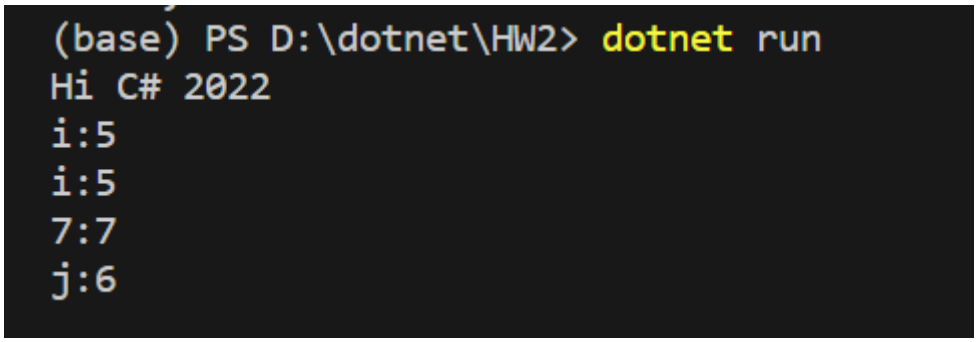
HW2 2150276 沈卓成

1

Code:

```
using System;
public class FirstClass
{
    static void Main()
    {
        int i = 5;
        object j = 6;
        Console.WriteLine("Hi C# 2022");
        Console.WriteLine("i:{0}", i);
        Console.WriteLine("i:{0}", i.ToString());
        Console.WriteLine("7:{0}", 7.ToString());
        Console.WriteLine("j:{0}", j);
        Console.ReadLine();
}
}
```

Ans:



2

Code:

```
using System;
using System.Security.Authentication;
public class SomeType
{
    class SomeNestType
    {
        public void f() { Console.WriteLine("f"); }
    }
    const Int32 SomeConstant = 1000;
```

```
    public readonly Int32 SomereadOnlyFiled = 2;
    static Int32 SomeReadWriteFiled = 3;
    static SomeType()
    {
        Console.WriteLine("Static SomeType");
    }
    public SomeType()
    {
        Console.WriteLine("Inst SomeType");
        SomereadOnlyFiled = 100;
    }
    public SomeType(Int32 x) { }
    ~SomeType()
    {
        Console.WriteLine("~SomeType");
    }
    int II
    {
        get; set;
    }
    int F
    {
    get { return SomereadOnlyFiled; }
    }
    public int this[int v]
    {
    get
    {
        int[] bb = { 55, 33, 44 };
        return bb[v];
    }
}
public event EventHandler SomeEvent;
    public void f()
    {
    //SomereadOnlyFiled = 1000;
    }
    public override string ToString()
    {
        return "SomeTypeToStringVal";
    }
    static void M1(Object sender, EventArgs e)
    {
        SomeType o = (SomeType)sender;
        Console.WriteLine("M1:{0}", o.SomereadOnlyFiled);
    }
    void M2(Object sender, EventArgs e)
    {
        SomeType o = (SomeType)sender;
        Console.WriteLine("M2:{0}", o.SomereadOnlyFiled);
    }
    void Trigger()
    {
        SomeEvent(this, null);
    }
    static void Main()
    {
        SomeType s = new SomeType();
```

```
        Console.WriteLine("Hi");
        SomeNestType o = new SomeNestType();
        o.f();
        Console.WriteLine("ToString {0}", s);
        Console.WriteLine("SomereadOnlyFiled {0}", s.SomereadOnlyFiled);
        Console.WriteLine("SomeReadWriteFiled {0}", SomeReadWriteFiled);
        Console.WriteLine("SomeConstant {0}", SomeConstant);
        s.MyF(300);
        s.II = 9;
        Console.WriteLine("Hi {0} {1}", s.II, s.F);
        Console.WriteLine("{0}, {1}", s[0], s[2]);
        s.SomeEvent += SomeType.M1;
        s.SomeEvent += s.M2;
        s.Trigger();
    }
}
    public static class SomeExtent
    {
    public static void MyF(this SomeType e, int nn)
    {
        Console.WriteLine("MyF {0} {1}", nn, e.SomereadOnlyFiled);
    }
}
```

Ans:

```
(base) PS D:\dotnet\HW2> dotnet run SomeType.cs
Static SomeType
Inst SomeType
Hi
f
ToString SomeTypeToStringVal
SomereadOnlyFiled 100
SomeReadWriteFiled 3
SomeConstant 1000
MyF 300 100
Hi 9 100
55, 44
M1:100
M2:100
```

3

Code:

```
using System;
class Test
{
    static void Main()
    {
        Console.WriteLine("Hello c#");
        Type t = 7.GetType();
```

```
        string s = 7.ToString();
        Console.WriteLine("Type {0} {1}", t, s);
        int i = 5;
        System.Int32 k = 6;
        uint kk = 70;
        System.Char u = 'c';
        Console.WriteLine("Type {0} {1}", u.GetType(), kk.GetType());
        Console.WriteLine("Type {0} {1}", i, k);
        int[] numbers = { 1, 2, 3, 4, 5 };
        // numbers.Append(66);
        Console.WriteLine("numbers {0} {1}", numbers.Length, numbers[3]);
        const int nArray = 6;
        int[] arr = new int[nArray];
        int[] arr2 = new int[3] { 1, 2, 3 };
        int[] arr3 = new int[nArray] { 1, 2, 3, 4, 5, 6 };
        foreach (var v in arr3)
        {
            Console.WriteLine("Arr {0}", v);
        }
        string[] strArr =  {"fred","dela","nina"};
        foreach (var v in strArr)
        {
            Console.WriteLine("Arr {0}", v);
        }
        for (int j = 0; j < strArr.Length; j++)
        {
            Console.WriteLine("Arr {0}", strArr[j]);
        }
            string[] ss = new string[4];
        foreach (var item in ss)
        {
            //Console.WriteLine("Arr {0}", item.ToString());
        }
        int?[] intArr = { 1, 2, 3, null, 6 };
        foreach (var item in intArr)
        {
            Console.WriteLine("int? {0}", item);
        }
        string sFred = "Fred";
        if (sFred.ToLower() == "fred")
        {
            Console.WriteLine($"sFred {sFred}");
        }
    }
}
```

Ans:

```
D:\dotnet\HW2>dotnet run Test.cs
Hello c#
Type System.Int32 7
Type System.Char System.UInt32
Type 5 6
numbers 5 4
Arr 1
Arr 2
Arr 3
Arr 4
Arr 5
Arr 6
Arr fred
Arr dela
Arr nina
Arr fred
Arr dela
Arr nina
int? 1
int? 2
int? 3
int?
int? 6
sFred Fred
```

4

Code:

```
abstract class BaseClass
{
    public virtual void vf()
    {
        Console.WriteLine("In BaseClass vf");
    }
    public virtual void vf_new()
    {
        Console.WriteLine("In BaseClass vf_new");
    }
    public abstract void vf_abstract();
    public sealed override string ToString()
    {
        Console.WriteLine("In BaseClass vf_sealed");
        return "BaseClass_ToString";
    }
}

internal class DerivedClass : BaseClass
```

```
{
    public override void vf()
    {
        Console.WriteLine("In DerivedClass vf");
    }
    public new void vf_new()
    {
        Console.WriteLine("In DerviedClass vf_new");
    }
    public override void vf_abstract()
    {
        Console.WriteLine("In DerivedClass vf_abstract");
    }
}

class Test
{
    static void Main()
    {
        System.Console.WriteLine("Hi");
        DerivedClass o2 = new DerivedClass();
        BaseClass o = o2;
        o.vf();
        o.vf_new();
        o.vf_abstract();
        o.ToString();
        o2.vf();
        o2.vf_new();
        o2.vf_abstract();
        o2.ToString();
    }
}
```

Ans:

```
D:\dotnet\HW2>dotnet run BaseClass.cs
Hi
In DerivedClass vf
In BaseClass vf_new
In DerivedClass vf_abstract
In BaseClass vf_sealed
In DerivedClass vf
In DerviedClass vf_new
In DerivedClass vf_abstract
In BaseClass vf_sealed
```