

Name: Upoma khatun

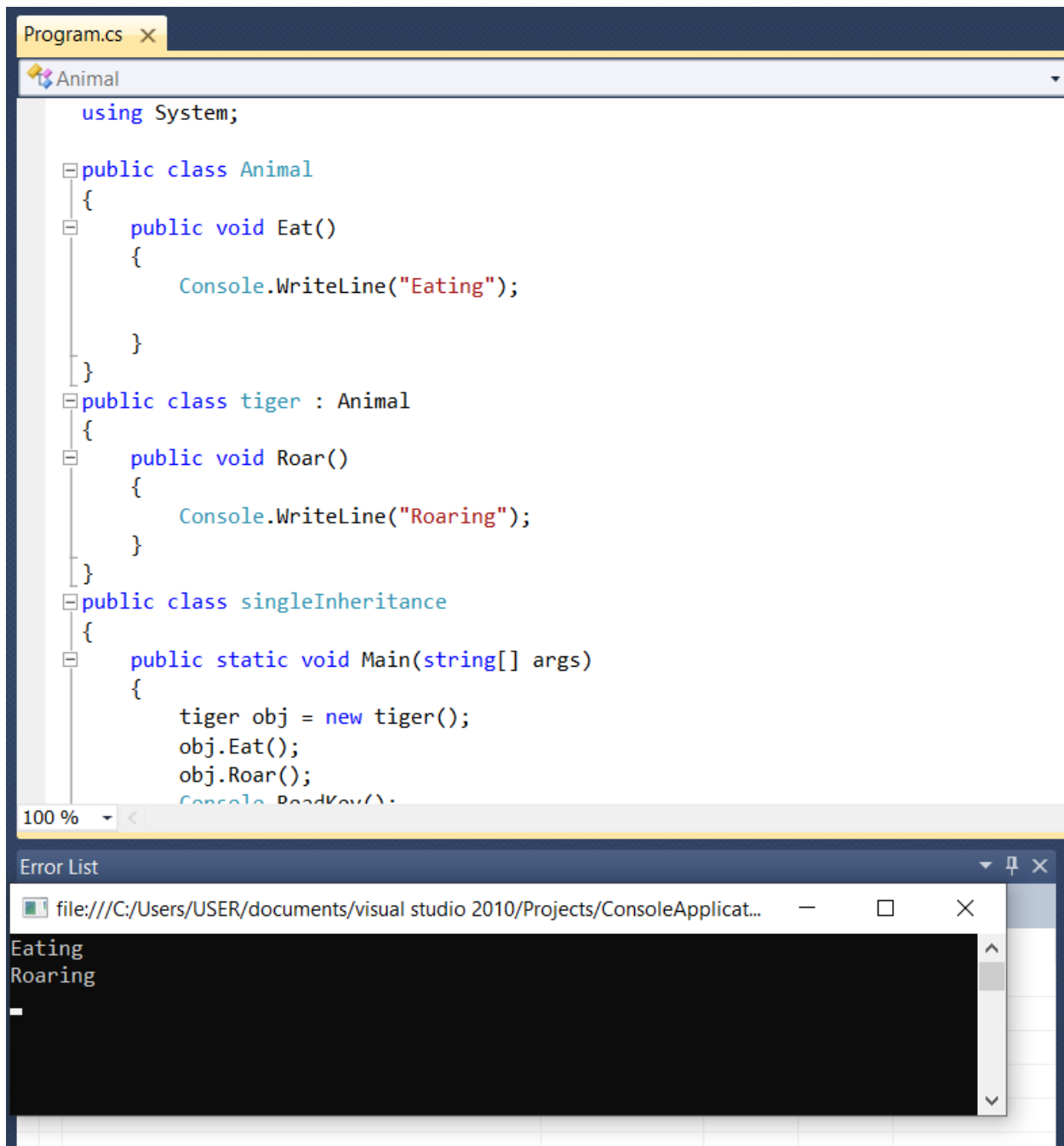
Semester:4th

Technology:Computer

Shift:2nd

Roll: 420650

1. Write a program about single inheritance



```
Program.cs x
Animal
using System;

public class Animal
{
    public void Eat()
    {
        Console.WriteLine("Eating");
    }
}

public class tiger : Animal
{
    public void Roar()
    {
        Console.WriteLine("Roaring");
    }
}

public class singleInheritance
{
    public static void Main(string[] args)
    {
        tiger obj = new tiger();
        obj.Eat();
        obj.Roar();
        Console.ReadKey();
    }
}
```

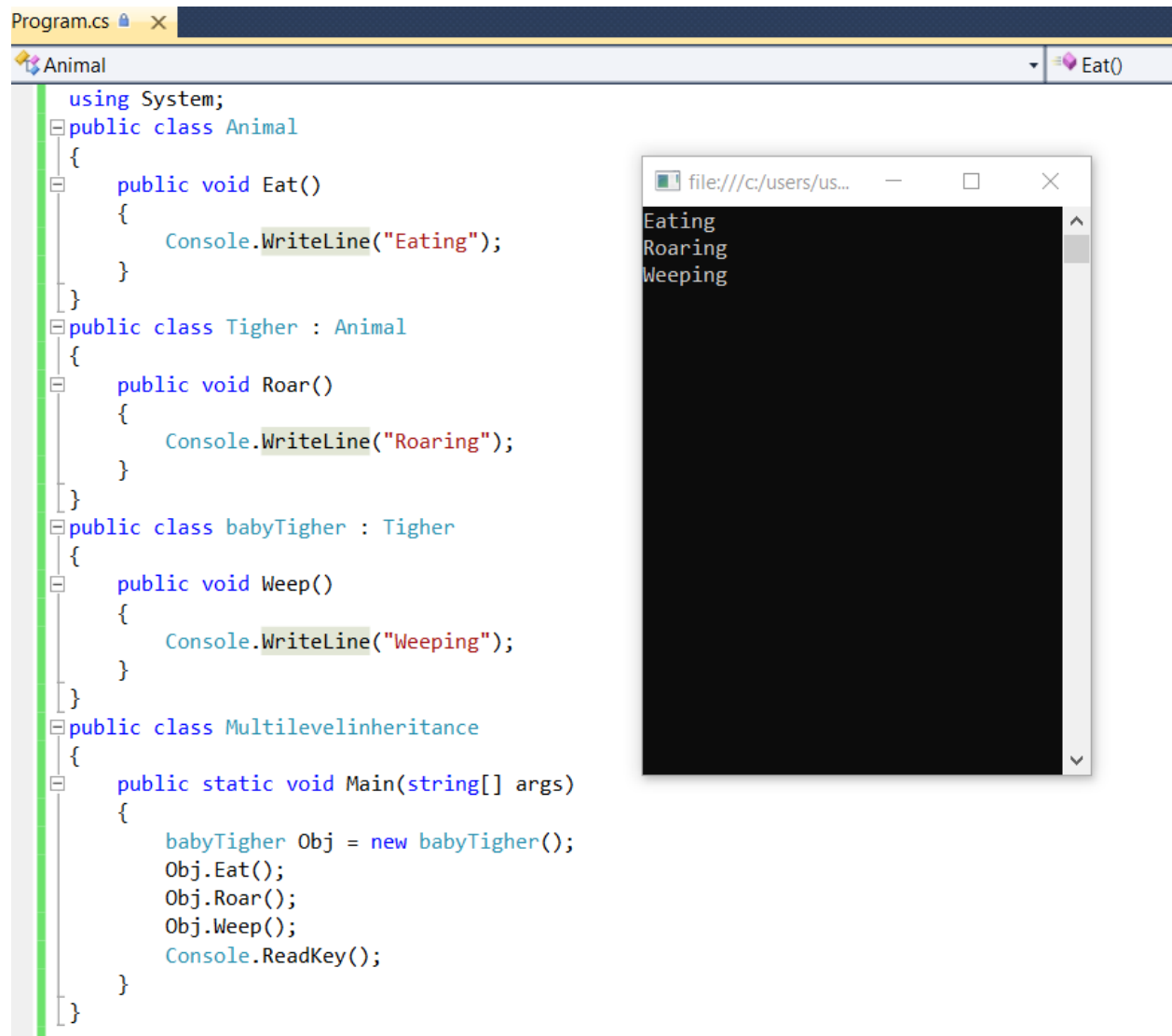
100 %

Error List

file:///C:/Users/USER/documents/visual studio 2010/Projects/ConsoleApplicat...

Eating
Roaring

2. Write a program about multilevel inheritance



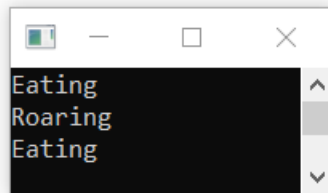
```
Program.cs x
Animal Eat()
using System;
public class Animal
{
    public void Eat()
    {
        Console.WriteLine("Eating");
    }
}
public class Tiger : Animal
{
    public void Roar()
    {
        Console.WriteLine("Roaring");
    }
}
public class babyTiger : Tiger
{
    public void Weep()
    {
        Console.WriteLine("Weeping");
    }
}
public class Multilevelinheritance
{
    public static void Main(string[] args)
    {
        babyTiger Obj = new babyTiger();
        Obj.Eat();
        Obj.Roar();
        Obj.Weep();
        Console.ReadKey();
    }
}
```

file:///c:/users/us... Eating
Roaring
Weeping

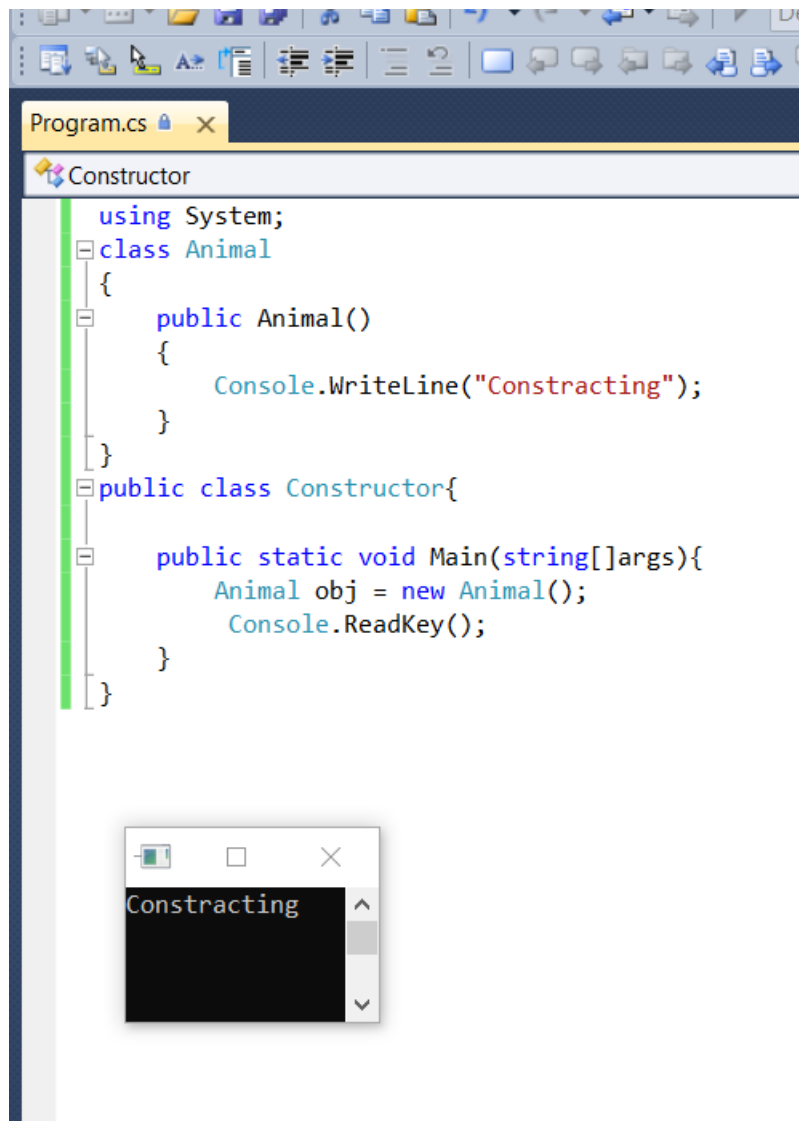
3. Write a program about hierarchical inheritance

```
Program.cs
Lion

using System;
public class Animal
{
    public void Eat()
    {
        Console.WriteLine("Eating");
    }
}
public class Tiger : Animal
{
    public void Roar(){
        Console.WriteLine("Roaring");
    }
}
public class Lion : Animal
{
    public void Run()
    {
        Console.WriteLine("Roaring");
    }
}
public class Hiearinheritance
{
    public static void Main(String[] args){
        Tiger obj = new Tiger();
        Lion obj2 = new Lion();
        obj.Eat();
        obj.Roar();
        obj2.Eat();
        obj2.Run();
        Console.ReadKey();
    }
}
```



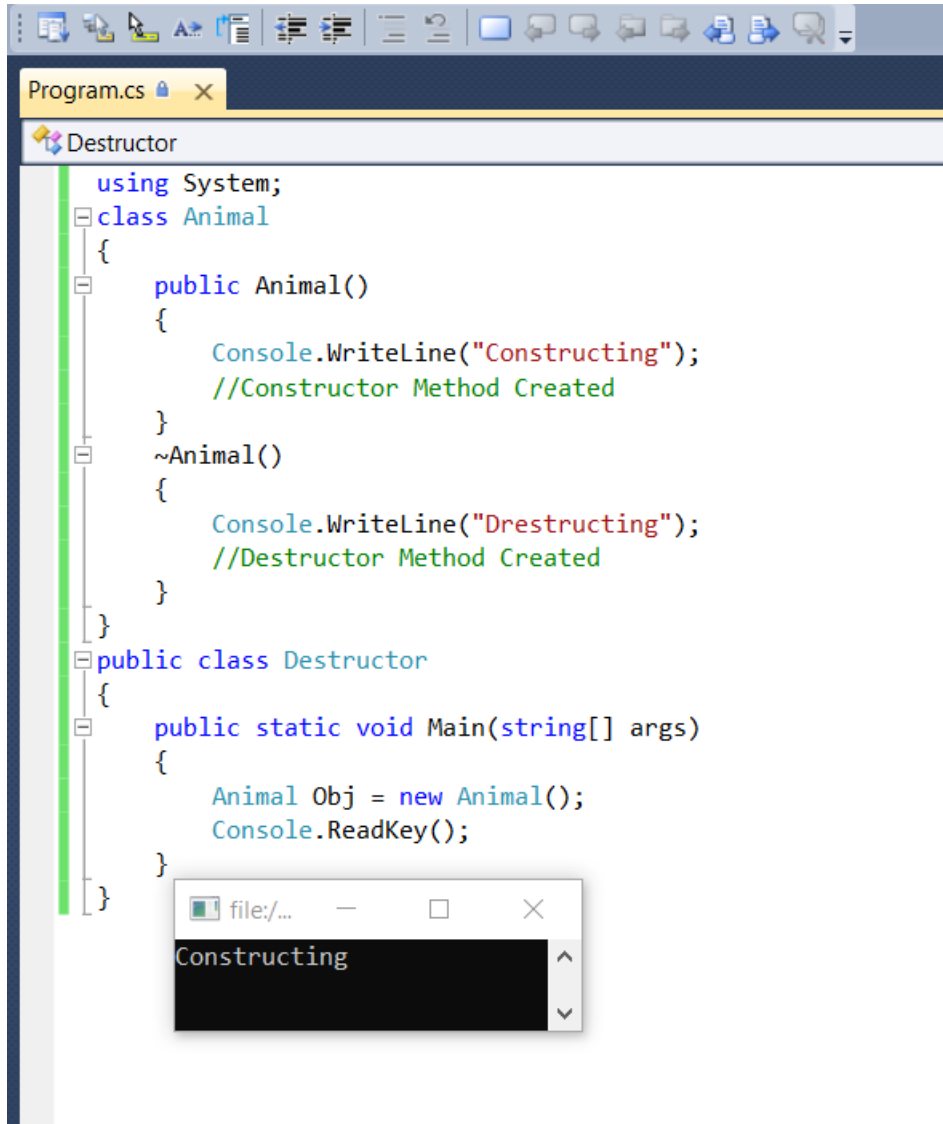
4.write a program about constructor



```
using System;
class Animal
{
    public Animal()
    {
        Console.WriteLine("Constructing");
    }
}
public class Constructor{
    public static void Main(string[]args){
        Animal obj = new Animal();
        Console.ReadKey();
    }
}
```

The screenshot shows a Visual Studio IDE with a file named 'Program.cs'. The code defines a class 'Animal' with a public constructor 'Animal()' that prints 'Constructing' to the console. Below this, a 'public class Constructor' contains a 'Main' method that creates a new 'Animal' object and calls 'Console.ReadKey()' to pause the program. A small console window is open at the bottom, displaying the output 'Constructing'.

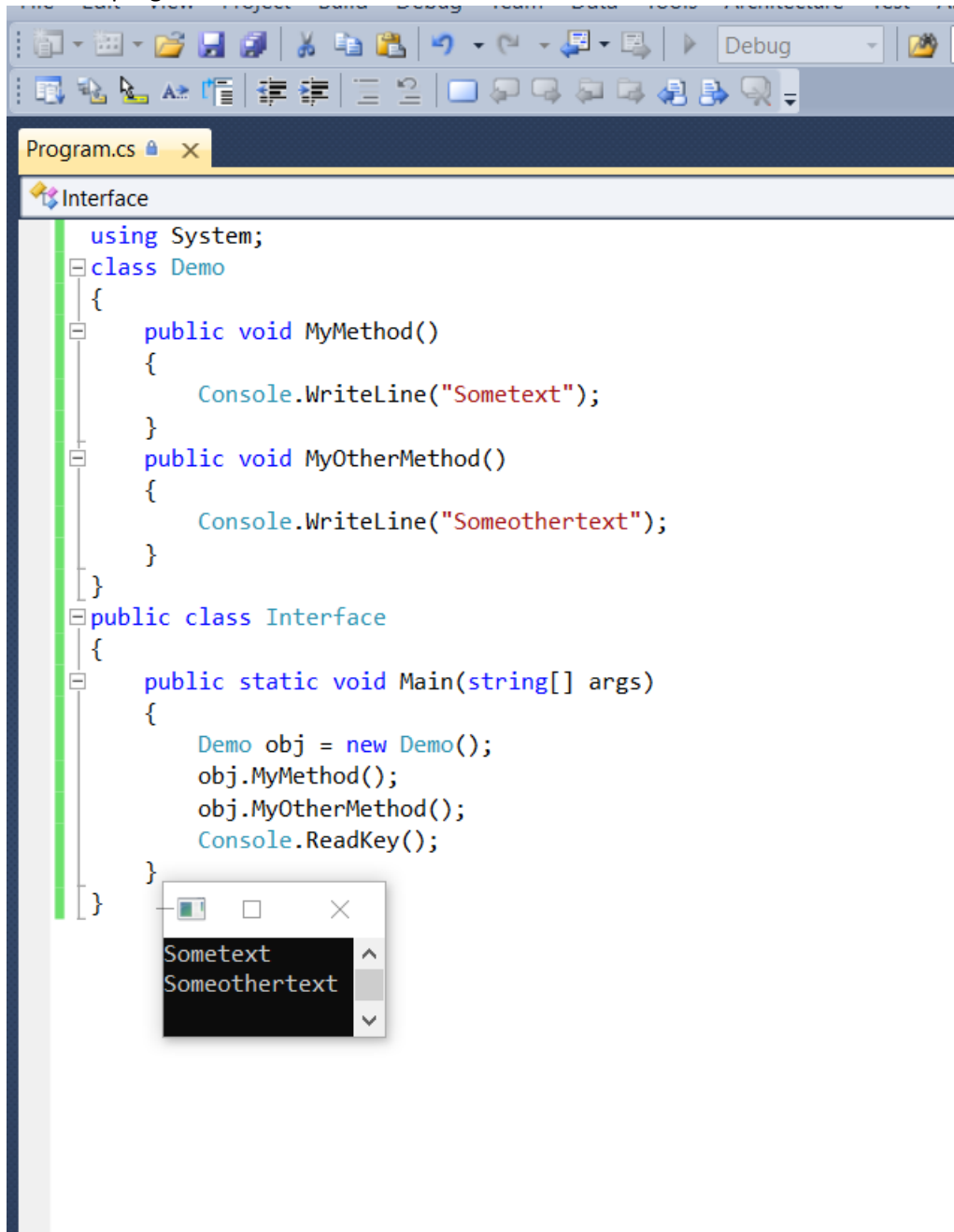
5.write a program about Destructor



```
using System;
class Animal
{
    public Animal()
    {
        Console.WriteLine("Constructing");
        //Constructor Method Created
    }
    ~Animal()
    {
        Console.WriteLine("Drestructuring");
        //Destructor Method Created
    }
}
public class Destructor
{
    public static void Main(string[] args)
    {
        Animal Obj = new Animal();
        Console.ReadKey();
    }
}
```

The screenshot shows a Visual Studio window titled "Program.cs" with a tab labeled "Destructor". The code defines an `Animal` class with a constructor `Animal()` that prints "Constructing" and a destructor `~Animal()` that prints "Drestructuring". A `Destructor` class contains a `Main` method that creates an `Animal` object and calls `Console.ReadKey()`. A console window is open in the foreground, displaying the output "Constructing".

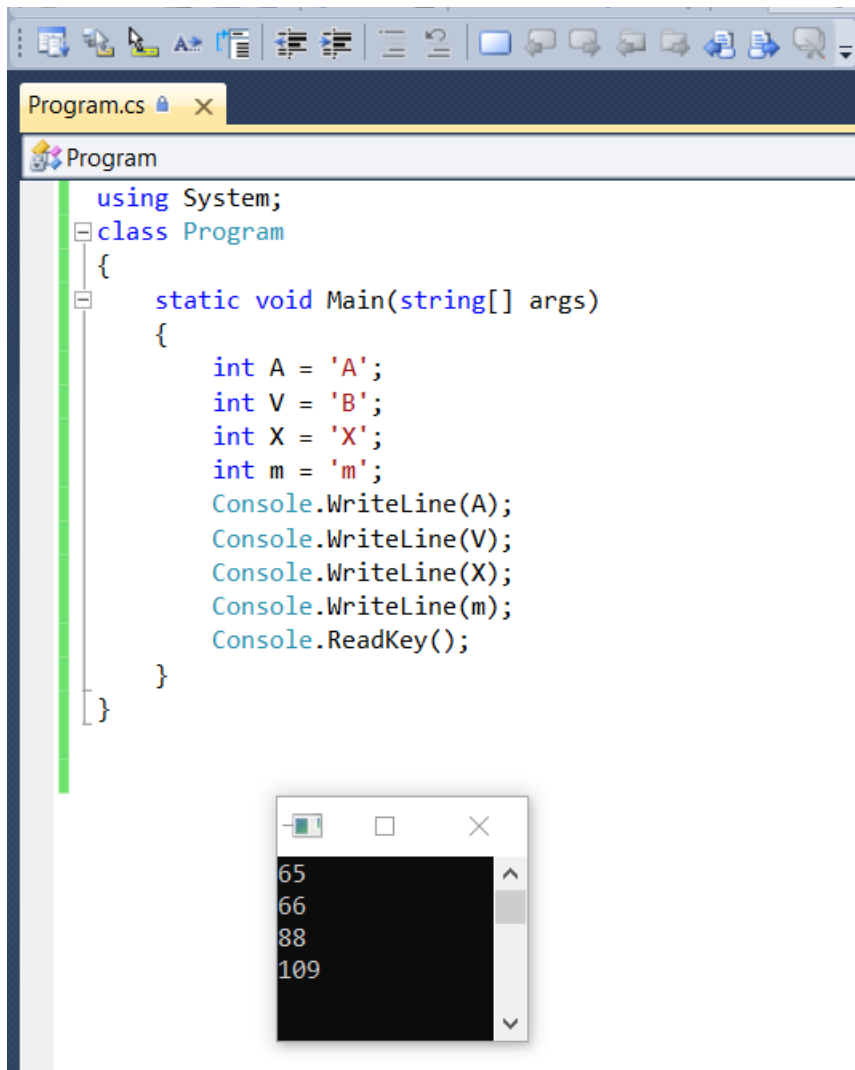
6.write a program about interface



```
using System;
class Demo
{
    public void MyMethod()
    {
        Console.WriteLine("Sometext");
    }
    public void MyOtherMethod()
    {
        Console.WriteLine("Someothertext");
    }
}
public class Interface
{
    public static void Main(string[] args)
    {
        Demo obj = new Demo();
        obj.MyMethod();
        obj.MyOtherMethod();
        Console.ReadKey();
    }
}
```

The screenshot shows a C# program in Visual Studio. The code defines a class `Demo` with two methods: `MyMethod()` which prints "Sometext" and `MyOtherMethod()` which prints "Someothertext". A static class `Interface` contains a `Main` method that creates an instance of `Demo`, calls both methods, and then reads a key from the console. The output window at the bottom shows the two lines of text being printed.

7.write a program about Ascivalue

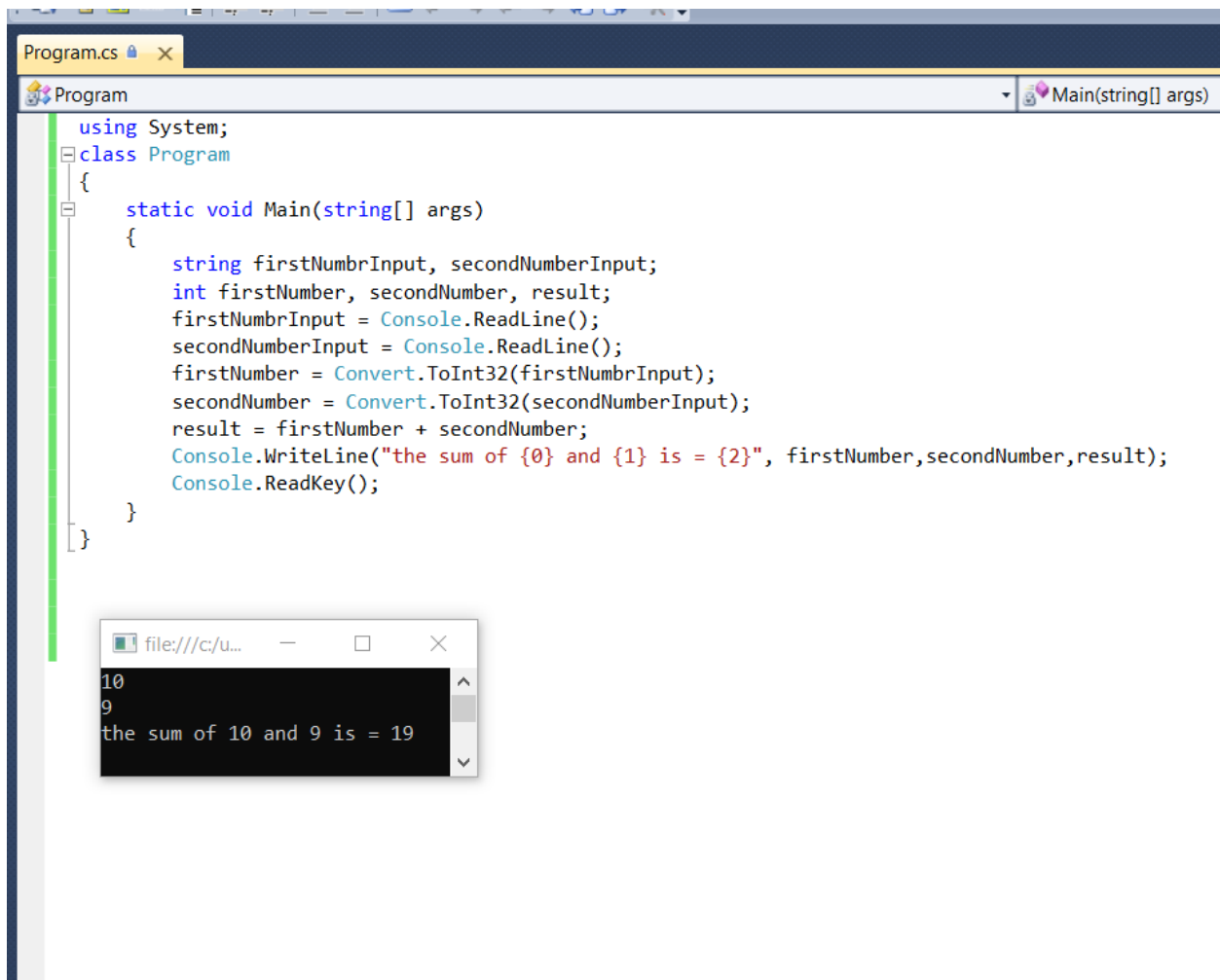


The image shows a screenshot of a C# program in Visual Studio. The code is in a file named `Program.cs` and defines a class `Program` with a `Main` method. The `Main` method declares four integer variables: `A`, `V`, `X`, and `m`, each assigned a character value. It then prints each variable's value to the console using `Console.WriteLine` and waits for a key press with `Console.ReadKey()`. Below the code editor, a console window displays the output of the program, showing the ASCII values of the characters: 65 for 'A', 66 for 'B', 88 for 'X', and 109 for 'm'.

```
using System;
class Program
{
    static void Main(string[] args)
    {
        int A = 'A';
        int V = 'B';
        int X = 'X';
        int m = 'm';
        Console.WriteLine(A);
        Console.WriteLine(V);
        Console.WriteLine(X);
        Console.WriteLine(m);
        Console.ReadKey();
    }
}
```

65
66
88
109

8. 2 ti sonkhar jogfol er program



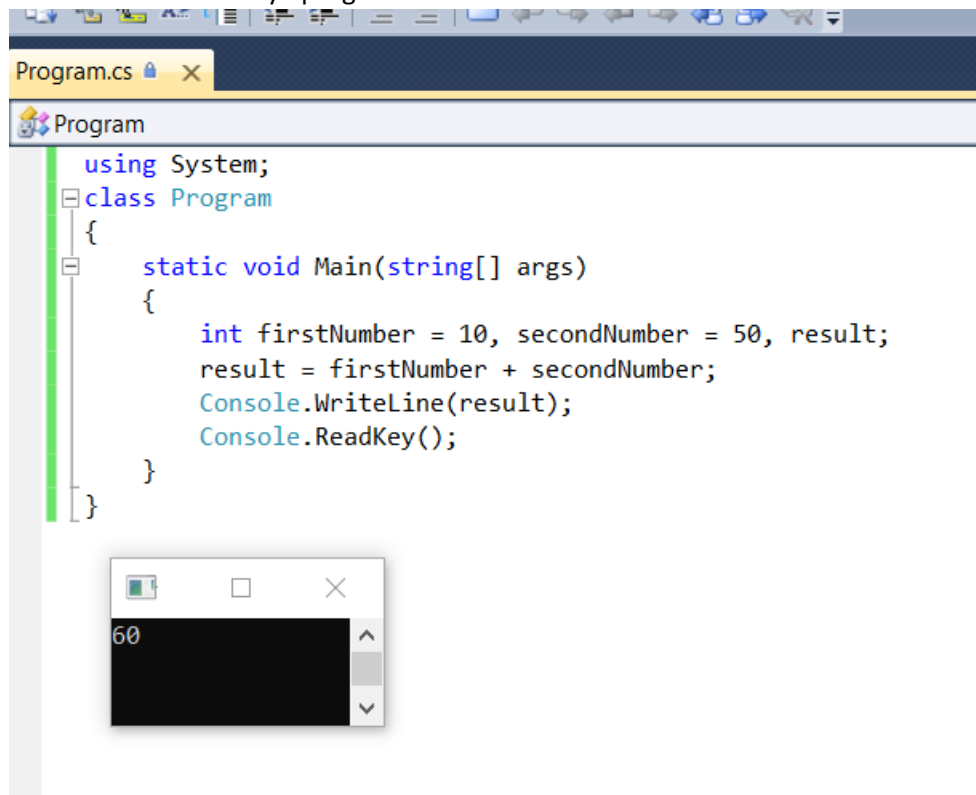
The image shows a C# program in a Visual Studio editor window titled 'Program.cs'. The code defines a class 'Program' with a static 'Main' method. The method prompts the user for two numbers, converts them to integers, calculates their sum, and displays the result. A console window is open below the editor, showing the input '10' and '9', and the output 'the sum of 10 and 9 is = 19'.

```
Program.cs
Program
Main(string[] args)

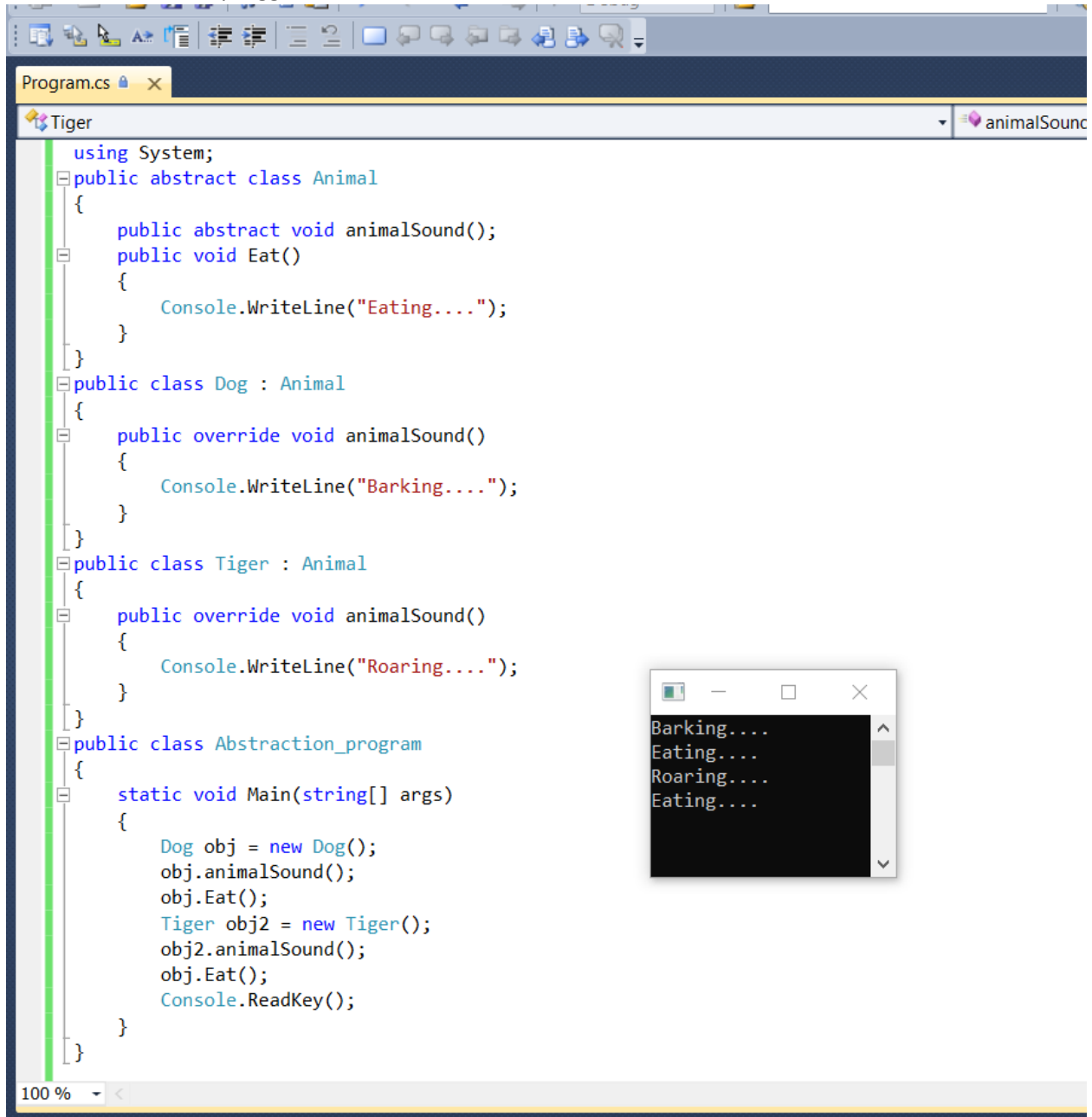
using System;
class Program
{
    static void Main(string[] args)
    {
        string firstNumbrInput, secondNumberInput;
        int firstNumber, secondNumber, result;
        firstNumbrInput = Console.ReadLine();
        secondNumberInput = Console.ReadLine();
        firstNumber = Convert.ToInt32(firstNumbrInput);
        secondNumber = Convert.ToInt32(secondNumberInput);
        result = firstNumber + secondNumber;
        Console.WriteLine("the sum of {0} and {1} is = {2}", firstNumber, secondNumber, result);
        Console.ReadKey();
    }
}
```

file:///c:/u...
10
9
the sum of 10 and 9 is = 19

9. 2 ti sonkhar man niye program



10. Data abstraction program



The screenshot shows a C# IDE with a file named `Program.cs` open. The code defines an abstract class `Animal` with two methods: `animalSound()` and `Eat()`. Two concrete classes, `Dog` and `Tiger`, inherit from `Animal` and override the `animalSound()` method. The `Abstraction_program` class contains a `Main` method that creates instances of `Dog` and `Tiger`, calls their `animalSound()` and `Eat()` methods, and then reads a key from the console.

```
using System;

public abstract class Animal
{
    public abstract void animalSound();
    public void Eat()
    {
        Console.WriteLine("Eating....");
    }
}

public class Dog : Animal
{
    public override void animalSound()
    {
        Console.WriteLine("Barking....");
    }
}

public class Tiger : Animal
{
    public override void animalSound()
    {
        Console.WriteLine("Roaring....");
    }
}

public class Abstraction_program
{
    static void Main(string[] args)
    {
        Dog obj = new Dog();
        obj.animalSound();
        obj.Eat();
        Tiger obj2 = new Tiger();
        obj2.animalSound();
        obj2.Eat();
        Console.ReadKey();
    }
}
```

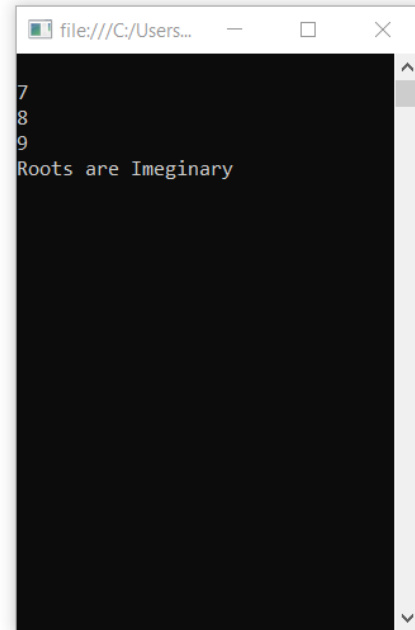
On the right side of the IDE, a small console window is visible, displaying the output of the program:

```
Barking....
Eating....
Roaring....
Eating....
```

11. Dighat somikaron program

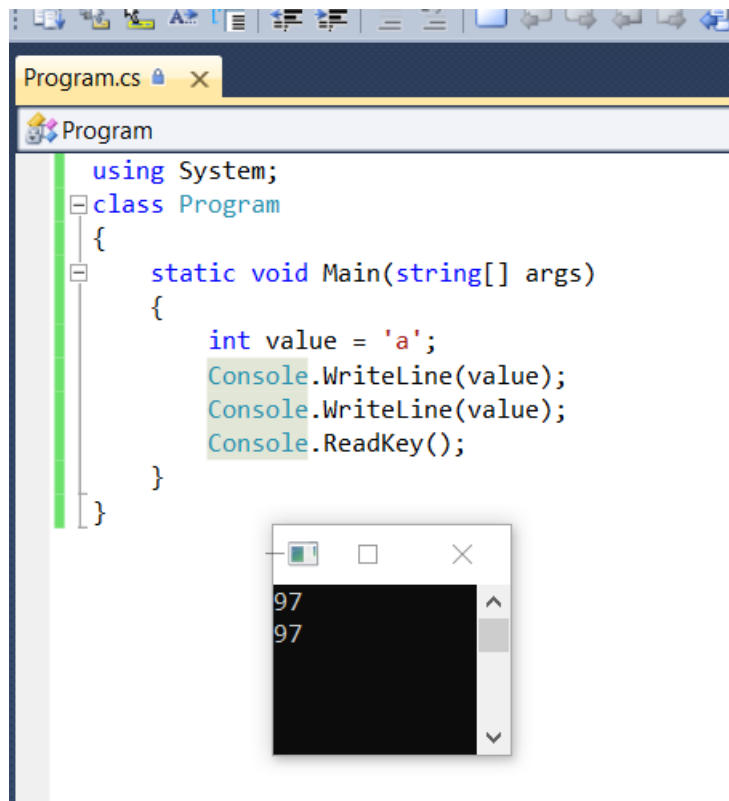
rogram Main(String[] args)

```
using System;
public class Program
{
    public static void Main(String[] args)
    {
        double a = Convert.ToDouble(Console.ReadLine());
        double b = Convert.ToDouble(Console.ReadLine());
        double c = Convert.ToDouble(Console.ReadLine());
        Double D = b * b - 4 * a * c;
        if (D > 0)
        {
            Double r1 = (-b + Math.Sqrt(D)) / 2 * a;
            Double r2 = (-b - Math.Sqrt(D)) / 2 * a;
            Console.WriteLine("Roots are = {0},{1}", r1, r2);
        }
        else if (D == 0)
        {
            double r = -b / 2 * a;
            Console.WriteLine("Roots is = {0}, r");
        }
        else
            Console.WriteLine("Roots are Imeginary");
        Console.ReadKey();
    }
}
```



```
file:///C:/Users...
7
8
9
Roots are Imeginary
```

12. Int type program

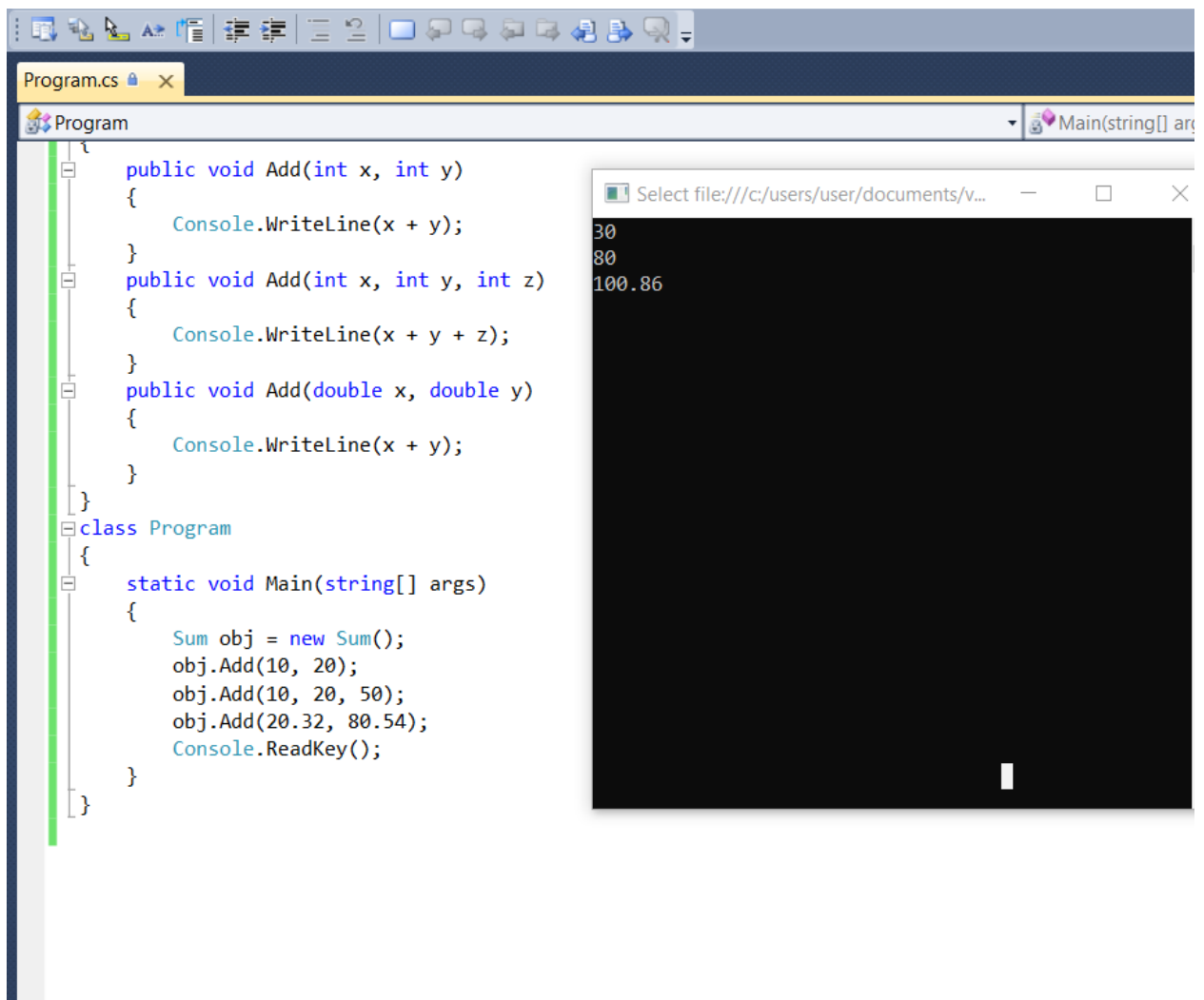


The image shows a Visual Studio window with a C# file named `Program.cs`. The code defines a `Program` class with a `Main` method. Inside `Main`, an `int` variable `value` is assigned the character `'a'`. This character is then printed to the console twice using `Console.WriteLine(value)`, and the program waits for a key press with `Console.ReadKey()`. Below the code editor, a small console window is open, displaying the output of the program: the ASCII value of the character `'a'`, which is `97`, printed on two separate lines.

```
using System;
class Program
{
    static void Main(string[] args)
    {
        int value = 'a';
        Console.WriteLine(value);
        Console.WriteLine(value);
        Console.ReadKey();
    }
}
```

97
97

13. Int sonkha program



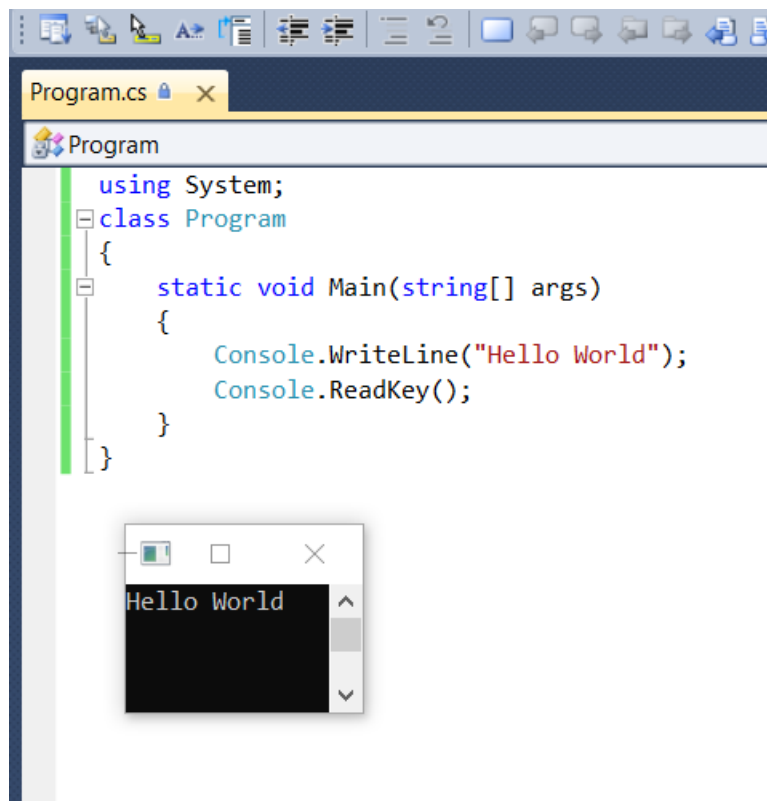
The image shows a C# program in Visual Studio and its console output. The program defines a `Sum` class with three `Add` methods and a `Program` class with a `Main` method. The `Main` method creates a `Sum` object and calls the `Add` methods with various arguments. The console output shows the results of these additions.

```
Program.cs
Program
Main(string[] args)

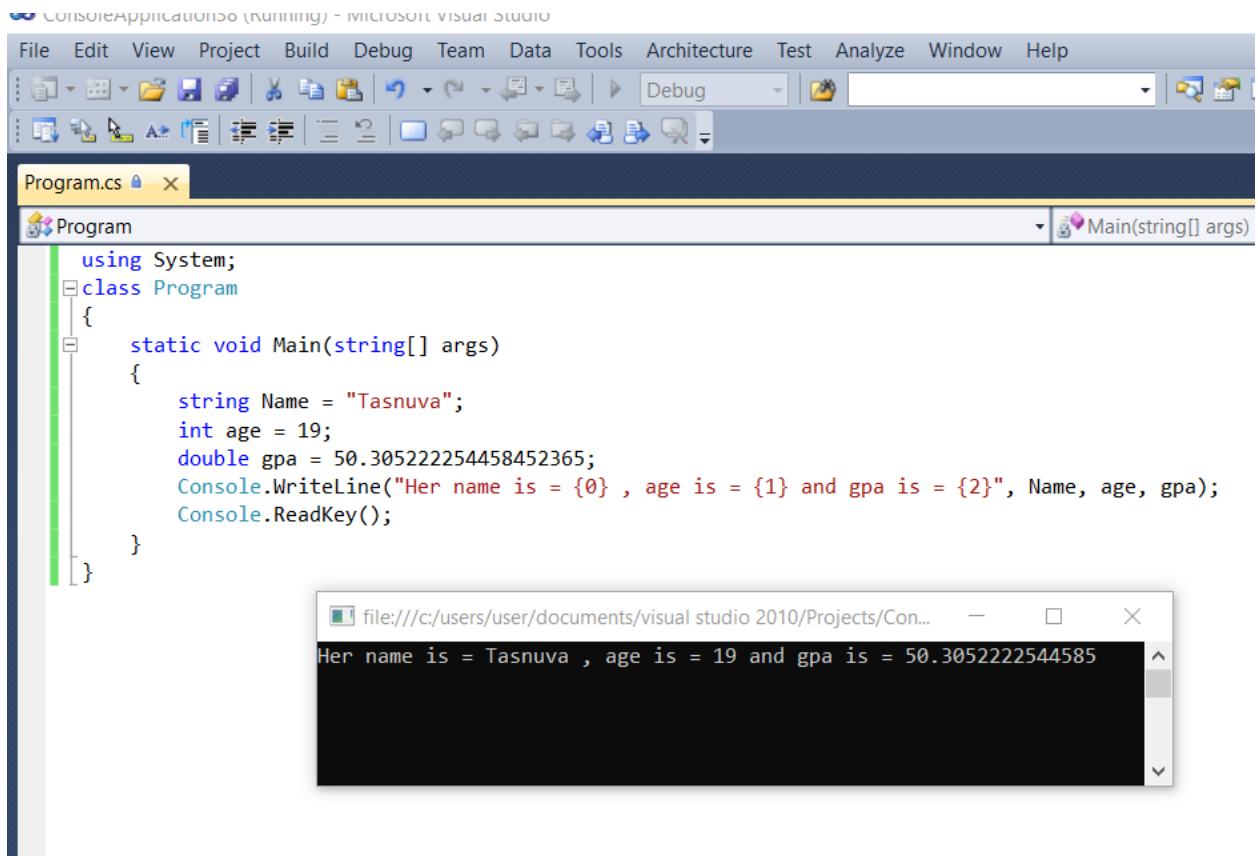
public void Add(int x, int y)
{
    Console.WriteLine(x + y);
}
public void Add(int x, int y, int z)
{
    Console.WriteLine(x + y + z);
}
public void Add(double x, double y)
{
    Console.WriteLine(x + y);
}
}
class Program
{
    static void Main(string[] args)
    {
        Sum obj = new Sum();
        obj.Add(10, 20);
        obj.Add(10, 20, 50);
        obj.Add(20.32, 80.54);
        Console.ReadKey();
    }
}
```

Select file:///c:/users/user/documents/v...
30
80
100.86

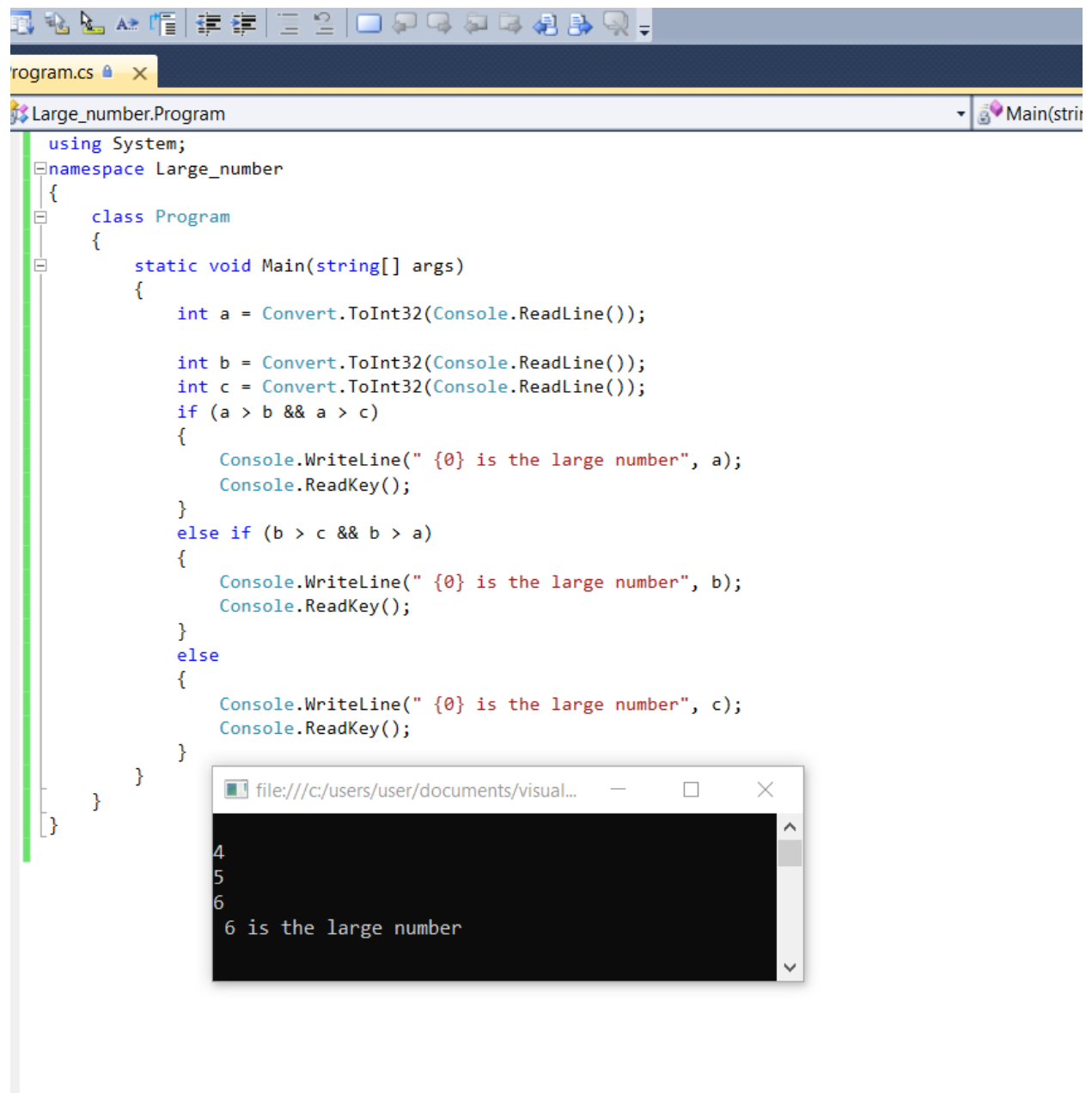
14. Name input niye program



15. Keyword niye program



16. Largest number ber korar program



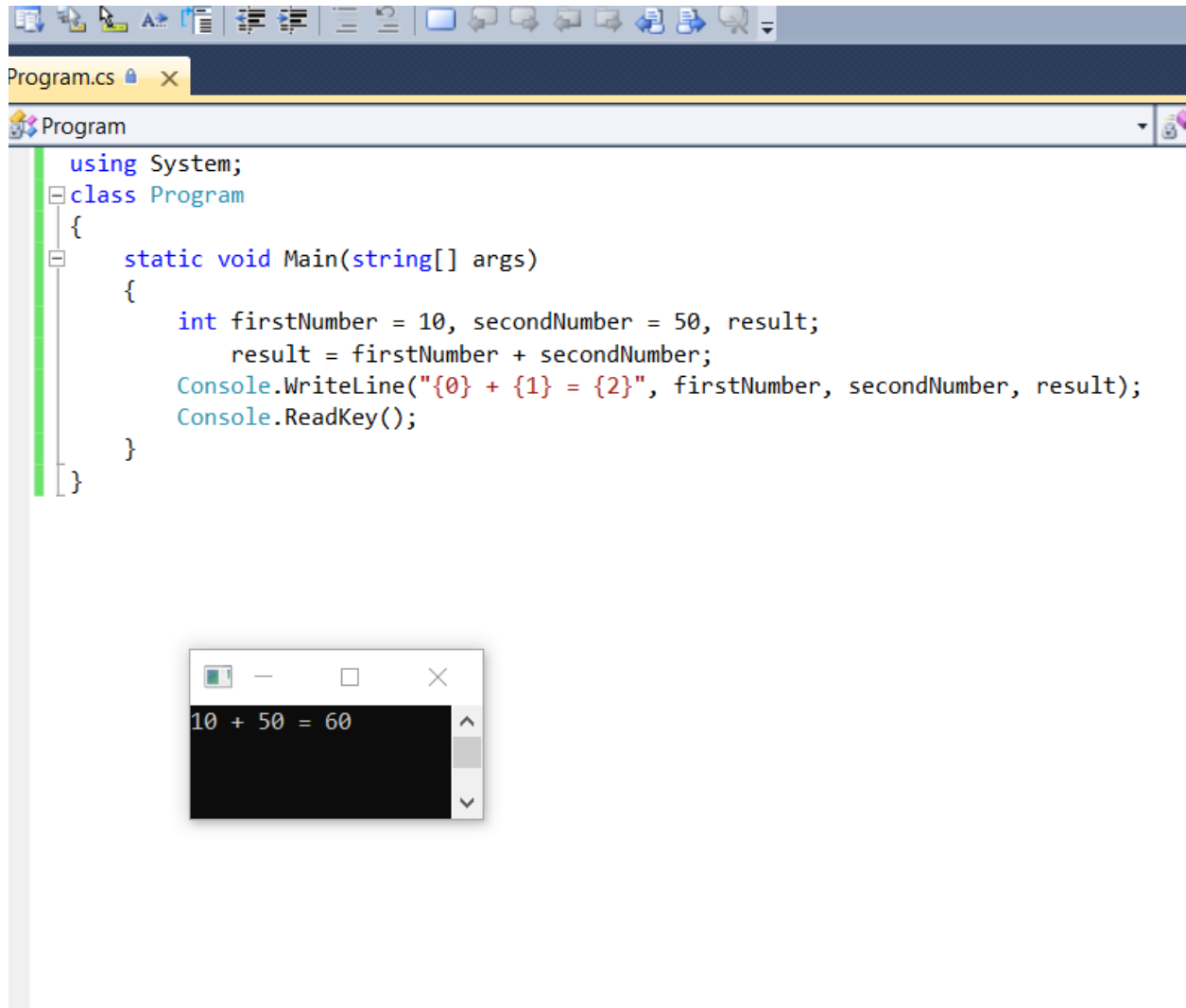
The image shows a screenshot of a C# program in Visual Studio. The program is named 'Large_number.Program' and is located in the file 'program.cs'. The code defines a class 'Program' with a static method 'Main' that takes an array of strings 'args'. The 'Main' method reads three integers 'a', 'b', and 'c' from the console using 'Convert.ToInt32(Console.ReadLine())'. It then uses a series of 'if' and 'else if' statements to determine the largest number. If 'a' is greater than both 'b' and 'c', it prints '{0} is the large number', where '{0}' is replaced by 'a'. If 'b' is greater than both 'a' and 'c', it prints '{0} is the large number', where '{0}' is replaced by 'b'. If 'c' is greater than both 'a' and 'b', it prints '{0} is the large number', where '{0}' is replaced by 'c'. After printing the message, it calls 'Console.ReadKey()' to wait for a key press. The console output shows the numbers 4, 5, and 6 entered on separate lines, followed by the message '6 is the large number'.

```
using System;
namespace Large_number
{
    class Program
    {
        static void Main(string[] args)
        {
            int a = Convert.ToInt32(Console.ReadLine());

            int b = Convert.ToInt32(Console.ReadLine());
            int c = Convert.ToInt32(Console.ReadLine());
            if (a > b && a > c)
            {
                Console.WriteLine(" {0} is the large number", a);
                Console.ReadKey();
            }
            else if (b > c && b > a)
            {
                Console.WriteLine(" {0} is the large number", b);
                Console.ReadKey();
            }
            else
            {
                Console.WriteLine(" {0} is the large number", c);
                Console.ReadKey();
            }
        }
    }
}
```

file:///c:/users/user/documents/visual...
4
5
6
6 is the large number

17. Variable er man ber korar program

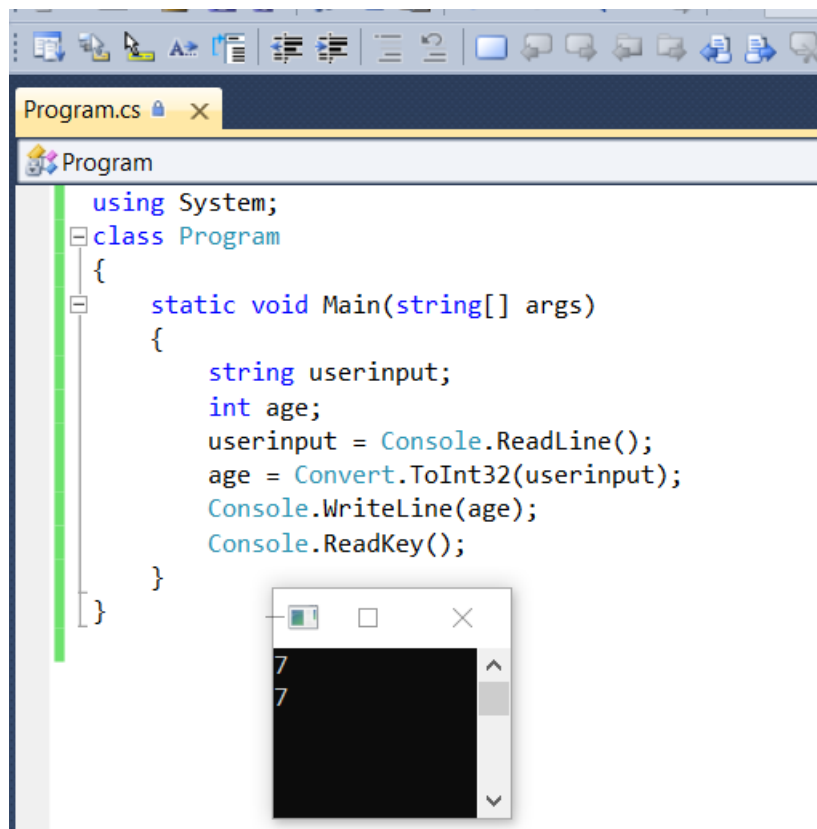


The image shows a screenshot of a C# program in Visual Studio. The code is in a file named `Program.cs` and defines a class `Program` with a `Main` method. The `Main` method initializes two integers, `firstNumber` (10) and `secondNumber` (50), calculates their sum (`result`), and prints it to the console using `Console.WriteLine`. The output in the console window is `10 + 50 = 60`.

```
using System;
class Program
{
    static void Main(string[] args)
    {
        int firstNumber = 10, secondNumber = 50, result;
        result = firstNumber + secondNumber;
        Console.WriteLine("{0} + {1} = {2}", firstNumber, secondNumber, result);
        Console.ReadKey();
    }
}
```

10 + 50 = 60

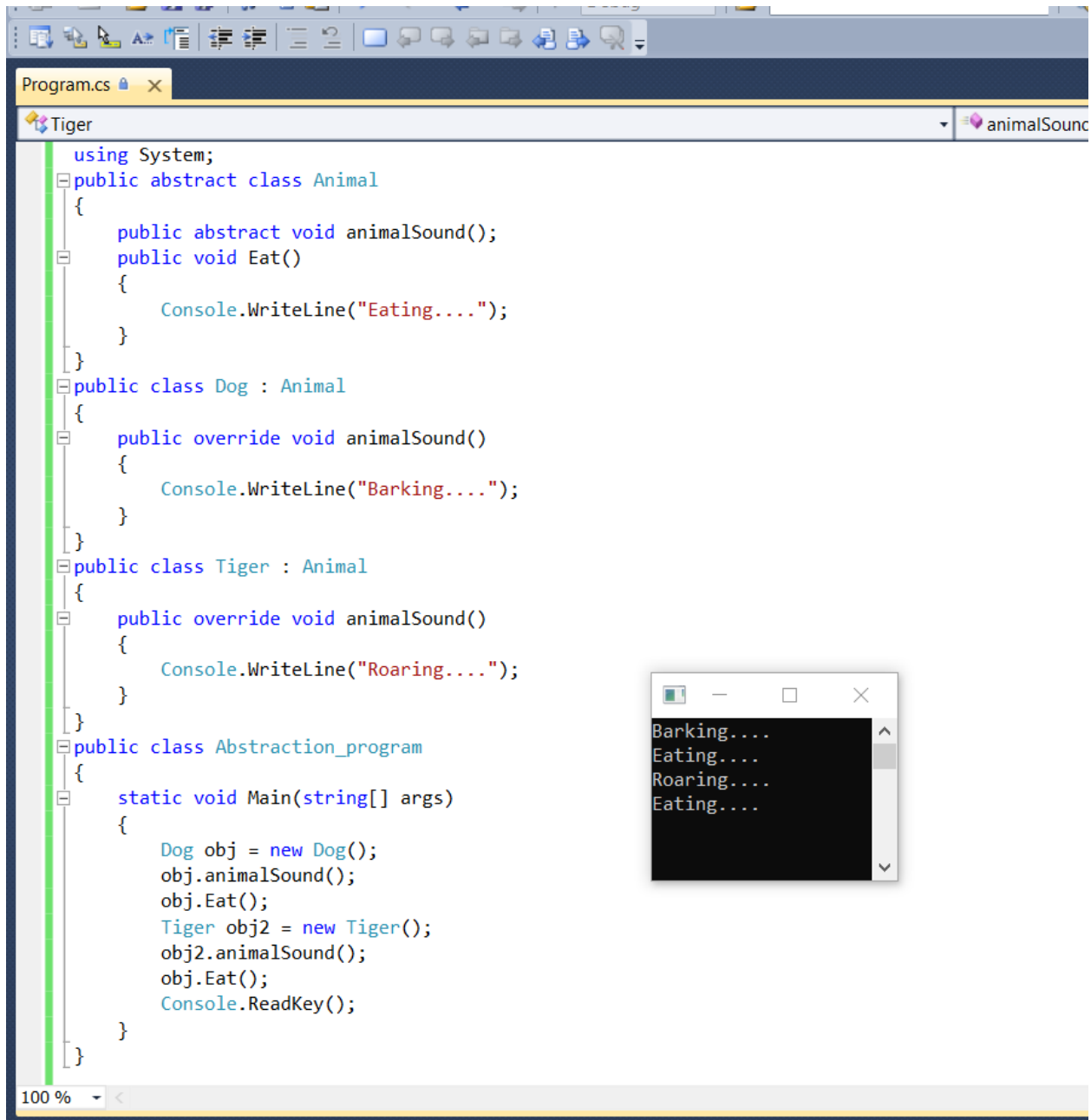
18. Write a program about userinput



The screenshot shows a Visual Studio IDE with a C# program named 'Program.cs'. The code defines a class 'Program' with a static 'Main' method. Inside 'Main', it declares a 'string' variable 'userinput' and an 'int' variable 'age'. It then reads a line of input from the console, converts it to an integer using 'Convert.ToInt32', and writes the integer value back to the console. Finally, it calls 'Console.ReadKey()' to pause the program. A small console window is open in the foreground, displaying the number '7' twice, indicating that the user entered '7' and the program successfully converted and printed it.

```
using System;
class Program
{
    static void Main(string[] args)
    {
        string userinput;
        int age;
        userinput = Console.ReadLine();
        age = Convert.ToInt32(userinput);
        Console.WriteLine(age);
        Console.ReadKey();
    }
}
```

19. Write a program about data abstraction



The screenshot shows a Visual Studio IDE with a C# program named 'Program.cs'. The program defines an abstract class 'Animal' with two methods: 'animalSound()' and 'Eat()'. Two subclasses, 'Dog' and 'Tiger', inherit from 'Animal' and override the 'animalSound()' method. The 'Dog' class overrides it to print 'Barking....' and the 'Tiger' class overrides it to print 'Roaring....'. A 'Main' method in the 'Abstraction_program' class creates instances of 'Dog' and 'Tiger', calls their 'animalSound()' methods, and then calls 'Eat()' on both. A console window is open, showing the output of the program: 'Barking....', 'Eating....', 'Roaring....', and 'Eating....'.

```
using System;

public abstract class Animal
{
    public abstract void animalSound();
    public void Eat()
    {
        Console.WriteLine("Eating....");
    }
}

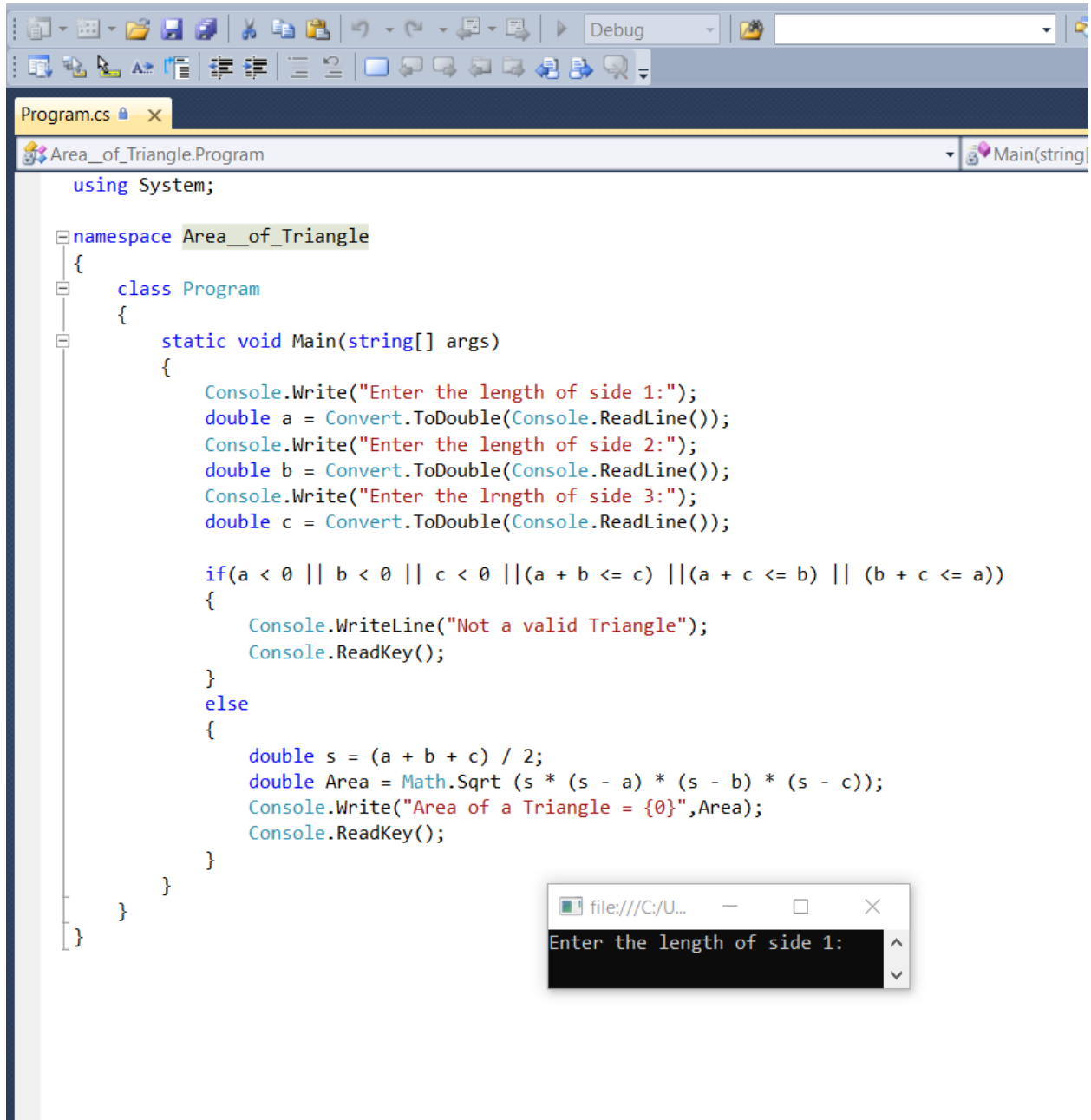
public class Dog : Animal
{
    public override void animalSound()
    {
        Console.WriteLine("Barking....");
    }
}

public class Tiger : Animal
{
    public override void animalSound()
    {
        Console.WriteLine("Roaring....");
    }
}

public class Abstraction_program
{
    static void Main(string[] args)
    {
        Dog obj = new Dog();
        obj.animalSound();
        obj.Eat();
        Tiger obj2 = new Tiger();
        obj2.animalSound();
        obj2.Eat();
        Console.ReadKey();
    }
}
```

100 %

20. Write a program about Tringle program



```
using System;

namespace Area_of_Triangle
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter the length of side 1:");
            double a = Convert.ToDouble(Console.ReadLine());
            Console.WriteLine("Enter the length of side 2:");
            double b = Convert.ToDouble(Console.ReadLine());
            Console.WriteLine("Enter the lnrngth of side 3:");
            double c = Convert.ToDouble(Console.ReadLine());

            if(a < 0 || b < 0 || c < 0 || (a + b <= c) || (a + c <= b) || (b + c <= a))
            {
                Console.WriteLine("Not a valid Triangle");
                Console.ReadKey();
            }
            else
            {
                double s = (a + b + c) / 2;
                double Area = Math.Sqrt(s * (s - a) * (s - b) * (s - c));
                Console.WriteLine("Area of a Triangle = {0}",Area);
                Console.ReadKey();
            }
        }
    }
}
```

file:///C:/U... Enter the length of side 1: