Introduction

In this course, we will learn a new programming language that is known as C# (C Sharp). Now that you have successfully installed and set up the Visual Studio on your computer, let's learn about coding some basic operations in C#.

Let's do some coding.

Output Operation in C#

We have learned in the previous manual about printing output on the console screen. For revision, let's use the "Hello World" example.

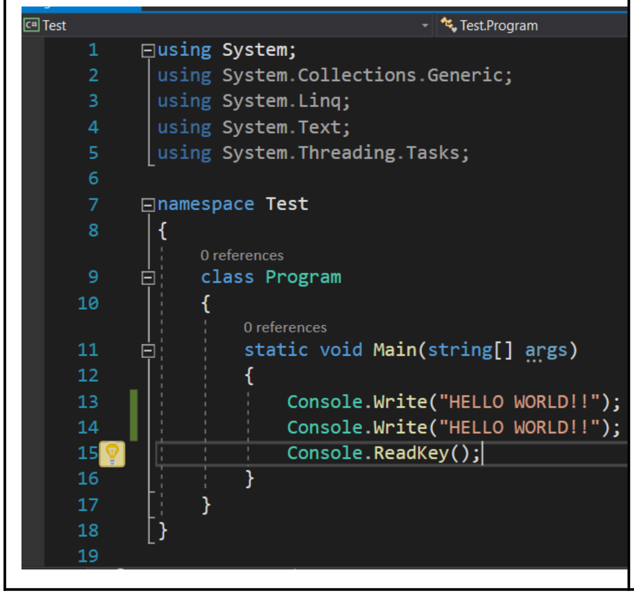
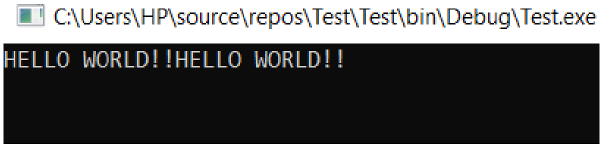
"Console.Write("Hello World")"

This line will print the message inside the quotation mark on the existing line where the cursor is in the console.

Task: To understand this concept, try writing a program that prints "Hello World" twice on the screen.

Solution:

Write the following code into the main function of the code and execute the program by clicking on the start button.

Code:	Output:
	

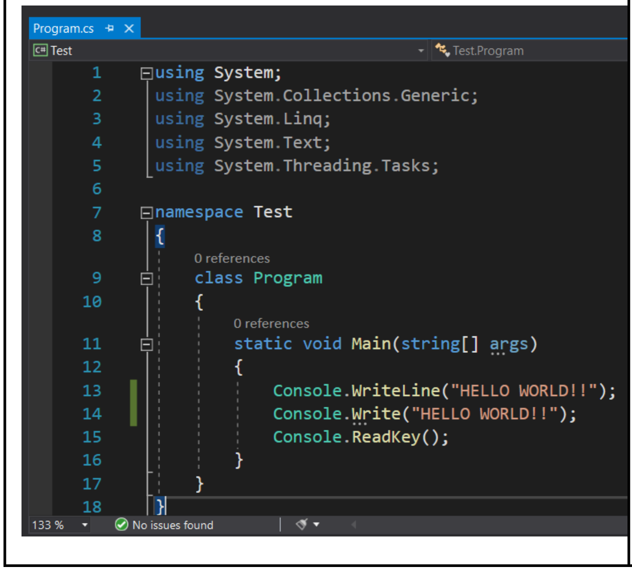
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.Write("HELLO WORLD!!");
            Console.Write("HELLO WORLD!!");
            Console.ReadKey();
        }
    }
}
```

```
HELLO WORLD!!HELLO WORLD!!
```

“Console.WriteLine(“Hello World”)”
This line will print the message on the next line.

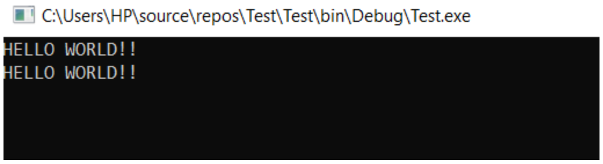
Task: To understand this concept, try writing a program that prints “Hello World” on two separate lines on the screen.
Solution:
Write the following code into the main function of the code and execute the program by clicking on the start button.

Code:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("HELLO WORLD!!");
            Console.Write("HELLO WORLD!!");
            Console.ReadKey();
        }
    }
}
```

Output:



```
HELLO WORLD!!
HELLO WORLD!!
```

Variables in C#

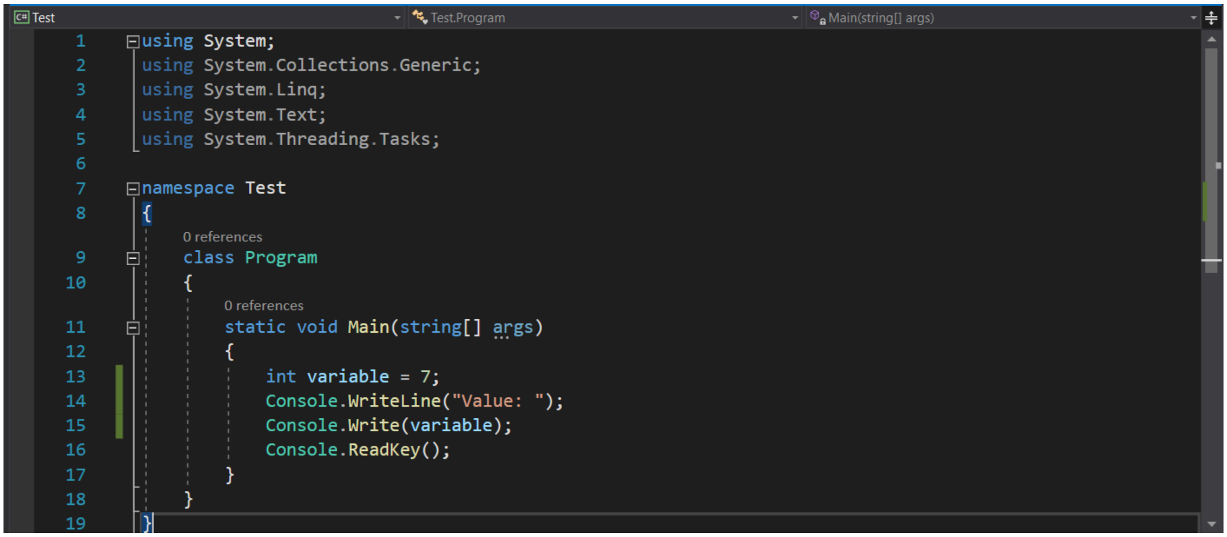
Just like other programming languages, we have variables of various data types that are used in the programs according to user requirements.

Integer Datatype

The declaration and use of integer datatype are very simple and straightforward.

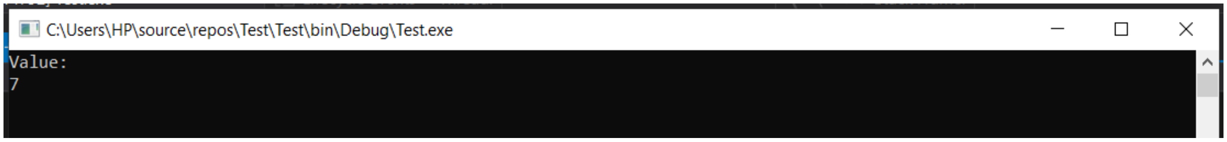
Look at the following code snippet to understand the use of the “**int**” datatype in C#.

Code:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            int variable = 7;
            Console.WriteLine("Value: ");
            Console.Write(variable);
            Console.ReadKey();
        }
    }
}
```

Output:



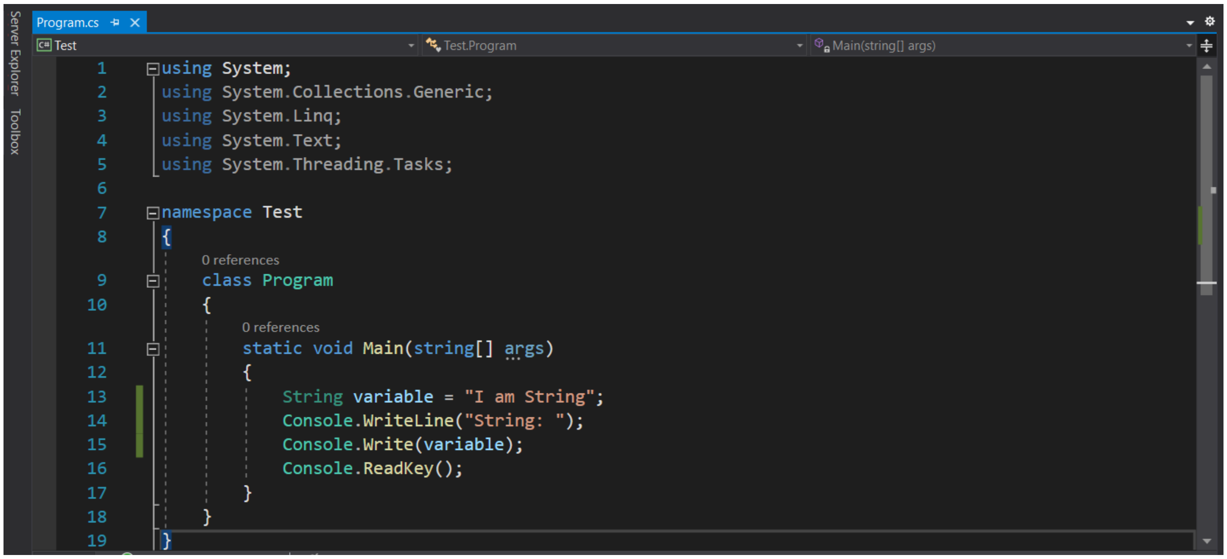
```
Value:
7
```

String Datatype

Similarly, the string datatype is very simple to use. The “**String**” keyword is used for declaring a string-type keyword.

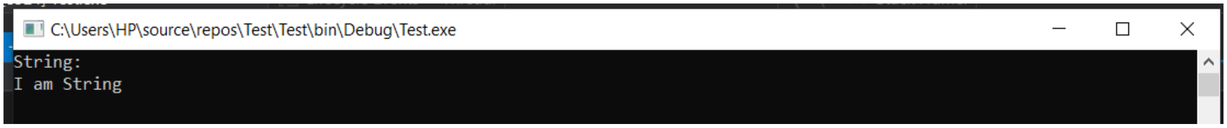
Look at the following code snippet to understand the use of the “**String**” datatype in C#.

Code:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            String variable = "I am String";
            Console.WriteLine("String: ");
            Console.Write(variable);
            Console.ReadKey();
        }
    }
}
```

Output:



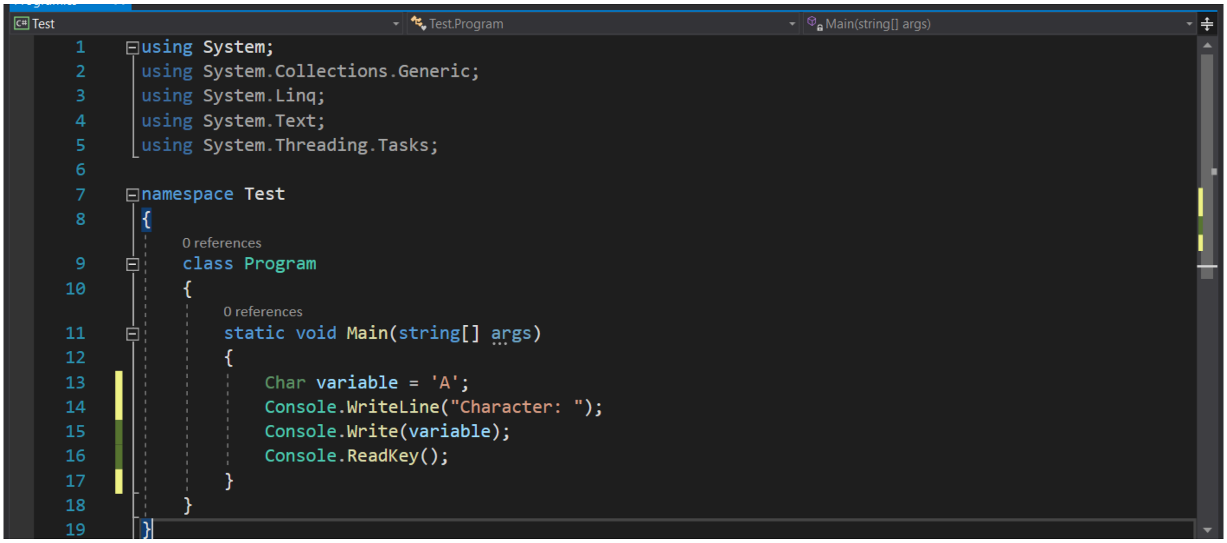
```
String:
I am String
```

Character Datatype

Variable of character datatype are declared using the “**Char**” keyword.

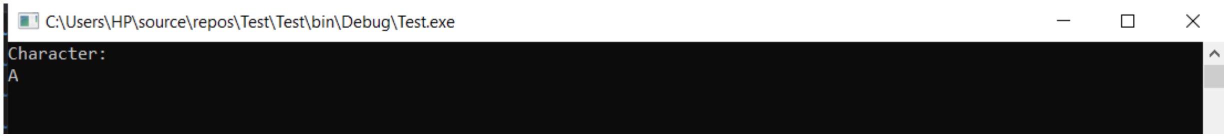
Look at the following code snippet to understand the use of the “**Char**” datatype in C#.

Code:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Char variable = 'A';
            Console.WriteLine("Character: ");
            Console.Write(variable);
            Console.ReadKey();
        }
    }
}
```

Output:



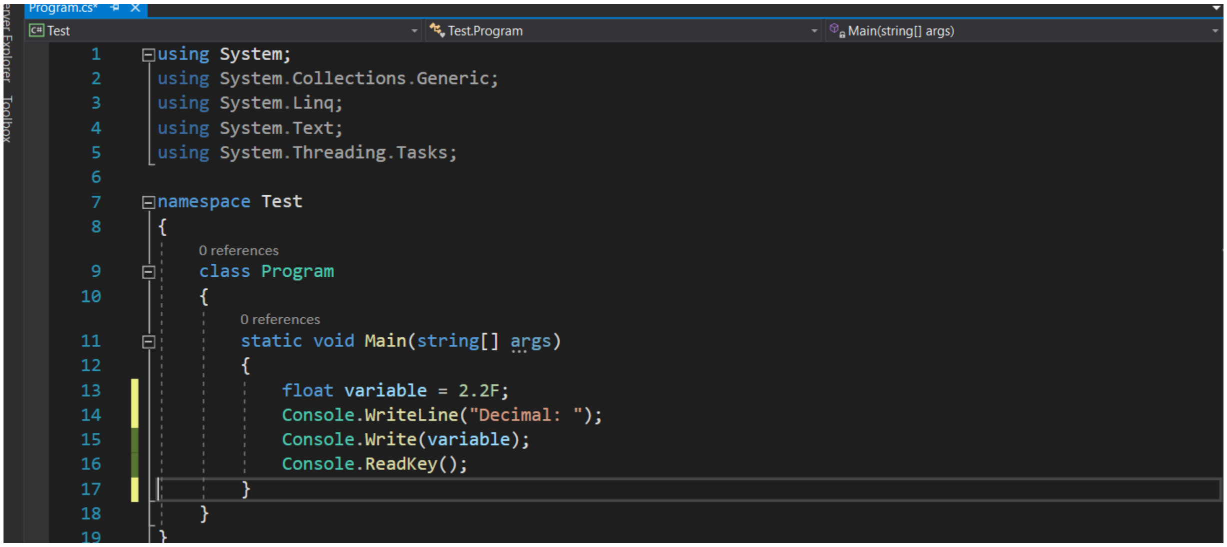
```
Character:
A
```

Float Datatype

The Float datatype is slightly different in syntax as we need to define the variable explicitly by adding an “F” at the end of the given value.

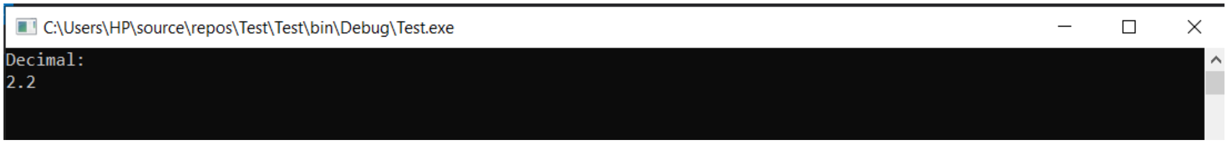
Look at the following code snippet to understand the use of the “**Float**” datatype in C#.

Code:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            float variable = 2.2F;
            Console.WriteLine("Decimal: ");
            Console.Write(variable);
            Console.ReadKey();
        }
    }
}
```

Output:



```
Decimal:
2.2
```

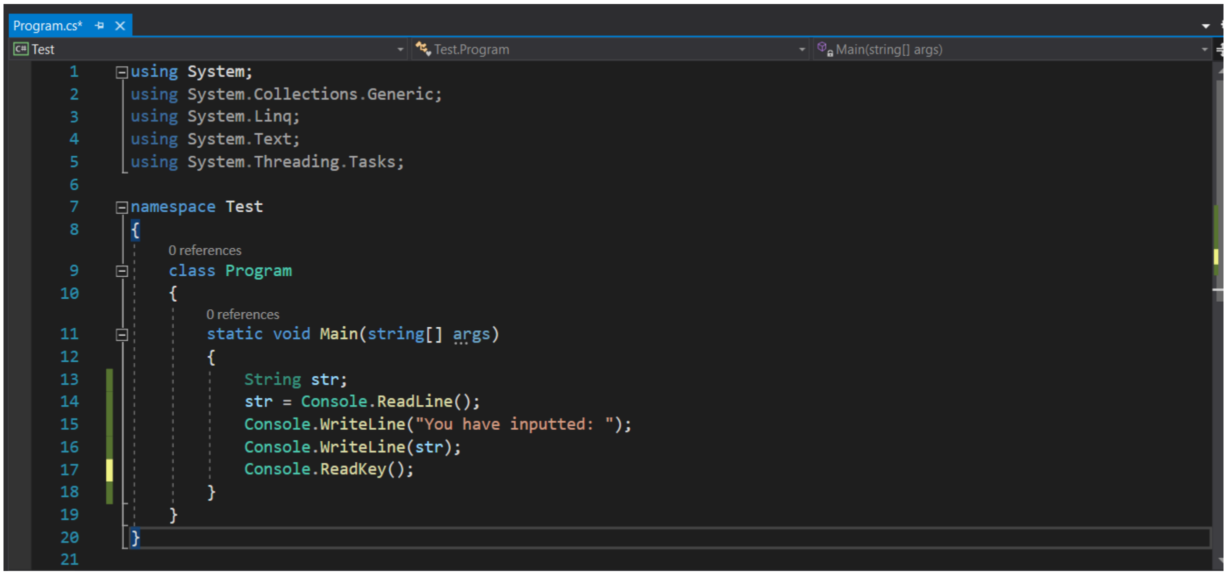
Getting Input from User

The most important thing to remember is that every time you will take input from the user, it will be considered as String type input by the C#. This is the by-default behavior of the language.

“**Console.Readline()**” is used to take input from the user.

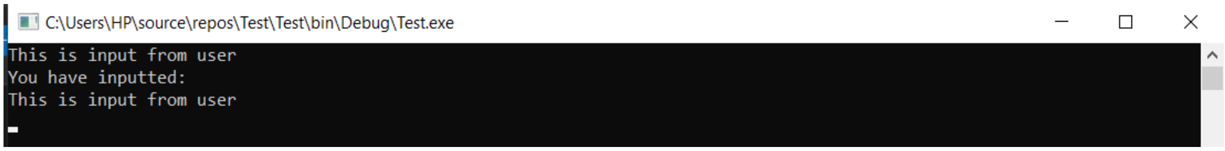
Look at the following code snippet to understand this functionality.

Code:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            String str;
            str = Console.ReadLine();
            Console.WriteLine("You have inputted: ");
            Console.WriteLine(str);
            Console.ReadKey();
        }
    }
}
```

Output:



```
This is input from user
You have inputted:
This is input from user
```

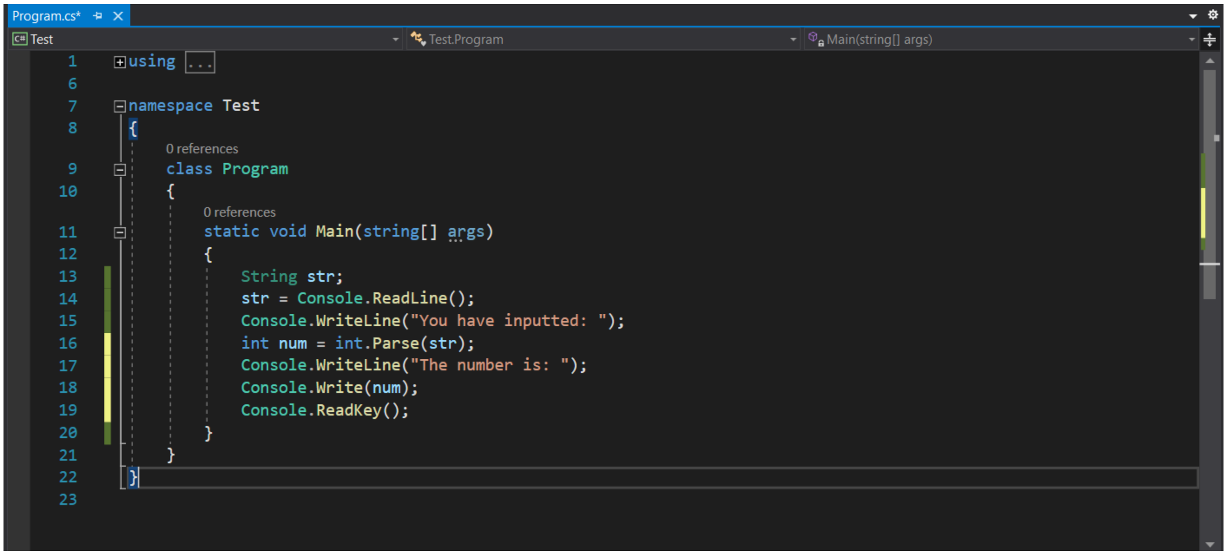
Taking input as Integer from User

Now, we know that every time we will take input from the user, it will be taken as "**string**" datatype. However, what if we need to take the input as "**int**" or "**float**".

To solve this problem, we will use the built-in functionality of typecasting in C#. It will convert the given string type input into the required format.

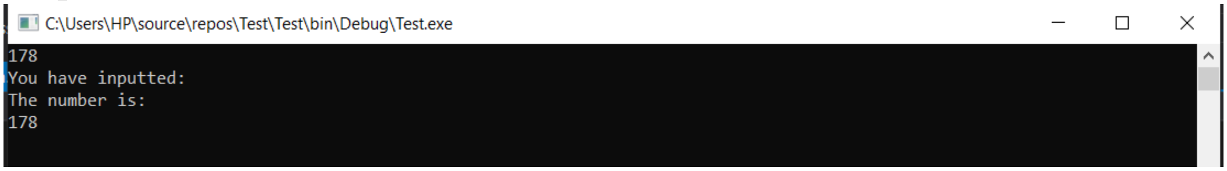
Look at the following code snippet to grasp the understanding of this concept.

Code:



```
using ...

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            String str;
            str = Console.ReadLine();
            Console.WriteLine("You have inputted: ");
            int num = int.Parse(str);
            Console.WriteLine("The number is: ");
            Console.Write(num);
            Console.ReadKey();
        }
    }
}
```

Output:



```
178
You have inputted:
The number is:
178
```

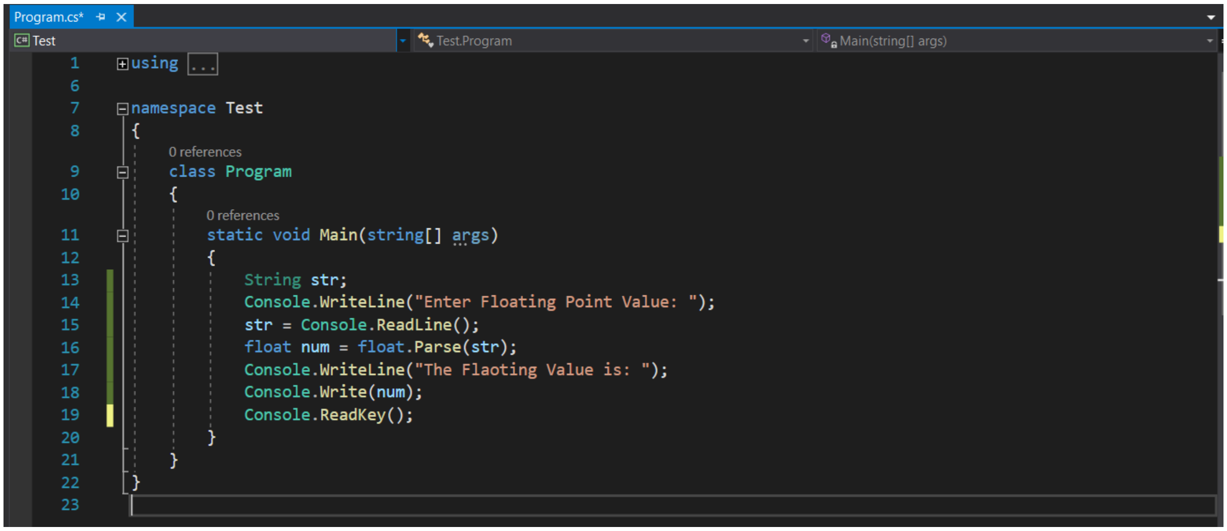
Taking input as a float from User

Just like the previous phenomenon, we need to convert the received input from the user into Float as well if it is the kind of data that we want to use in our program.

We would use the “**float.Parse()**” function to convert the received input through the “**Console.ReadLine()**” function.

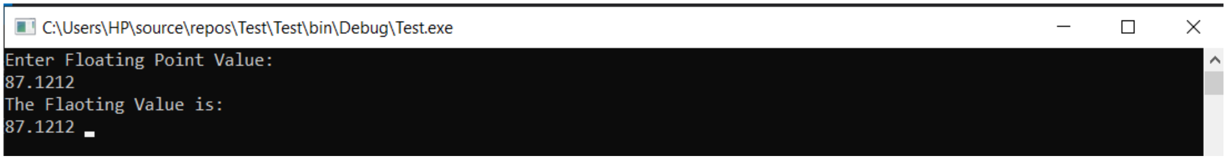
Look at the following code snippet to grasp the understanding of this concept.

Code:



```
using ...

namespace Test
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            String str;
            Console.WriteLine("Enter Floating Point Value: ");
            str = Console.ReadLine();
            float num = float.Parse(str);
            Console.WriteLine("The Flaoting Value is: ");
            Console.Write(num);
            Console.ReadKey();
        }
    }
}
```

Output:



```
Enter Floating Point Value:
87.1212
The Flaoting Value is:
87.1212
```

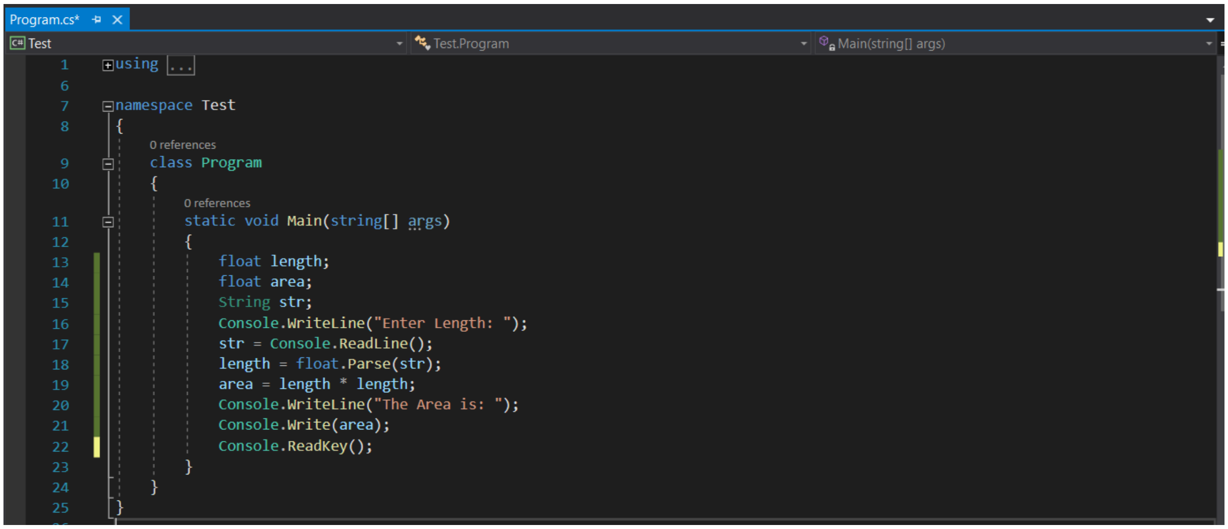
Calculating the Area of Square

Task: Write a program that takes the length of one side of a square and prints the total area of the square on the screen.

Solution:

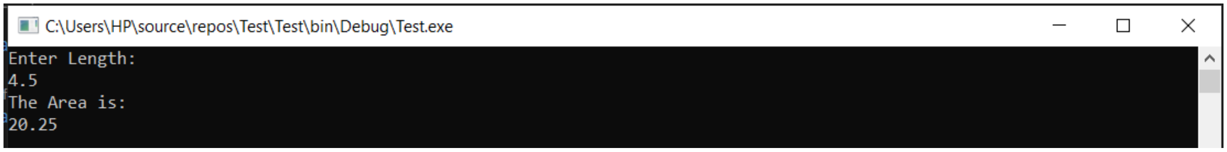
You can look below for a working example of this code.

Code:



```
using ...

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            float length;
            float area;
            String str;
            Console.WriteLine("Enter Length: ");
            str = Console.ReadLine();
            length = float.Parse(str);
            area = length * length;
            Console.WriteLine("The Area is: ");
            Console.Write(area);
            Console.ReadKey();
        }
    }
}
```

Output:



```
Enter Length:
4.5
The Area is:
20.25
```

Working Details for the above-mentioned program.

Actions		Description
	1	Creating a new project and putting the cursor inside the main function.

float area;
float length;
String str;
	2	Declaring two variables for
		– **Length** and **Area**
		Declaring a String type variable for taking input.

Console.WriteLine(“Enter Length: ”);	3	Display on Screen to enter Length

str = Console.ReadLine();	**4**	Take input from the user and storing into a variable
length = float.Parse(str);	**5**	Convert the input into Float type
area = length * length;	**6**	Calculation of Area

Console.WriteLine(“The Area is: ”);
Console.Write(area);
	7	Print the results on screen

Console.ReadKey();	**8**	Write the code for holding the output on screen until the user enters a key

Good Luck and Best Wishes !!

Happy Coding ahead :)