1.1P: Preparing for OOP – Answer Sheet

1. Explain the following terminal instructions:
   a. cd: *to change the directory that is accessed*
   b. ls: *to list any contents of the current directory*
   c. pwd: *to show the full path name from the root directory to the current directory*

2. Consider the following kinds of information, and suggest the most appropriate data type to store or represent each:

Information	Suggested Data Type
A person's name	*String*
A person's age in years	*Int*
A phone number	*String*
A temperature in Celsius	*String*
The average age of a group of people	*Float, Double*
Whether a person has eaten lunch	*Boolean*

3. Aside from the examples already provided in question 2, come up with an example of information that could be stored as:

Data type	Suggested Information
String	*My favourites film's name*
Integer	*Number of units I have enrolled in this semester*
Float	*My height*
Boolean	*Whether I have done my homework*

4. Fill out the last two columns of the following table, evaluating the value of each expression and identifying the data type the value is most likely to be:

Expression	Given	Value	Data Type

`6`		*6*	*int*
`True`		*True*	*Boolean*
`a`	`a = 2.5`	*2.5*	*Double*
`1 + 2 * 3`		*7*	*Int*
`a and False`	`a = True`	*False*	*Boolean*
`a or False`	`a = True`	*True*	*Boolean*

`a + b`	`a = 1`
	`b = 2`
		3	*Int*

`2 * a`	`a = 3`	*6*	*Int*

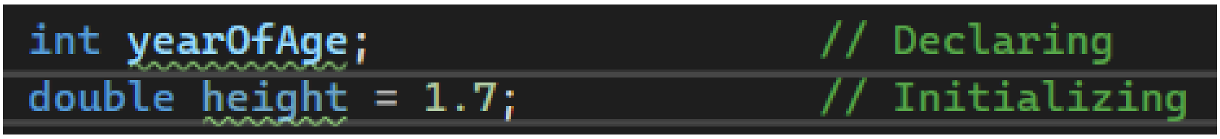
`a * 2 + b`	`a = 2.5`
	`b = 2`
		7.0	*Double*

`a + 2 * b`	`a = 2.5`
	`b = 2`
		6.5	*Double*

`(a + b) * c`	`a = 1`
	`b = 1`
	`c = 5`
		10	*Int*

`“Fred” + “ Smith”`		*Fred Smith*	*String*
`a + “ Smith”`	`a = “Wilma”`	*Wilma Smith*	*String*

5. Using an example, explain the difference between **declaring** and **initialising** a variable.
    a. *Declaring: Create a placeholder for a value by giving the name and the type of the variable*
    b. *Initialising: Assign the initial value to a variable, it happens after declaring.*

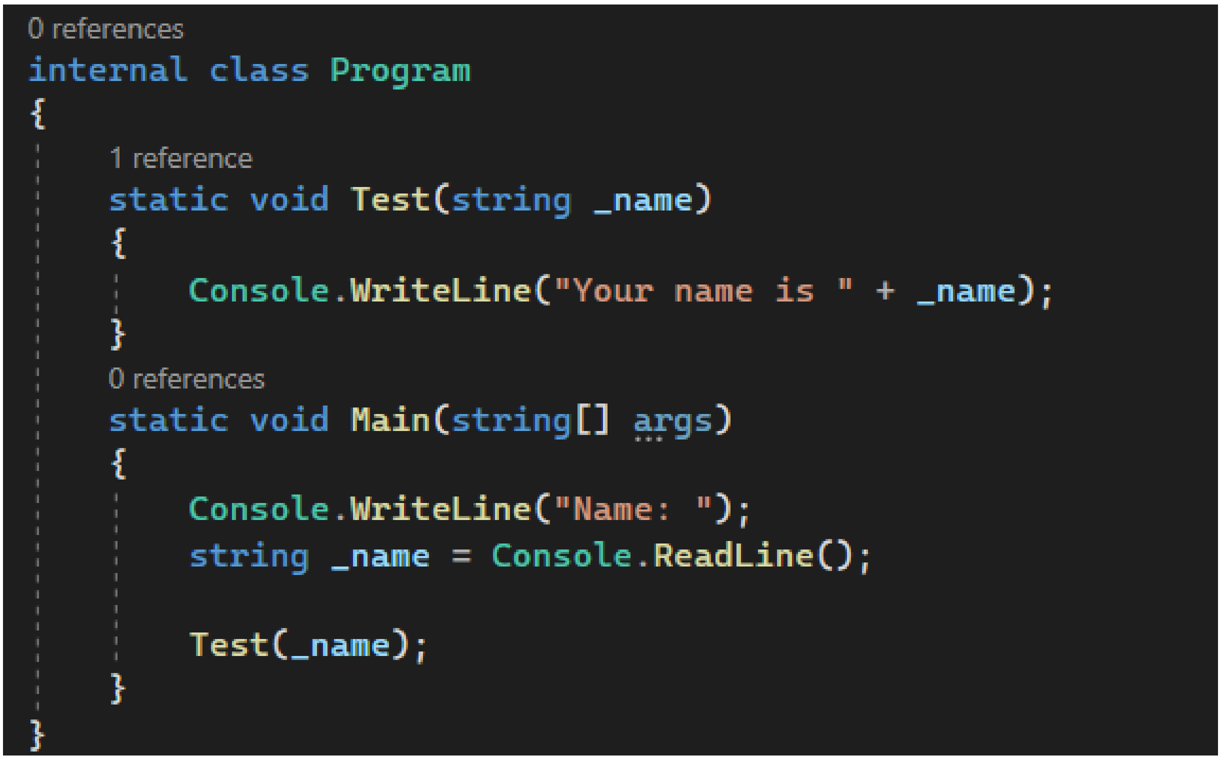


```
int yearOfAge;                    // Declaring
double height = 1.7;              // Initializing
```

6. Explain the term **parameter**. Write some code that demonstrates a simple of use of a parameter. You should show a procedure or function that uses a parameter, and how you would call that procedure or function.

A parameter is *a variable passed into a method, used to take arguments into methods.*

*This method of '*Test*' takes a single parameter named* _name*, ('string' type). The method uses this parameter to greet the person whose name is passed as an argument.*



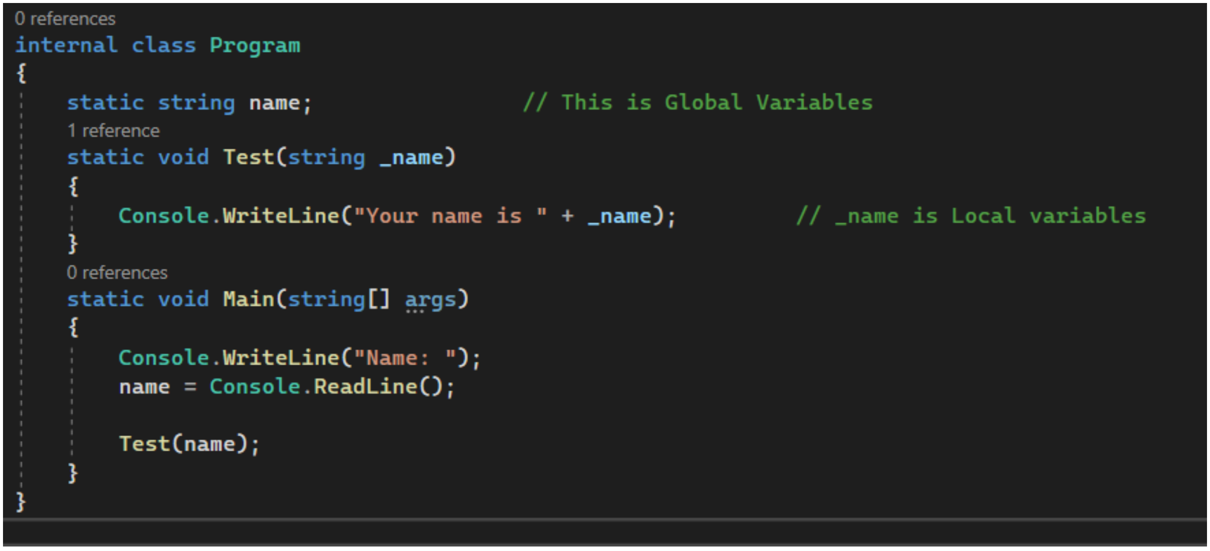
```
0 references
internal class Program
{
    1 reference
    static void Test(string _name)
    {
        Console.WriteLine("Your name is " + _name);
    }
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("Name: ");
        string _name = Console.ReadLine();

        Test(_name);
    }
}
```

7. Using an example, describe the term **scope** as it is used in procedural programming (not in business or project management). Make sure you explain the different kinds of scope.

Scope is *a term that refers to where variables can be accessed.*

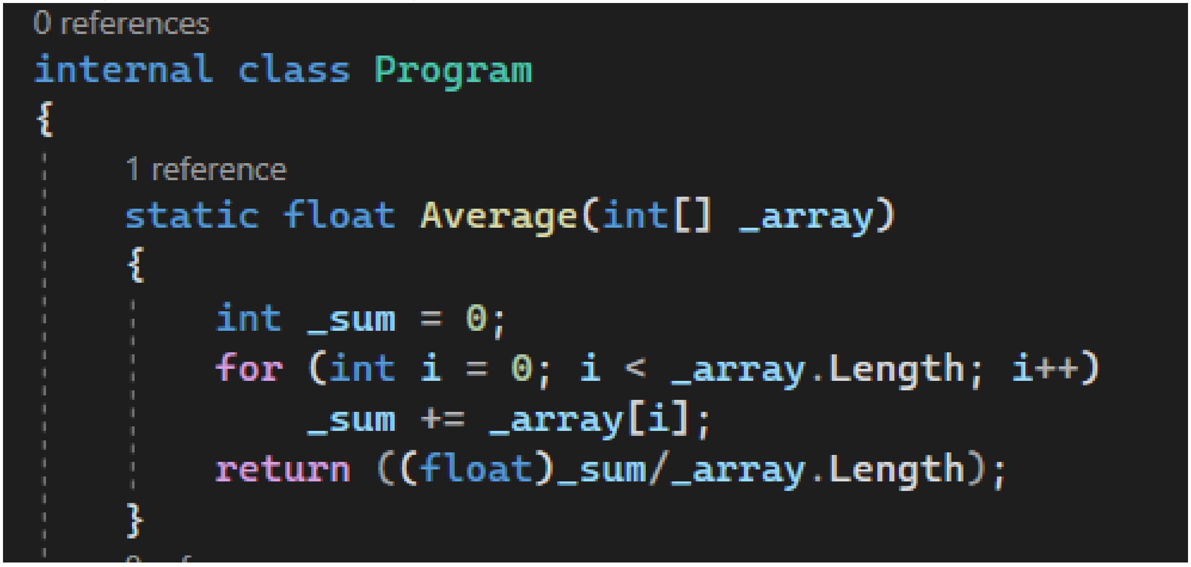
Local Scope: can only be used within the method

Global Scope: variables in the global scope can be used anywhere in the entire program



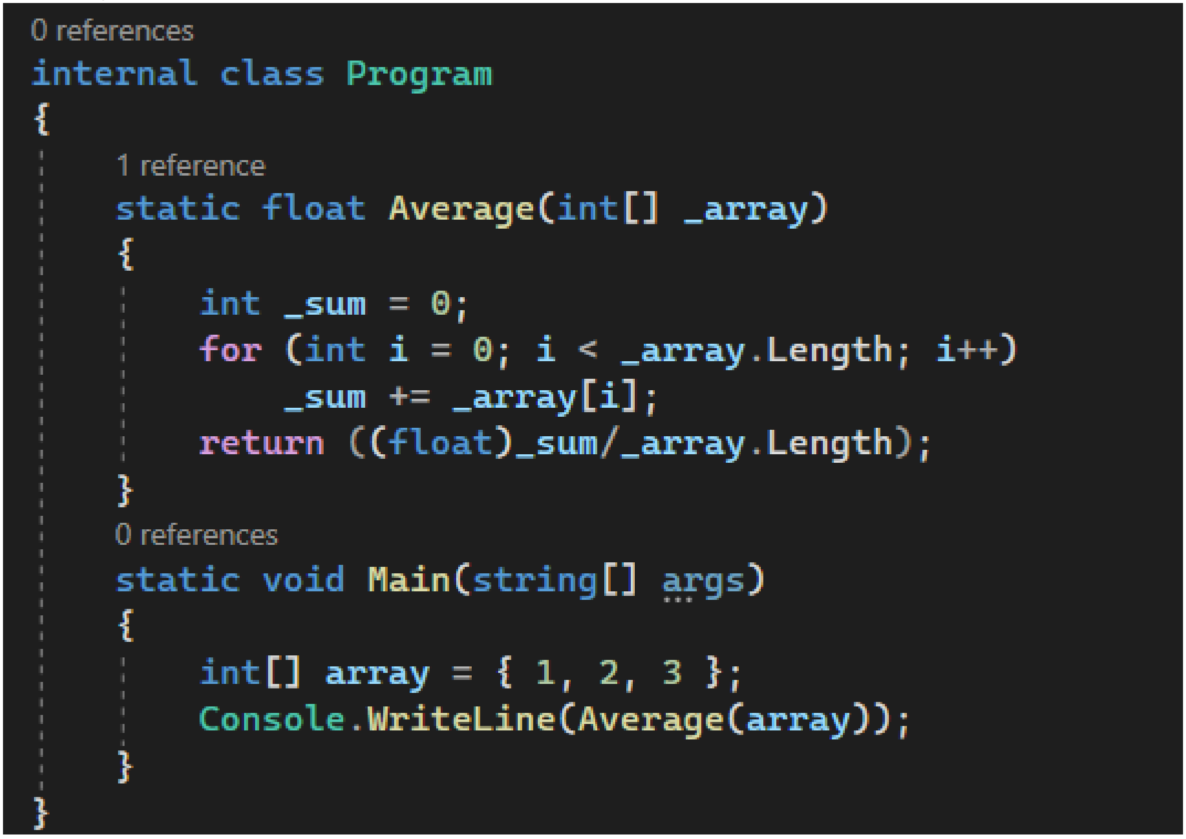
```
0 references
internal class Program
{
    static string name;                    // This is Global Variables
    1 reference
    static void Test(string _name)
    {
        Console.WriteLine("Your name is " + _name);         // _name is Local variables
    }
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("Name: ");
        name = Console.ReadLine();

        Test(name);
    }
}
```

8. In a procedural style, in any language you like, write a function called Average, which accepts an array of integers and returns the average of those integers. Do not use any libraries for calculating the average. You must demonstrate appropriate use of parameters, returning and assigning values, and use of a loop. Note — just write the function at this point, we'll *use* it in the next task. You shouldn't have a complete program or even code that outputs anything yet at the end of this question.



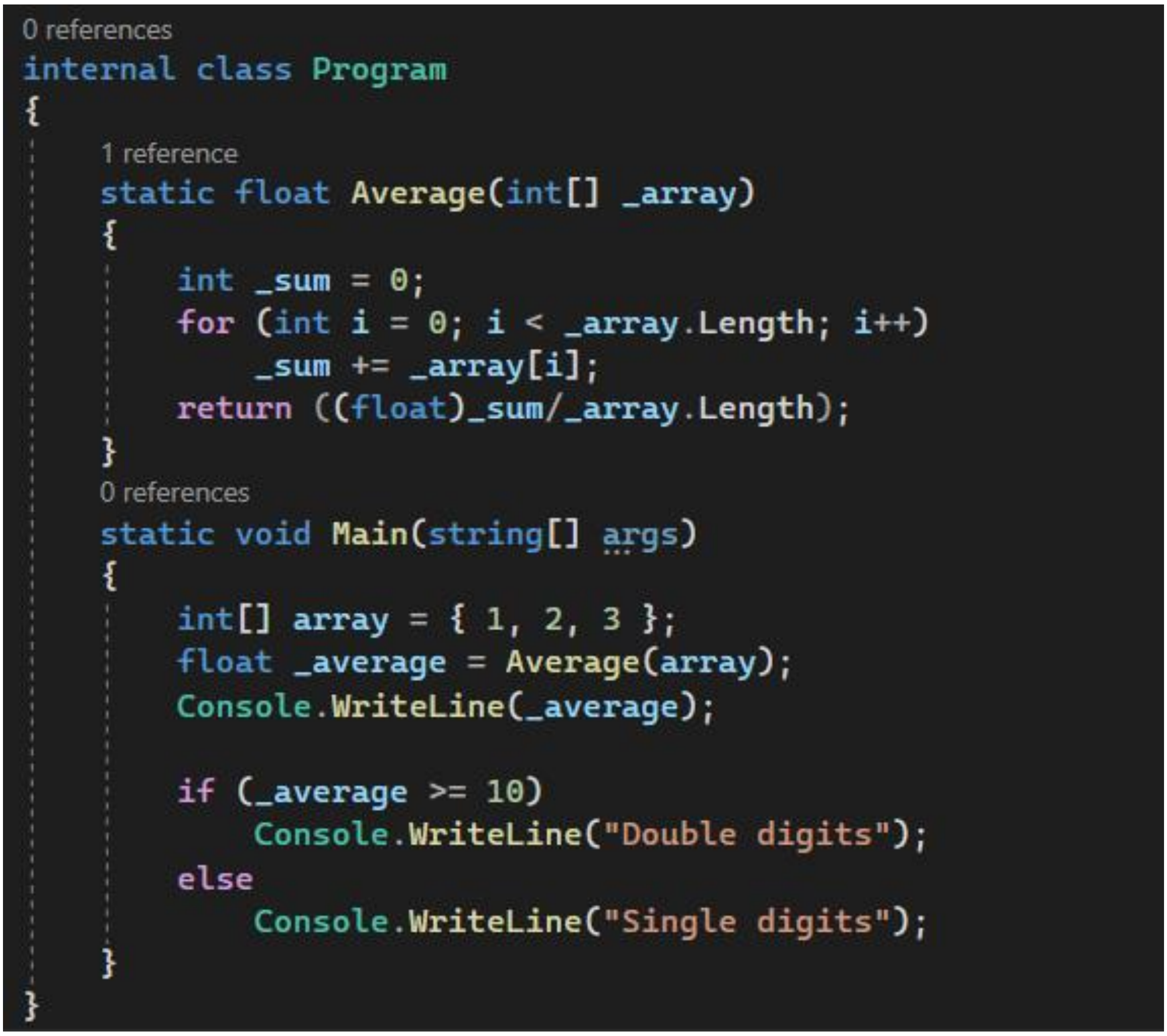
```
0 references
internal class Program
{
    1 reference
    static float Average(int[] _array)
    {
        int _sum = 0;
        for (int i = 0; i < _array.Length; i++)
            _sum += _array[i];
        return ((float)_sum/_array.Length);
    }
```

9. In the same language, write the code you would need to call that function and print out the result.



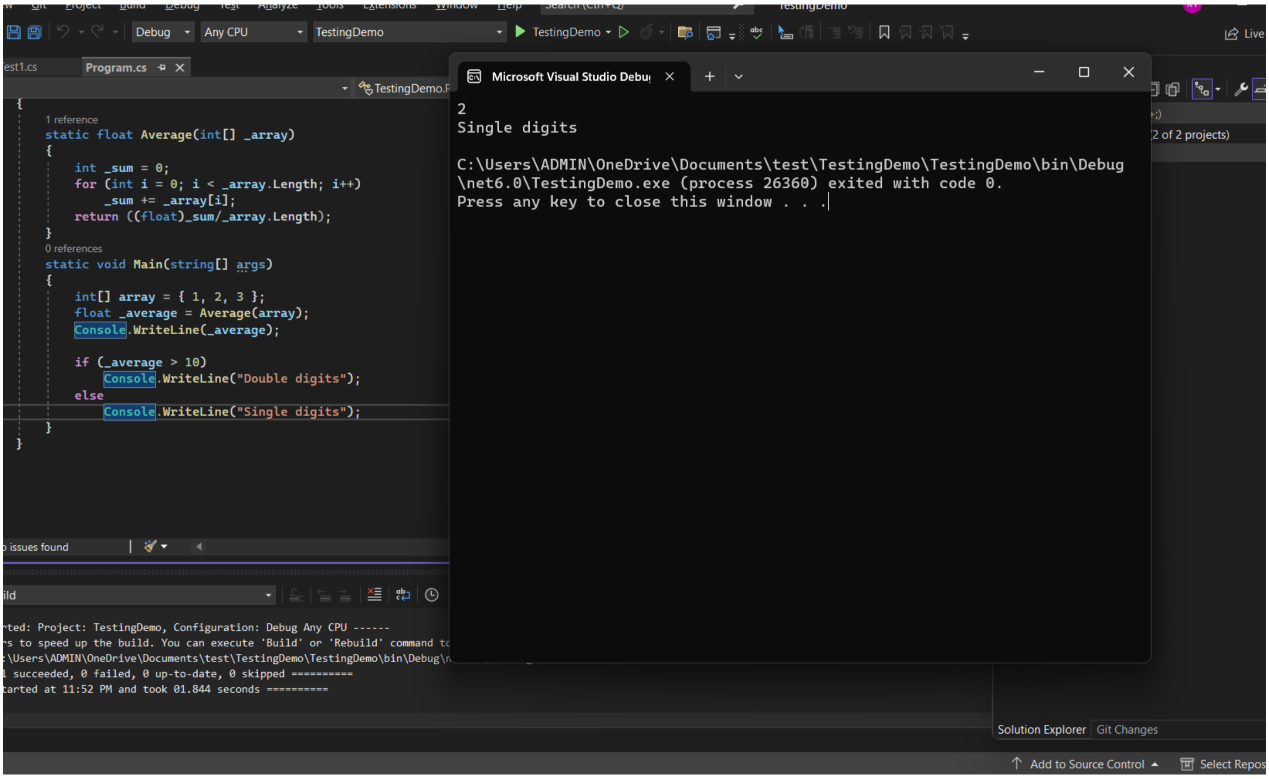
```
0 references
internal class Program
{
    1 reference
    static float Average(int[] _array)
    {
        int _sum = 0;
        for (int i = 0; i < _array.Length; i++)
            _sum += _array[i];
        return ((float)_sum/_array.Length);
    }
    0 references
    static void Main(string[] args)
    {
        int[] array = { 1, 2, 3 };
        Console.WriteLine(Average(array));
    }
}
```

10. To the code from 9, add code to print the message “Double digits” if the average is above or equal to 10. Otherwise, print the message “Single digits”. Provide a screenshot of your program running.



```
0 references
internal class Program
{
    1 reference
    static float Average(int[] _array)
    {
        int _sum = 0;
        for (int i = 0; i < _array.Length; i++)
            _sum += _array[i];
        return ((float)_sum/_array.Length);
    }
    0 references
    static void Main(string[] args)
    {
        int[] array = { 1, 2, 3 };
        float _average = Average(array);
        Console.WriteLine(_average);

        if (_average >= 10)
            Console.WriteLine("Double digits");
        else
            Console.WriteLine("Single digits");
    }
}
```

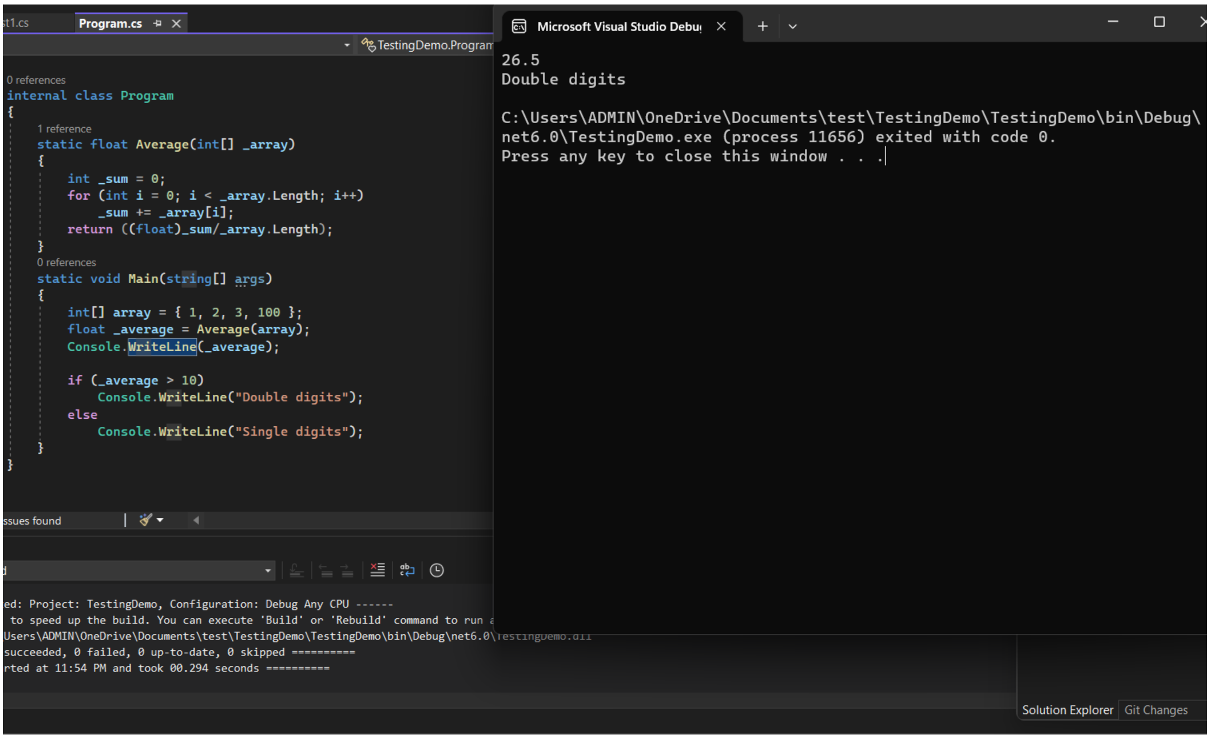
Screenshots of Output:



```
{
    1 reference
    static float Average(int[] _array)
    {
        int _sum = 0;
        for (int i = 0; i < _array.Length; i++)
            _sum += _array[i];
        return ((float)_sum/_array.Length);
    }
    0 references
    static void Main(string[] args)
    {
        int[] array = { 1, 2, 3 };
        float _average = Average(array);
        Console.WriteLine(_average);

        if (_average > 10)
            Console.WriteLine("Double digits");
        else
            Console.WriteLine("Single digits");
    }
}
```

```
2
Single digits

C:\Users\ADMIN\OneDrive\Documents\test\TestingDemo\TestingDemo\bin\Debug
\net6.0\TestingDemo.exe (process 26360) exited with code 0.
Press any key to close this window . . .
```

```
rted: Project: TestingDemo, Configuration: Debug Any CPU ------
rs to speed up the build. You can execute 'Build' or 'Rebuild' command to
:\Users\ADMIN\OneDrive\Documents\test\TestingDemo\TestingDemo\bin\Debug\n
1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
tarted at 11:52 PM and took 01.844 seconds ==========
```



```
0 references
internal class Program
{
    1 reference
    static float Average(int[] _array)
    {
        int _sum = 0;
        for (int i = 0; i < _array.Length; i++)
            _sum += _array[i];
        return ((float)_sum/_array.Length);
    }
    0 references
    static void Main(string[] args)
    {
        int[] array = { 1, 2, 3, 100 };
        float _average = Average(array);
        Console.WriteLine(_average);

        if (_average > 10)
            Console.WriteLine("Double digits");
        else
            Console.WriteLine("Single digits");
    }
}
```

```
26.5
Double digits

C:\Users\ADMIN\OneDrive\Documents\test\TestingDemo\TestingDemo\bin\Debug\
net6.0\TestingDemo.exe (process 11656) exited with code 0.
Press any key to close this window . . .
```

```
ed: Project: TestingDemo, Configuration: Debug Any CPU ------
 to speed up the build. You can execute 'Build' or 'Rebuild' command to run a
Users\ADMIN\OneDrive\Documents\test\TestingDemo\TestingDemo\bin\Debug\net6.0\TestingDemo.dll
succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
rted at 11:54 PM and took 00.294 seconds ==========
```