Objectives:

- **Statements & Operators**
- **Programming Construct**
- **Array**

1. Using selection construct: **if**

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class CollectionDemo1
    {
        public static void Main()
        {
            int reactorTemp = 1500;
            string emergencyValve = " ";

            if (reactorTemp < 1000)
            {
                System.Console.WriteLine("Reactor temperature normal");
            }
            else
            {
                System.Console.WriteLine("Reactor temperature too
high!");
                if (emergencyValve == "closed")
                {
                    System.Console.WriteLine("Reactor meltdown in
progress!");
                }
            }
```

```
            Console.ReadLine();
        }
    }
}
```

2. Using selection construct: **switch**

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class ConveyorControl
    {
        // enumerate the conveyor commands
        public enum action { start, stop, forward, reverse };

        public void conveyor(action com)
        {
            switch (com)
            {
                case action.start:
                    Console.WriteLine("Starting conveyor.");
                    break;
                case action.stop:
                    Console.WriteLine("Stopping conveyor.");
                    break;
                case action.forward:
                    Console.WriteLine("Moving forward.");
                    break;
                case action.reverse:
                    Console.WriteLine("Moving backward.");
                    break;
            }
        }
```

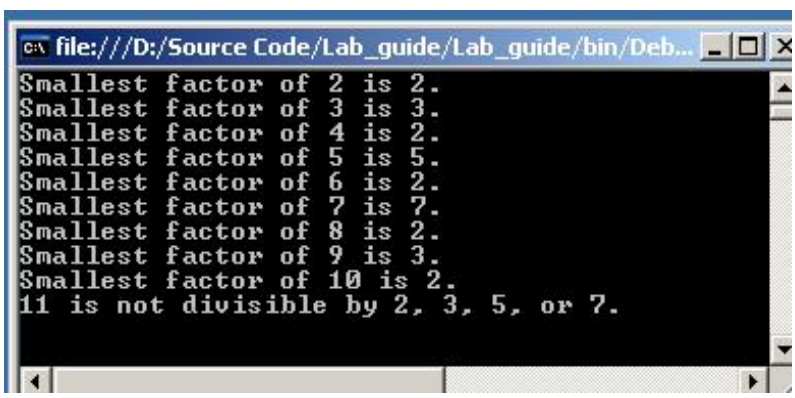
```
    }
    class SelectionDemo1
    {
        static void Main(string[] args)
        {
            ConveyorControl c = new ConveyorControl();

            c.conveyor(ConveyorControl.action.start);
            c.conveyor(ConveyorControl.action.forward);
            c.conveyor(ConveyorControl.action.reverse);
            c.conveyor(ConveyorControl.action.stop);
            Console.ReadLine();
        }
    }
}
```

3. Using for loop construct:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class SelectionDemo2
    {
        static void Main(string[] args)
        {

            int num;

            for (num = 2; num < 12; num++)
            {
                if ((num % 2) == 0)
                    Console.WriteLine("Smallest factor of " + num + "
is 2.");
                else if ((num % 3) == 0)
                    Console.WriteLine("Smallest factor of " + num + "
is 3.");
```

```
                else if ((num % 5) == 0)
                    Console.WriteLine("Smallest factor of " + num + "
is 5.");
                else if ((num % 7) == 0)
                    Console.WriteLine("Smallest factor of " + num + "
is 7.");
                else
                    Console.WriteLine(num + " is not divisible by 2, 3,
5, or 7.");
            }



            Console.ReadLine();
        }
    }
}
```

The output of the program:



4. Using for each loop:

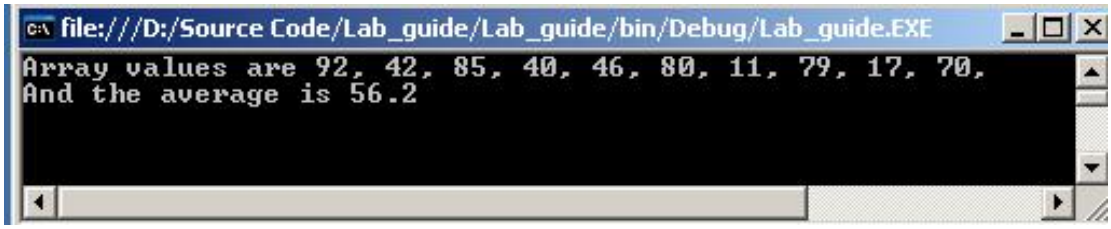
```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
```

```
    class SelectionDemo1
    {
        static void Main(string[] args)
        {
            int[] nums = new int[10];
            int val;
            bool found = false;

            // give nums some values
            for(int i = 0; i < 10; i++)
            nums[i] = i;

            val = 5;

            // use foreach to search nums for key
            foreach(int x in nums)
            {
                if(x == val)
                {
                    found = true;
                    break;
                }
            }

            if(found)
            Console.WriteLine("Value found!");




            Console.ReadLine();
        }
    }
}
```

5. Using for each loop

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class SelectionDemo1
    {
        static void Main(string[] args)
        {
            DateTime now = DateTime.Now;
            Random rand = new Random((int)now.Millisecond);
            int[] Arr = new int[10];
            for (int x = 0; x < Arr.Length; ++x)
            {
                Arr[x] = rand.Next() % 100;
            }
            int Total = 0;
            Console.Write("Array values are ");
            foreach (int val in Arr)
            {
                Total += val;
                Console.Write(val + ", ");
            }
            Console.WriteLine("\nAnd the average is {0,0:F1}",
                              (double)Total / (double)Arr.Length);



            Console.ReadLine();
        }
    }
}
```

The output of the program:

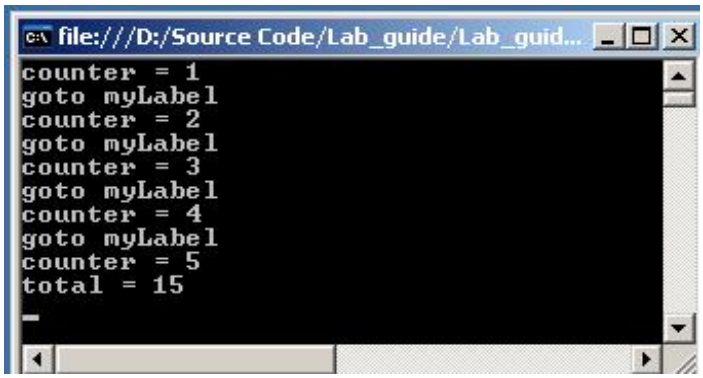


6. jump statement: goto

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class SelectionDemo1
    {
        static void Main(string[] args)
        {
            int total = 0;
            int counter = 0;

            myLabel:
            counter++;
            total += counter;
            System.Console.WriteLine("counter = " + counter);
            if (counter < 5)
            {
                System.Console.WriteLine("goto myLabel");
                goto myLabel;
            }
            System.Console.WriteLine("total = " + total);



            Console.ReadLine();
        }
```

```
    }
}
```

The output of the program:

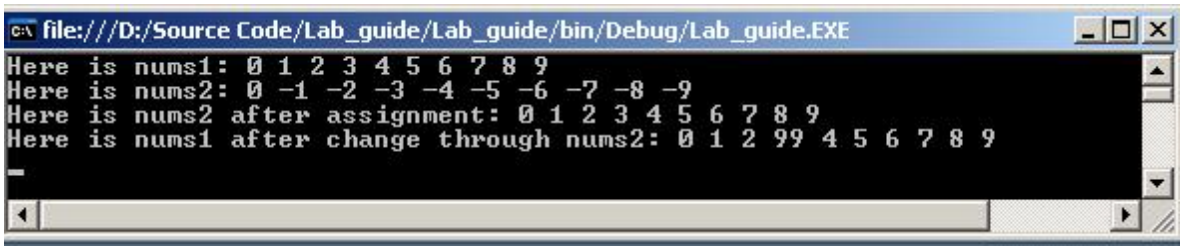


6. Using array, type this below code:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class ArrayDemo1
    {
        static void Main(string[] args)
        {
            int i;
            int[] nums1 = new int[10];
            int[] nums2 = new int[10];

            for (i = 0; i < 10; i++) nums1[i] = i;

            for (i = 0; i < 10; i++) nums2[i] = -i;

            Console.Write("Here is nums1: ");
            for (i = 0; i < 10; i++)
                Console.Write(nums1[i] + " ");
```

```
            Console.WriteLine();

            Console.Write("Here is nums2: ");
            for (i = 0; i < 10; i++)
                Console.Write(nums2[i] + " ");
            Console.WriteLine();
            nums2 = nums1; // now nums2 refers to nums1
            Console.Write("Here is nums2 after assignment: ");
            for (i = 0; i < 10; i++)
                Console.Write(nums2[i] + " ");
            Console.WriteLine();
            // now operate on nums1 array through nums2
            nums2[3] = 99;
            Console.Write("Here is nums1 after change through nums2:
");
            for (i = 0; i < 10; i++)
                Console.Write(nums1[i] + " ");
            Console.WriteLine();

            Console.ReadLine();
        }
    }
}
```

The output of the program:

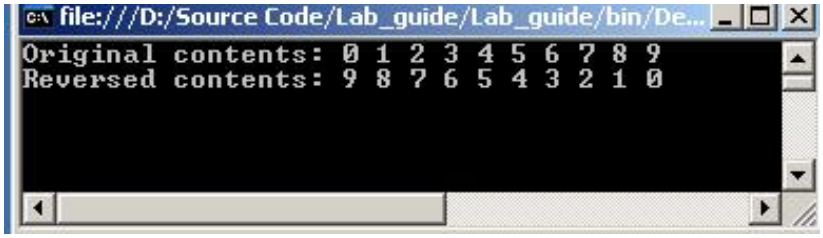


7. Create an array and reverse this array

```
using System;
using System.Collections.Generic;
using System.Text;
```

```
namespace Lab_guide
{
    class ArrayDemo2
    {
        static void Main(string[] args)
        {
            int i, j;
            int[] nums1 = new int[10];
            int[] nums2 = new int[10];

            for (i = 0; i < nums1.Length; i++) nums1[i] = i;

            Console.Write("Original contents: ");
            for (i = 0; i < nums2.Length; i++)
                Console.Write(nums1[i] + " ");

            Console.WriteLine();

            // reverse copy nums1 to nums2
            if (nums2.Length >= nums1.Length) // make sure nums2 is
long enough
                for (i = 0, j = nums1.Length - 1; i < nums1.Length;
i++, j--)
                    nums2[j] = nums1[i];

            Console.Write("Reversed contents: ");
            for (i = 0; i < nums2.Length; i++)
                Console.Write(nums2[i] + " ");

            Console.WriteLine();

            Console.ReadLine();
        }
    }
}
```

The output of the program:

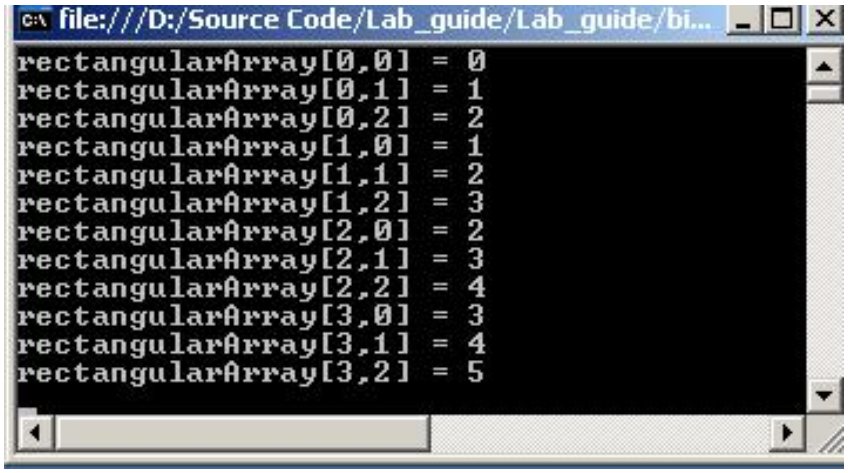


8. Using multi-dimensional array:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class ArrayDemo3
    {
        static void Main(string[] args)
        {
            const int rows = 4;
            const int columns = 3;

            // declare a 4x3 integer array
            int[,] rectangularArray = new int[rows, columns];

            // populate the array
            for (int i = 0; i < rows; i++)
            {
                for (int j = 0; j < columns; j++)
                {
                    rectangularArray[i, j] = i + j;
                }
            }

            // report the contents of the array
            for (int i = 0; i < rows; i++)
            {
                for (int j = 0; j < columns; j++)
                {
```

```
                    Console.WriteLine("rectangularArray[{0},{1}] =
{2}",
                        i, j, rectangularArray[i, j]);
                }
            }
            Console.ReadLine();
        }
    }
}
```

The output of the program:



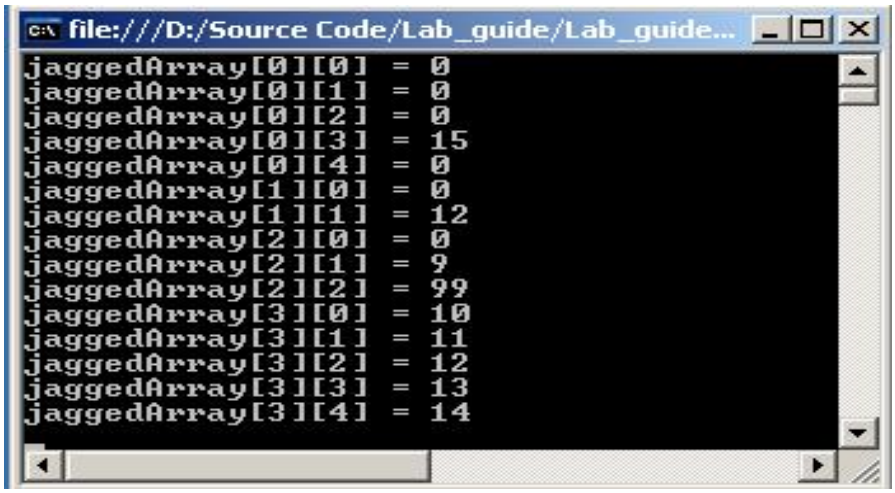
9. Using jagged array, type this below code:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class ArrayDemo4
    {
        static void Main(string[] args)
        {
            const int rows = 4;
            const int rowZero = 5;  // num elements
            const int rowOne = 2;
```

```
            const int rowTwo = 3;
            const int rowThree = 5;

            // declare the jagged array as 4 rows high
            int[][] jaggedArray = new int[rows][];

            // declare the rows of various lengths
            jaggedArray[0] = new int[rowZero];
            jaggedArray[1] = new int[rowOne];
            jaggedArray[2] = new int[rowTwo];
            jaggedArray[3] = new int[rowThree];

            // Fill some (but not all) elements of the rows
            jaggedArray[0][3] = 15;
            jaggedArray[1][1] = 12;
            jaggedArray[2][1] = 9;
            jaggedArray[2][2] = 99;
            jaggedArray[3][0] = 10;
            jaggedArray[3][1] = 11;
            jaggedArray[3][2] = 12;
            jaggedArray[3][3] = 13;
            jaggedArray[3][4] = 14;

            for (int i = 0; i < rowZero; i++)
            {
                Console.WriteLine("jaggedArray[0][{0}] = {1}",
                    i, jaggedArray[0][i]);
            }

            for (int i = 0; i < rowOne; i++)
            {
                Console.WriteLine("jaggedArray[1][{0}] = {1}",
                    i, jaggedArray[1][i]);
            }

            for (int i = 0; i < rowTwo; i++)
            {
```

```
                Console.WriteLine("jaggedArray[2][{0}] = {1}",
                    i, jaggedArray[2][i]);
            }
            for (int i = 0; i < rowThree; i++)
            {
                Console.WriteLine("jaggedArray[3][{0}] = {1}",
                    i, jaggedArray[3][i]);
            }
            Console.ReadLine();
        }
    }
}
```

The output of the program:



10. Using properties and methods of "Array" class. Type this below code:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class ArrayDemo5
```

```
    {
        static void Main(string[] args)
        {
            int[] Arr = new int[12] { 29, 82, 42, 46, 54, 65, 50, 42,
5, 94, 19, 34 };
            Console.WriteLine("The first occurrence of 42 is at index "
                                + Array.IndexOf(Arr, 42));
            Console.WriteLine("The last occurrence of 42 is at index "
                                + Array.LastIndexOf(Arr, 42));
            int x = 0;
            while ((x = Array.IndexOf(Arr, 42, x)) >= 0)
            {
                Console.WriteLine("42 found at index " + x);
                ++x;
            }
            x = Arr.Length - 1;
            while ((x = Array.LastIndexOf(Arr, 42, x)) >= 0)
            {
                Console.WriteLine("42 found at index " + x);
                --x;
            }

            Console.WriteLine("Array that befor sorted");
            for (int i = 0; i < Arr.Length; i++)
            {
                Console.WriteLine("{0} :      {1}",i+1,Arr[i]);
            }
            Array.Sort(Arr);
            Console.WriteLine("Array that after sorted");
            for (int i = 0; i < Arr.Length; i++)
            {
                Console.WriteLine("{0} :      {1}", i + 1, Arr[i]);
            }
            Array.Reverse(Arr);
            Console.WriteLine("Array that after reserse");
            for (int i = 0; i < Arr.Length; i++)
            {
```

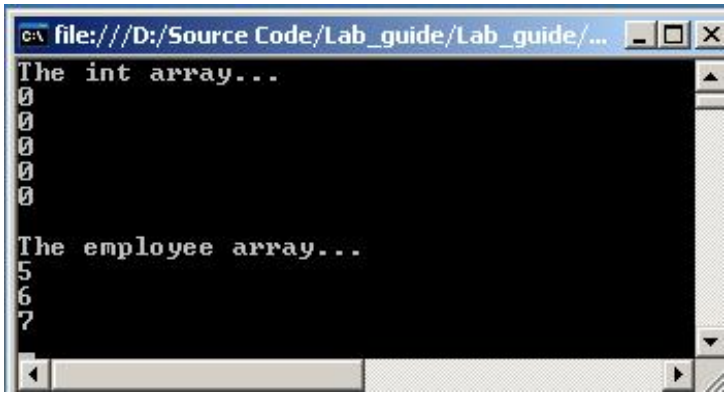
```
                Console.WriteLine("{0} :      {1}", i + 1, Arr[i]);
            }
                Console.ReadLine();
        }
    }
}
```

11. A simple class to store in the array, type this below code:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab_guide
{
    class Employee
    {
        private int empID;

        // constructor
        public Employee(int empID)
        {
            this.empID = empID;
        }
        public override string ToString()
        {
            return empID.ToString();
        }
    }

    class ArrayDemo6
    {
        public void Run()
        {
            int[] intArray;
            Employee[] empArray;
            intArray = new int[5];
```

```
            empArray = new Employee[3];

            // populate the array
            for (int i = 0; i < empArray.Length; i++)
            {
                empArray[i] = new Employee(i + 5);
            }

            Console.WriteLine("The int array...");
            for (int i = 0; i < intArray.Length; i++)
            {
                Console.WriteLine(intArray[i].ToString());
            }

            Console.WriteLine("\nThe employee array...");
            for (int i = 0; i < empArray.Length; i++)
            {
                Console.WriteLine(empArray[i].ToString());
            }
        }

        static void Main(string[] args)
        {
            ArrayDemo6 arr = new ArrayDemo6();
            arr.Run();
            Console.ReadLine();
        }
    }
}
```

The output of the program:



Do it yourself

2.1. Write a program to sum the value of integer numbers that are specified via parameters. You should use the foreach loop. Before displaying the sum result, show the integer array in ascesding order by using the Array class.

2.2. Design and code a class named Atom that holds information about a single atom. Place your class definition in a file named Atom.cs

Include the following member functions in your design:
+ boolean accept() - prompts for and accepts from standard input
    - an integer holding the atomic number,
    - a string holding the atomic symbol,
    - a string holding the full name of the atom and
    - a floating-point value holding the atomic weight.

If any input is invalid, your function rejects that input and requests fresh data.
+ void display() - displays the atomic information on standard output.

Design and code a main program that accepts information for up to 10 atomic elements and displays the atomic information in tabular format.

The program output might look something like:

```
Atomic Information
==================
Enter atomic number : 3
Enter symbol : Li
Enter full name : lithium
Enter atomic weight : 6.941
```

```
Enter atomic number : 20
Enter symbol : Ca
Enter full name : calcium
Enter atomic weight : 40.078

Enter atomic number : 30
Enter symbol : Zn
Enter full name : zinc
Enter atomic weight : 65.409

Enter atomic number : 0

 No Sym Name                  Weight
-----------------------------------
  3 Li  lithium                6.941
 20 Ca  calcium               40.078
 30 Zn  zinc                  65.409
```

2.3. Do Workshop 3, 4, 5 in CD.

2.4. Do ACTCSharp_Module3_Assignment.pdf in CD

2.5. Do ACTCSharp_Module4_Assignment.pdf in CD

2.6. Do ACTCSharp_Module5_Assignment.pdf in CD

References

1) **CD ROM C# Programming, Aptech Education**
2) **http://www.java2s.com/Tutorial/CSharp/CatalogCSharp.htm**
3) **MSDN Document**
4) **[ebook] MSDN training, Introduction to C#, Microsoft Press**