Laboratory -2: Basic of C# Programming

Objectives:

O[1]. To debug a C# program and trace the errors in a program.
O[2]. To learn class library(.dll) creation, reference as assembly, and import as a namespace.
O[3]. To learn how to use the console to take input and display output.
O[4]. To implement an array, and learn how to pass array in methods.

In this laboratory works, students are going to learn program writing in C# console-based platform, to understand the debugging facilities of Visual Studio 2019, to trace the errors in a program, and solve them. Students will learn how to create class libraries, increase reusability by using the .dll in another project.

Task1: Understand and run the following program(Program1) as ConsoleAPP(ASP.Net Core)

Program 1:

```
namespace Lab2Ex1
{
    class Program
    {
       static double Add(double a, double b)
        {
            return a * b; // deliberate bug!
        }
        static void Main(string[] args)
        {
            double a = 4.5; // or use var
            double b = 2.5;
            double answer = Add(a, b);
           // Person p = new Person();
            Console.WriteLine($"{a} + {b} = {answer}");
            Console.ReadLine(); // wait for user to press ENTER
        }    }
}
```

a) Debug the program using Toggle Breakpoint, Step Into and other debug options.
b) Execute the program using Visual Studio 2019 Editor.

The $ special character identifies a string literal as an **interpolated string**. An interpolated string is a string literal that might contain interpolation expressions. When an interpolated string is resolved to a result string, items with interpolation expressions are replaced by the string representations of the expression results.

```
string name = "Mark";
var date = DateTime.Now;

// Composite formatting:
Console.WriteLine("Hello, {0}! Today is {1}, it's {2:HH:mm} now.", name, date.DayOfWeek, date);
```

```
// String interpolation:
Console.WriteLine($"Hello, {name}! Today is {date.DayOfWeek}, it's {date:HH:mm} now.");

// Both calls produce the same output that is similar to:
// Hello, Mark! Today is Wednesday, it's 19:40 now.
```

Task2: Class library(.dll) creation, reference as assembly, and import as namespace

a) Create two namespaces in two different projects and import them into the third project and use the imported namespaces methods.

b) Apply your knowledge into program1 and use Add methods from separate .dll. Explain the code reusability.

Task3: Use the console to take input to a C# program and display contents to the console.

Problem2:

Read a three digit integer input from the console, and print the reverse of that number in the console.

- You can call a reverse() method from Main() and pass the integer values.
- Reverse() method will return the reverse value to Main().
- Main() will catch and print the value.

i=Convert.ToInt32(Console.ReadLine());
or
int res;
string myStr = "200";
res = int.Parse(myStr);

Convert.ToInt32() takes an object as its argument. Convert.ToInt32() also does not throw ArgumentNullException when its argument is null the way Int32.Parse() does. That also means that Convert.ToInt32() is probably a wee bit slower than Int32.Parse(). Convert.ToInt32() calls int.Parse() internally. Except for one thing Convert.ToInt32() returns 0 when argument is null, Otherwise both work the same way.

Task4: Understand the following program(Program3) and run it as ConsoleAPP(ASP.Core)

Problem3:
Take 10 inputs from console and print their average in console.

- You should use separate function to take input and find average.
- Main() will call those functions and print the average.

type [ ] array_name= new type[size];
int [ ] sample = new int [10];
int [ ] sample; sample=new int [10];

In C++ arrays are not valid as return type. However, in C# arrays are implemented as objects, a method can also return as an array.

Considering the following code, an array is a reference type and so for this function:

public static void FirstDouble(int[] array)
the variable array is a reference because int[] is a reference type. So array is a *reference* that is *passed by value*.

Thus, modifications made to the array inside the function are applied to the int[] object to which the array refers. And so those modifications are visible to all references that refer to that same object. And that includes the reference that the caller holds.

Now, if we look at the implementation of this function:

```
public static void FirstDouble(int[] array)
{
   //double each elements value
   for (int i = 0; i < array.Length; i++)
      array[i] *= 2;

   //create new object and assign its reference to array
   array = new int[] { 11, 12, 13 };
}
```

If the function had been declared like this:

public static void FirstDouble(ref int[] array)

then the reference array would have been passed by reference and the caller would see the newly created object { 11, 12, 13 } when the function returned

Task5: Understand the following program(Program4) and run it as ConsoleAPP(ASP.Core)

Problem3:
Write a program so that if the user gives input ‘d’ or ‘D’; ‘k’ or ‘K’ and ‘s’ or ‘S’ the program will print Dhaka, Khulna, and Sylhet respectively.

- You need to use a switch statement.
- Dhaka, Khulna, and Sylhet will be stored in a string array.