LAB 6A

You cannot use multiple return statements in any method submitted for this project. Additionally, the use of var is not permitted. You must write the necessary programming statements/logic to accomplish each task. No credit will be given for using existing methods (thereby avoiding the need to write the programming statements) to accomplish the assigned tasks.

Test 1 – Convert char array to int array

```
public static int[] Test1(char [] phrase)
```

Given an array of char, `phrase`, cast/convert each element to an equivalent int value and place in an int array. Return the int array

Example input

```
{ 'A','B','C' }
```

Example output

```
{ 65,66,67 }
```

Test 2 – Array statistics

```
public static double [] Test2(double [] data)
```

Given an array of double, `data`, find the smallest element, the largest element and the numeric mean (average). Store the results in an array (in that order: smallest, largest, mean). Return the array. You are not allowed to use any methods in the Array class to accomplish these tasks.

Example input

```
{ 9.0, 11.0, 4.0 }
```

Example output

```
{ 4.0, 11.0, 6.0 }
```

Test 3 – Normalize an array

```
public static void Test3(double [] numbers)
```

Given an array of double, `numbers`, normalize the array. To normalize an array:

1) Find the largest element stored in the array
2) Divide each element in the array by the largest value

and replace each array element with the result of the division. Since array's are reference types and the array's contents are being modified, there is nothing to return

Example input

```
{ 7.0, 3.5, 1.75 }
```

Example output

```
{ 1.0, 0.5, 0.25 }
```

Test 4 – Uniqueness

```
public static bool Test4(string [] names)
```

Given an array of string, `names`, verify that each name is unique mean that none of the names are duplicated within the array. If the array is unique, return true; otherwise, return false

Tip: Use 2 for loops (nested) to accomplish this task.

Example input

```
{ "Bob", "Fred", "Harry", "Bob" }
```

Example output

```
false
```

Test 5 – Acronym

```
public static string Test5(string [] words)
```

Given an array of string, `words`, create a string that is the acronym (first letter of each word) using the concatenation operator (+). Return the string.

Example input

```
    { "World", "of", "Wonder" }
```

Example output

```
    "WoW"
```

Test 6 – Array reverse

```
public static char [] Test6(char [] letters)
```

Given a char array, `letters`, create another array that has the same elements but in reverse order. Return the array
You are not allowed to use Array.Reverse (or any existing method) to reverse the array

Example input

```
    { 'a','b','c' }
```

Example output

```
    { 'c','b','a' }
```

Test 7 – Transpose array

```
public static int[,] Test7(int [,] table)
```

Given a 2-Dimension array of int, `table`, create a new array that the original array. Transposing means that each row in the original array will be a column in the new array and each column in the original array will be a row in the new array.

Example input

```
4 3 1 5
2 7 0 8
```

Example output

```
4 2
3 7
1 0
5 8
```

Test 8 – Return a 2D array

```
public static int [,] Test8(int [] mins, int [] maxes, int [] seeds)
```

Given three arrays of the same type (int) and size, `mins`, `maxes` and `seeds`, combine the arrays into a single 2D array. Return the 2D array
Tip: This task only requires a single loop (not 3)

Example input

```
    { 1, 2, 3 } / { 7, 8, 9 } / { 5, 10, 15 }
```

Example output

```
    { { 1, 2, 3 }, { 7,8,9 }, { 5,10,15 } }
```

Test 9 – Convert int array to char array

```
public static char [] Test9(int [] ascii)
```

Given an array of int, `ascii`, cast/convert each element to an equivalent char value and place in a char array. Return the char array

Example input

```
{ 65, 67, 79 }
```

Example output

```
{ 'A', 'C', 'E' }
```

Test 10 – Modify an existing array

```
public static void Test10(char [] word)
```

Given an array of char (all uppercase), `word`, modify the array so that every other element will be lowercase (even indexes are upper, odd indexes are lower). You can either use the ToLower method of the Char class or you can add 32 to the char and then cast the resulting value back to a char.

Example input

```
GENER
```

Example output

```
GeNeR
```