1. **What is the expected running time of the following C# code? Explain why. Assume the array's size is n.**

**long Compute(int[] arr)**

**{**

**long count = 0;**

**for (int i=0; i<arr.Length; i++)**

**{**

**int start = 0, end = arr.Length-1;**

**while (start < end)**

**if (arr[start] < arr[end])**

**{ start++; count++; }**

**else**

**end--;**

**}**

**return count;**

**}**

Runs in O(n2).

The program goes through all elements from the array once in the “for” cycle and then again in the “while” loop. In the “while” loop there is no matter, whether the program goes through the “if” or the “else” statement, because in each iteration the “start” variable is incremented or the “end” variable is decremented. In both cases the program goes through each element from the array once more time. Therefore the running time is O(n2).

1. **What is the expected running time of the following C# code? Explain why.**   
     
     
     
     
     
     
     
     
     
     
     
     
     
   **Assume the input matrix has size of n \* m.**

**long CalcCount(int[,] matrix)**

**{**

**long count = 0;**

**for (int row=0; row<matrix.GetLength(0); row++)**

**if (matrix[row, 0] % 2 == 0)**

**for (int col=0; col<matrix.GetLength(1); col++)**

**if (matrix[row,col] > 0)**

**count++;**

**return count;**

**}**

Runs in O(n\*m).

The program goes through the elements from the two-dimensional array. It always goes through the first dimension of the matrix and only if it goes in the “if” statement, it goes through the second dimension of the array. Therefore the running time is O(n\*m) in average and worst case, but if all elements from the first column are odd numbers the running time is O(n), because the program never goes in the “if” statement and through the second “for” loop.

1. **What is the expected running time of the following C# code? Explain why.   
     
     
     
     
     
     
     
     
     
     
     
     
     
   Assume the input matrix has size of n \* m.**

**long CalcSum(int[,] matrix, int row)**

**{**

**long sum = 0;**

**for (int col = 0; col < matrix.GetLength(0); col++)**

**sum += matrix[row, col];**

**if (row + 1 < matrix.GetLength(1))**

**sum += CalcSum(matrix, row + 1);**

**return sum;**

**}**

**Console.WriteLine(CalcSum(matrix, 0));**

Runs in O(n\*m).

The program goes through the elements from the two-dimensional array. It always goes through the first dimension of the matrix and only if it goes through the “if” statement, the method calls itself recursively. The method is repeated m-1 times according to the “if” statement, but 1 is constant, so it is ignored. Therefore the running time is O(n\*m).