**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;  
}  
  
Външният цикъл се изпълнява n пъти, като на всяка итерация изпълняваме вътрешния също n пъти, т.е. сложността е O(n\*n).  
  
**2 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;  
}  
Running time-а е: O(n-х + х\*m), където х е броя четни числа в първата колона на matrix.   
  
**3 \* 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));

Ако броят редове n е по-голям от броя колони m, в първия цикъл ще бъде хвърлен ексепшън; иначе програмата ще го изпълни n пъти, след което ще рекурсира - общият брой рекурсивни извиквания ще е n (докато row не излезе извън матрицата) => O(n^2).