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;
}
```

It runs in $O(n^2)$. There are two nested loops (*for* and *while*) iterating individually through the total array length or "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;
}
```

There are two nested loops. Each one is iterating through the total length of the columns and rows of the matrix. I'm not sure but I think that in the **worst case** it runs in quadratic time O(n*m). The worst case is when the first cell in each row can be divided by 2. That way both loops will be executed. **Best case** is when the first cell in each row can NOT be divided by 2. Then the algorithm will never enter the second loop and the running time gets linear O(n). The average case should be something in between.