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;

}

Тук сложността на кода е n2, защото имаме 2 вложени цикъла, единият от които върти върти от 0 до arr.length, другият – от start=0 до end=arr.length, следователно и двата са еднакви и въртят до дължината на масива, което е n.

1. **What is the expected running time of the following C# code? Explain why.**

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;

}

**Assume the input matrix has size of n \* m.**

Тук сложността на кода е n\*m, защото имаме отново 2 вложени цикъла, единият от които върти до n, другия до m и въпреки че втория цикъл се изпълнява само, ако if statement-a е изпълнен, в най-лошия случай това условие винаги е вярно => в най-лошия случай вътрешния цикъл винаги се изпълнява.

1. **\* What is the expected running time of the following C# code? Explain why.**

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));

**Assume the input matrix has size of n \* m.**

Тук сложността на алгоритъма отново е n\*m, защото имаме един цикъл със сложност n, който се изпълнява рекурсивно, ако if statement-a е верен. Той от своя страна се изпълнява m пъти, защото започва от row = 0, до row < matrix.GetLength(1) = m.