# Segment 1 solution:

int n = Int32.Parse(Console.ReadLine());

LinkedList<int> sumSequence = new LinkedList<int>();

            sumSequence.AddLast(0);

            sumSequence.AddLast(1);

            for (int i = 2; i < n; i++)

            {

                int currentSum = 0;

                for (int j = i - 1; j > 0; j--)

                {

                    currentSum += sumSequence.ElementAt(j);

                }

                sumSequence.AddLast(currentSum);

            }

            for (int i = 0; i < sumSequence.Count(); i++)

            {

                Console.WriteLine(sumSequence.ElementAt(i));

            }

LinkedList called sumSequence’s initialization as well as the two sumSequence.AddLast functions are simple operations and are also in constant-time so the most signification term is a constant (1) without (n).

First for loop in for (int i = 2; i < n; i++) with n running two times has a frequency count of n+1 and int currentSum = 0; has a frequency count of n.

The nested for loop for (int j = i - 1; j > 0; j--)  within the first for loop runs i 1 time. The frequency count of the nested for loop is n(n+1) which is n^2+n

The third for loop for (int i = 0; i < sumSequence.Count(); i++) with the frequency count of n since the loop runs as many times as there are elements in sumSequence

Hence, the total counts of the frequency count is 1+(n+1)+n+n^2+n+n = n^2+4n+2.

Discarding constant terms = n^2+4n.

Clearing coefficients = n^2+n

Picking the most significant term = n^2

Big O notation= O(n^2)

# Segment 2 solution:

LinkedList<string> names = new LinkedList<string>();

            string input = "";

            Console.WriteLine("Please enter who is attending!");

            Console.WriteLine("Type '!' to finish name entry...");

            Console.WriteLine("---");

            while (!input.Equals("!")){

                Console.WriteLine("Please enter name: ");

                input = Console.ReadLine();

                names.AddLast(input);

            }

            Console.WriteLine("The following are attending: ");

            for(int i = 0; i < names.Count(); i++)

            {

                Console.WriteLine(names.ElementAt(i));

            }

LinkedList called names initialization along with the 3 console messages amount to a frequency count of 3.

Inside the while loop, assuming the user enters “n” names until the user mentions “!”, the Console.ReadLine(); and the AddLast function for the names LinkedList for each iteration of the loop has a frequency count of 1 as well. Meanwhile, the while loop itself runs n times once for each name inputted, the frequency count is 4(n+1)= 4n+4.

For Loop Iterations: The for loop itself iterates n times (once for each element in the list).

Operations within the For Loop: Inside the loop, there are two distinct operations:

The loop condition check (i < names.Count()) is evaluated n+1 times (n times true and once false).

The Console.WriteLine(names.ElementAt(i)) operation is executed n times.

However, the complexity of names.ElementAt(i) in a LinkedList is O(i) for each call, because it starts from the beginning of the list and traverses to the i-th element. The total time complexity for accessing all elements via ElementAt in a LinkedList is the sum of the first n natural numbers, which is n(n+1)/2

So n+1 for the loop checks and n(n+1) for total cost of accessing all elements in ElementAt.

Hence, the total counts of the frequency count is 3+1+4n+4+n+1+n^2+n= n^2+6n+9

Discarding constant terms = n^2+6n.

Clearing coefficients = n^2+n

Picking the most significant term = n^2

Big O notation= O(n^2)