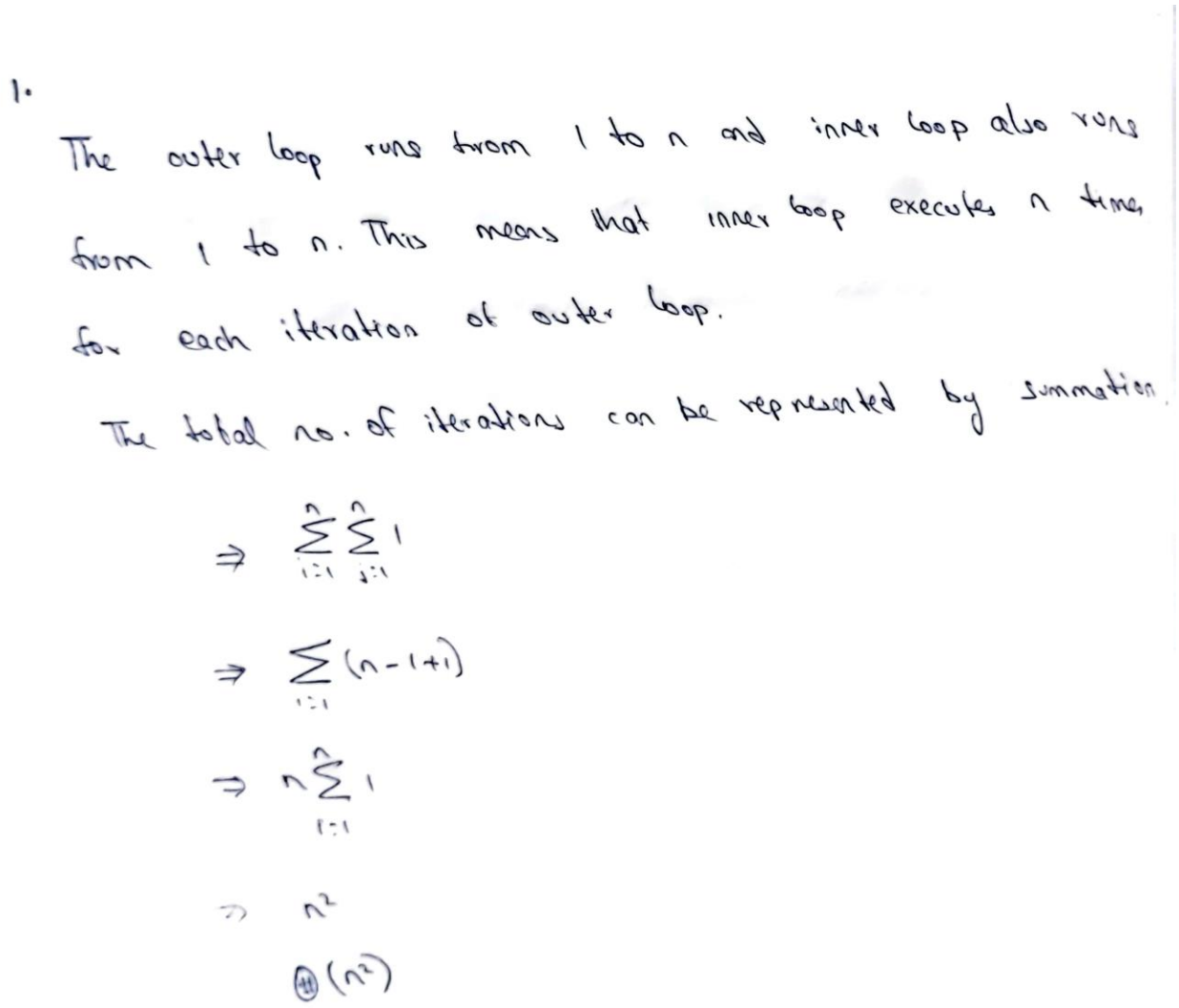
DAA HandsOn_3

1. Find the runtime of the algorithm mathematically (I should see summations).



1.

The outer loop runs from 1 to n and inner loop also runs from 1 to n. This means that inner loop executes n times for each iteration of outer loop.

The total no. of iterations can be represented by summation.

$$\Rightarrow \sum_{i=1}^{n} \sum_{j=1}^{n} 1$$

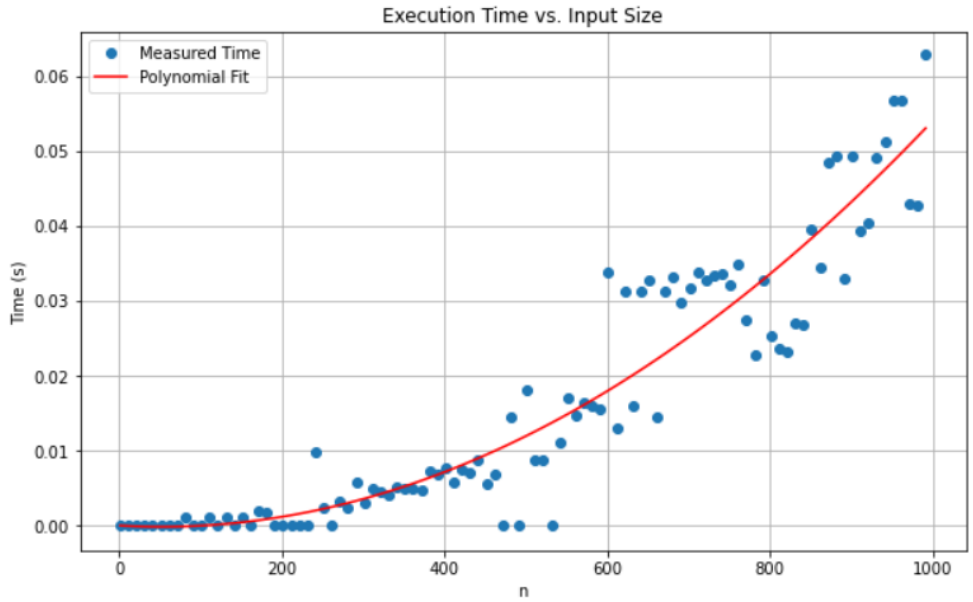
$$\Rightarrow \sum_{i=1}^{n} (n-1+1)$$

$$\Rightarrow n \sum_{i=1}^{n} 1$$

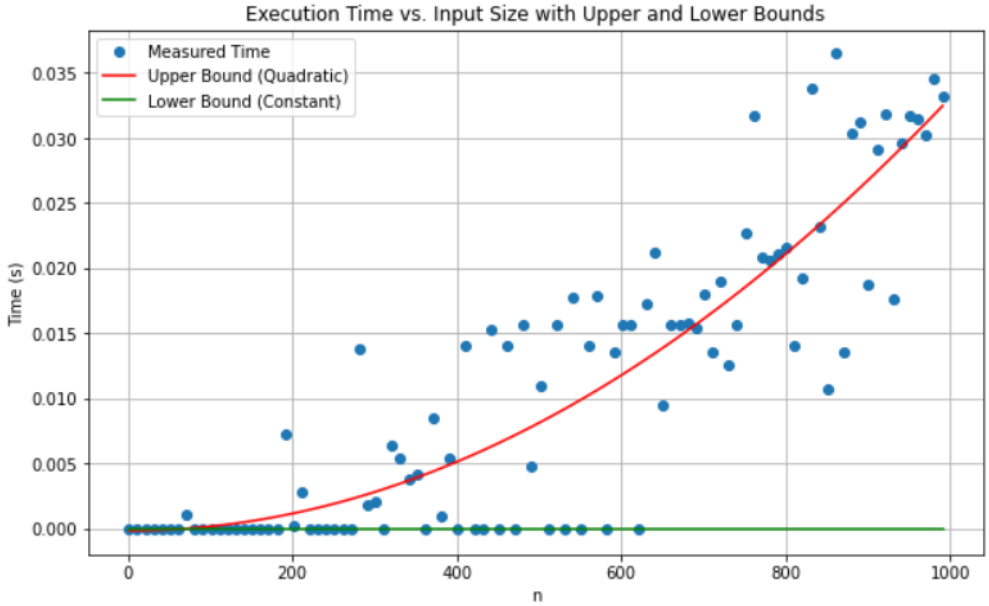
$$\Rightarrow n^2$$

$$\therefore O(n^2)$$

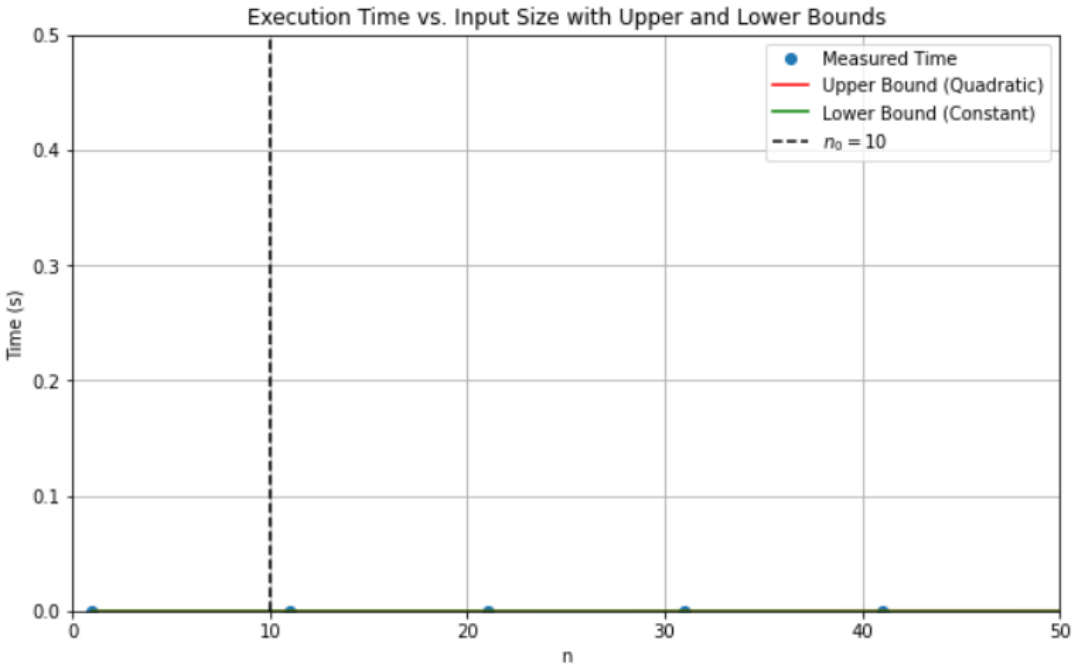
2. Tme this function for various n e.g. n = 1,2,3.... You should have small values of n all the way up to large values. Plot "time" vs "n" (time on y-axis and n on x-axis). Also, fit a curve to your data, hint it's a polynomial.



3. Find polynomials that are upper and lower bounds on your curve from #2. From this specify a big-O, a big-Omega, and what big-theta is.



4. Find the approximate (eye ball it) location of "n_0" . Do this by zooming in on your plot and indicating on the plot where n_0 is and why you picked this value. Hint: I should see data that does not follow the trend of the polynomial you determined in #2



The approximate input size at which the algorithm's behavior changes is represented by n_0 in the plot of execution time vs. input size. Usually, this behavior change is seen as a divergence from the fitted polynomial curve's trend. The plot's observation was used to determine the value of n_0=10. The reason this value was chosen was that it seemed to be the approximate input size at which the measured execution time started to significantly deviate from the fitted polynomial curve trend.

If I modified the function to be:

```
x = f(n)
  x = 1;
  y = 1;
  for i = 1:n
     for j = 1:n
```

```
x = x + 1;
y = i + j;
```

5. Will this increase how long it takes the algorithm to run (e.x. you are timing the function like in #2)?

Yes, the modification in the above code will increase the time it takes the algorithm to run. Because the additional operation y = i + j inside the nested loop will add extra computational for each iteration of the inner loop. Hence, the overall execution time of the algorithm will likely increase.

6. Will it effect your results from #1?

5. It will not affects the results. As the runtime complexity is same as previous. It will slightly increase the results compilation time because of the iteration.

$$T(n) \Rightarrow c + \sum_{i=1}^{n} \sum_{j=1}^{n} 1$$

$$\Rightarrow c + \sum_{i=1}^{n}$$

from previous (1)

$$\Rightarrow c + n^2$$

$$\Rightarrow \Theta n^2$$

The runtime for the modified functions is still same.

Merge sort working for given array [5,2,4,7,1,3,2,6]