

DAA HOMEWORK 3

1. function $x = f(n)$

$x = 1;$

for $i = 1:n$

for $j = 1:n$

$x = x + 1;$

Step-by-step analysis:

1. **Initialization:**

$x = 1;$ This operation takes constant time $O(1)$.

2. **Outer loop:**

for $i = 1:n$ loop runs n times.

3. **Inner loop:**

for $j = 1:n$ loop also runs n times for each iteration of the outer loop.

4. **Inside the inner loop:**

The operation $x = x + 1;$ is executed once when the inner loop runs, which takes $O(1)$ time.

Total number of iterations:

The total number of iterations for the $x = x + 1;$ for 2 loops. Since both loops run n times,

$$\text{Total iterations} = \sum_{i=0}^n \sum_{j=1}^n 1 = \sum_{i=1}^n n = n * n = n^2$$

Runtime:

- The initialization step takes constant time: $O(1)$.
- The nested loops iterate n^2 times.

Thus, the total runtime $T(n)$ of the function is:

$$T(n) = O(1) + O(n^2) = O(n^2)$$

Conclusion:

The runtime of the algorithm is $O(n^2)$.

3.

Since the runtime of the algorithm is $T(n) = (n^2)$ which is quadratic, from the above analysis

Big-O:

The function is $O(n^2)$ since the curve is a quadratic, which represents the upper bound.

Big-Omega:

The function is $\Omega(n^2)$ since data is at quadratic growth rate, representing the lower bound.

Big-Theta:

Since both Big-O and Big-Omega are n^2 , the Big-Theta is also $\Theta(n^2)$

4. $n = 5$, we can set $n_0 = 5$.

n_0 is chosen because, after this point, the data follows the expected polynomial curve. This is where the measured time starts to follow the polynomial equation of n^2

5. 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;
```

Yes, adding $y = i + j$; increases the number of iterations within the innermost loop, thereby increasing the actual time taken per iteration. However, this additional operation is still $O(1)$, so the time complexity remains $O(n^2)$

6.

- **Time Complexity:** The asymptotic runtime remains $O(n^2)$ since adding a constant-time operation does not change the dominant term in the complexity analysis.
- **Measured Execution Time:** The actual execution time will increase slightly due to the extra operation, but the corresponding bounds (Big-O, Big-Omega, Big-Theta) remain the same.

The analysis remains accurate, and only the constant factor changes slightly.