Hands on -3

Nagandla Sai Sreeja
1002269499

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

1) find the runtime of the algorithm mathematically.

Initialize x=1 (constant time , O(1))

Executes a loop:

*) The outer loop runs from i=1 to n, and it iterates n times for each

*) the inner Loop runs from j=1 to n, and it iterates n times for each outer loop iteration

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

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

$$\Rightarrow n^2$$

The total Runtime complexity $T(n) = O(n^2)$

3) the upper and lower bounds on curve, from this specify. big-O, big-Omega, big-theta is

Big-O (upper Bound) ⇒
the f(n) doesnot grow faster than $O(n^2)$.

Big-Omega → Lower bound:-

The f(n) does not grow Slower than $\Omega(n^2)$.

Big-Theta → tighlt Bound:-

$\Theta(n^2)$ this function grows exactly at the rate of n^2.

4) If I modified the function to be

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

will this increate how long it takes the algorithm to run?

By adding an extra operation y = i+j; means each iteration takes slightly longer in real execution time

∴ $O(n^2)$ is the time Complexity.

5) It increase execution time but does not effect and Change the overall $O(n^2)$ time Complexity.