Minh Duc Nguyen
CSC 143 Winter 2021

BIG O / WORST CASE EQUATION

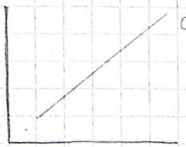
@ Algorithm Efficiency.

1) Consider the algorithm for finding the maximum number in an array of n elements. What is the order of n, O(?) for this algorithm? Explain.

Ans: O(n)

In this scenario, we use the loop, run through it and check if the element in the list is maximum whether or not. So it takes n times to do it.
=) Time complexity = O(n).

Graphical representation



2) What is the order of n, O(?) for the following nested loop?

```
for (i = 1; i <= n; i++) {
    for (j = 1; j <= i; j++) {
        k = k + i + j;
    }
}
```

(k = k + i + j: constant time)

Inner loop executes i times

Outer loop executes n times

=) Runtime: $\frac{n(n+1)}{2}$

$\rightarrow O(n^2)$

3) What is the order of n, O(?) for the following nested loop?

```
for (i = 1; i <= n; i++) {
    for (j = 1; j <= 20; j++) {
        k = k + i + j;
    }
}
```

Outer loop executes n times

j <= 20, so it runs 20 times => O(1)

2 loops -> O(1)

=) Runtime complexity = O(n).

4) What is the order of n, O(?) for the following sequence ?

```
for (j = 1; j <= 10; j++) {
    k = k + 4;
}
for (i = 1; i <= n; i++) {
    for (j = 1; j <= 20; j++) {
        k = k + i + j;
    }
}
```

The loop executes 10 times $\Rightarrow O(1)$

As (3), Runtime complexity is $O(n)$

$\Rightarrow$ Runtime complexity is $O(n)$

5) What is the order, O(?) of the following function ?

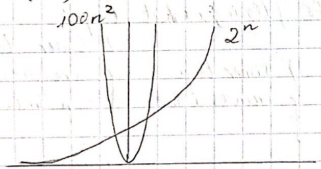
$n^3 + 100n^2 + n$

$\Rightarrow$ $O(n^3)$ because n^3 is the highest order.

6) What is the order, O(?) of the following function ?

$2^n + 100n^2 + 45n$

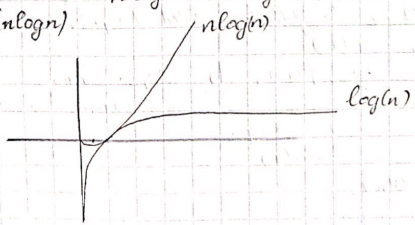
$\Rightarrow$ $O(2^n)$ because 2^n is higher order than $100n^2$



7) Suppose an algorithm takes exactly the given number of statements for each value below, in terms of the size of n, i.e. the order of n, O(?). Explain

$n \log n + \log n + n$

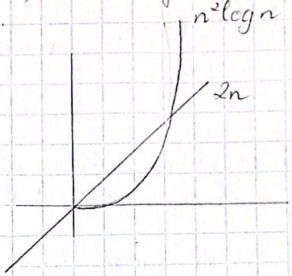
$\Rightarrow$ $O(n \log n)$



8) Suppose an algorithm takes exactly the given number of statements for each value below, in terms of the size of n, i.e. the order of n, O(?). Explain.

$n^2 \log n + 2n$

→ $O(n^2 \log n)$



⊛ WORST-CASE EQUATION

9, Algorithm - 01.

```
int sum = 0;          } runs 2 operations.
int J = 1;
while (J <= n) {      } => The loop executes untill J ≤ n
    sum ++;
    J = J * 2;        } -> J is doubled (×2)
}
```

→ The loop breaks till $J = 2^k n$ $\quad (2^k = n,\ k = \log(n))$

=) Total worst case runtime equation

$= 2 + \log(n) + \log(n) + \log(n) = 3\log(n) + 2$

→ $O(\log n)$

10, Algorithm - 02.

```
int sum = 0;                                  ∈ 1 time
for (int i = 0; i <= n; i++) {                ∈ n times
    for (int J = 0; J <= n; J++) {            ∈ n times      } -> n² times
        sum ++;
    }
}
for (int i = 1; i <= n; i++) {                ∈ n times
    for (int J = 1; J <= 20; J++) {           ∈ 20 times     } -> 20n times
        sum --;
    }
}
```

→ Worst case runtime : $n^2 + 20n + 1$

→ $O(n^2)$.