

C++ Assignments | Time and space complexity Analysis - 1 | Week 8

1. Calculate the time complexity for the following code snippet.

```
int c = 0;
for(int i = n; i > 0; i /= 2) {
    c++;
}
```

Time Complexity Analysis:

The loop starts with $i = n$ and halves i each iteration ($i /= 2$). This means the number of iterations is the number of times you can divide n by 2 until i becomes 0.

- The number of times n can be divided by 2 is approximately $\log_2(n)$.
- Therefore, the time complexity is **$O(\log)$**

2. Calculate the time complexity for the following code snippet.

```
int c = 0;
for(int i = n; i > 1; i /= i) {
    c++;
}
```

Time Complexity Analysis:

In each iteration, i is divided by i , which results in i becoming 1 immediately after the first iteration, if i is greater than 1.

- The loop only runs once.
- Therefore, the time complexity is **$O(1)$** .

3. Calculate the time complexity for the following code snippet where k is some constant ($k \ll n$).

```
int c = 0;
for(int i = 0; i < n; i += k) {
    c++;
}
```

Time Complexity Analysis:

The loop increments i by a constant k each iteration. Since k is much smaller than n ($k \ll n$), we consider it as a constant.

- The number of iterations is approximately n / k .
- Since k is a constant, the time complexity is **$O(n)$** .

4. Calculate the time complexity for the following code snippet.

```
int c = 0;
for(int i = 1; i < n; i *= 2) {
    c++;
}
```

Time Complexity Analysis:

The loop starts with $i = 1$ and multiplies i by 2 each iteration ($i *= 2$). This means the number of iterations is the number of times you can multiply 1 by 2 until i reaches or exceeds n .

- The number of times you can double i is approximately $\log_2(n)$.
- Therefore, the time complexity is $O(\log n)$.

5. Calculate the time complexity for the following code snippet.

```
int c = 0;
for(int i = 0; i < n; i++) {
    c += i;
}
```

Time Complexity Analysis:

The loop runs from $i = 0$ to $i < n$, making n iterations. Within each iteration, there is a constant time operation $c += i$.

- Therefore, the time complexity is $O(n)$.

6. Calculate the time complexity for the following code snippet.

```
int c = 0;
for(int i = 0; i < n; i++) {
    for(int j = 0; j < i; j++){
        c++;
    }
}
```

Time Complexity Analysis:

The outer loop runs n times (i ranges from 0 to $n-1$). The inner loop runs i times for each iteration of the outer loop.

- When $i = 0$, inner loop runs 0 times.
- When $i = 1$, inner loop runs 1 time.
- When $i = 2$, inner loop runs 2 times.
- ...
- When $i = n-1$, inner loop runs $n-1$ times.

The total number of iterations of the inner loop is the sum of the first $n-1$ integers:

0
+
1
+
2
+
...
+
(
n

—
1
)

$0 + 1 + 2 + \dots + (n-1)$

$0+1+2+\dots+(n-1)$.

- This sum is (

n

—

1

)

.

n

2

$\frac{(n-1) \cdot n}{2}$

$\frac{2(n-1) \cdot n}{2}$, which is O

(

n

2

)

$O(n^2)$

$O(n^2)$.

- Therefore, the time complexity is **$O(n^2)$** .