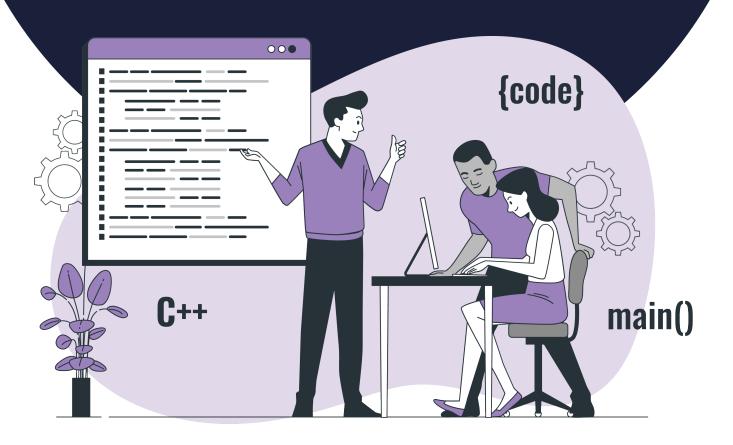
Lesson:



Problems on Time Complexity





Pre-Requisites

• Time and Space Complexity

List of Concepts Involved

• Problems on Time Complexity

Topic 1: Time Complexity Problems

Problem 1:

Calculate the time complexity for the following code snippet.

```
int val = 0;
for(int i = 1; i <= N; i += i){
    val++;
}</pre>
```

Explanation:

We will calculate the total number of iterations in the above loop to calculate the time complexity. Total number of iterations will be:

The values of i in the loop:

```
1, 2, 4, 8, 16, ... 2^k <= N
k ~ logN
```

Hence the time complexity will be the total number of iterations that will be k giving us a time complexity of **O(logN)**.

Problem 2:

Calculate the time complexity for the following code snippet.

```
int val = 0;
for(int i = 1; i <= N; i *= 2){
     for(int j = 1; j <= i; j++) {
         val++;
}</pre>
```

Explanation:

Here in this case we calculate the number of iterations in the given nested loops, to calculate this we just calculate the number of times j has iterated.

Let there be k times the ith loop is iterated, then we know that

```
2^k <= N ...equation (i)
```

```
1+2+4+8+...+2^k-1 times then
```

The total sum for the above Geometric progression will be $2^{(k)} - 1$equation (ii) Here from equation (i), we get that k = logN,

So the total time complexity from equation (ii) is sum of iterations will be $2^{(k)}$ where k = logN giving a worst case time complexity of $O(2^{(logN)}) \sim O(N)$.

Problem 3:

Calculate the time complexity for the following code snippet.

```
int val = 0;
for(int i = 1; i <= N; i *= 2){
     for(int j = N; j > i; j--) {
         val++;
}
```

Explanation:

Here we calculate the number of iterations j will take, let's say the number of iterations in ith loop be k, then here $2^k = N$, the value of k will come out to be k = logN.

The number of iterations in jth loop will be

```
(N-1) + (N-2) + (N-4) + ... + (N-2^(k-1))

= k*N-(1+2+4+8+...+2^(k-1))

= k*N-(2^(k) - 1)

Putting k = logN, we get

=> (NlogN - N) iterations

Final Time Complexity: O(N logN)
```

Problem 4:

Calculate the time complexity for the following code snippet.

```
int val = 0;
for(int i = 2; i <= N; i *= i){
    val++;</pre>
```

Explanation:

}

To calculate the time complexity for the following code snippet, we will calculate the total number of iterations. Let us first analyze the values of i in the above loop,

```
2, 4, 16, 256, ...
2, 2^2, 2^4, 2^8, 2^16,...
Let the total above terms in i be k,
Then values of i become
2, 2^2, 2^4, 2^8, 2^16, ..., 2^k
```



Here 2^k < N, k ~ logN,

Also we note that k is also getting incremented in powers of 2, let the total number of iterations be T then 2, 4, 8, 16, ... 2^T <= k

Hence **T = logk**

Overall time complexity will be the total number of iterations ie, T = logk, As we know k = logN, T = log(logN)Hence the overall time complexity becomes O(log(logN)).

Upcoming Class Teasers

Bubble sort

