

# Time Complexity

Mohammed Rizin  
Unemployed

March 20, 2025

## 1 Time Complexity

### 1.1 Simple For Loop

---

**Algorithm 1** Simple for loop

---

```
for  $i \leftarrow 0$  to  $n - 1$  do  
    ....STMT....  
end for
```

---

Time complexity:  $O(n)$ .

### 1.2 Simple Reverse Loop

---

**Algorithm 2** Simple reverse loop

---

```
for  $i \leftarrow n$  down to 1 do  
    ....STMT....  
end for
```

---

Time complexity:  $O(n)$ .

### 1.3 For Loop with Step

---

**Algorithm 3** Simple for loop with step

---

```
for  $i \leftarrow 0$  to  $n - 1$  step 2 do  
    ....STMT....  
end for
```

---

Time complexity:  $O(n)$  (as  $n/2$  is asymptotically  $O(n)$ ).

### 1.4 Nested Loops

---

**Algorithm 4** Nested loops

---

```
for  $i \leftarrow 0$  to  $n - 1$  do  
    for  $j \leftarrow 0$  to  $i - 1$  do  
        ....STMT....  
    end for  
end for
```

---

Time complexity:  $O(n^2)$ .

$i$	$j$	STMT	Total STMT	Total Time	Time Complexity
0	0	1	1	1	1
1	0	1	2	3	3
1	1	1	3	6	6
2	0	1	4	10	10
2	1	1	5	15	15
2	2	1	6	21	21

The total number of executions is  $n(n+1)/2$ , so the time complexity is  $O(n^2)$ .

## 1.5 Summation Loop

---

### Algorithm 5 Summation loop

---

```

 $p \leftarrow 0$ 
for  $i \leftarrow 1$  while  $p \leq n$  do
     $p \leftarrow p + i$ 
end for

```

---

$i$	$p$	STMT	Total STMT	Total Time	Time Complexity
1	1	1	1	1	1
2	3	1	2	3	3
3	6	1	3	6	6
4	10	1	4	10	10
5	15	1	5	15	15
6	21	1	6	21	21
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$k$	$\frac{k(k+1)}{2}$	1	$k$	$\frac{k(k+1)}{2}$	$\frac{k(k+1)}{2}$

Assuming  $p \leq n$ :

$$\begin{aligned}
 p &= \frac{k(k+1)}{2}, \\
 \frac{k(k+1)}{2} &> n, \\
 k^2 &> n, \\
 k &> \sqrt{n}.
 \end{aligned}$$

Time complexity:  $O(\sqrt{n})$ .

## 1.6 Multiplication Loop

---

### Algorithm 6 Multiplication loop

---

```

for  $i \leftarrow 1$  while  $i \leq n$  step  $2 \cdot i$  do
    ...Statement...
end for

```

---

step	$i$
1	2
2	4
3	8
4	16
5	32
$\vdots$	$\vdots$
k	$2^k$

This stops when  $i > n$  :

$$\begin{aligned} \text{i.e. } 2^k &> n, \\ \log_2 2^k &> \log_2 n \\ k \cdot \log_2 2 &> \log_2 n \\ k &> \log_2 n \end{aligned}$$

Time complexity:  $O(\log_2 n)$ .

However, If you notice log function, which gives both float and integer values, then should we take floor or ceil of the floating point value.

n = 8
1
2
4
$8 < 8$ fails !!
So in total we could execute it for 3 iterations. $\log_2 8 = 3$

n = 10
1
2
4
8
$16 < 10$ fails !!
So in total we could execute it for 4 iterations. $\log_2 10 = 3.32$ $\text{ceil}(\log_2 10) = 4$

## 1.7 Division Loop

---

**Algorithm 7** For Loop with Division

---

```

for  $i \leftarrow n$  while  $i \geq 1$  step  $\frac{i}{2}$  do
    Statement
end for
```

---

step	$i$
1	$n$
2	$\frac{n}{2}$
3	$\frac{n}{2^2}$
4	$\frac{n}{2^3}$
5	$\frac{n}{2^4}$
$\vdots$	$\vdots$
k	$\frac{n}{2^{k-1}}$
k+1	$\frac{n}{2^k}$

The loop stops when:

$$\begin{aligned}\frac{n}{2^{k-1}} &< 1, \\ 2^{k-1} &> n, \\ k-1 &> \log_2 n, \\ k &> \log_2 n + 1.\end{aligned}$$

Thus, the time complexity is  $O(\log_2 n)$ .

## 1.8 Square Loop

---

**Algorithm 8** Loop till square of i is less than n

---

```

for  $i \leftarrow 1$  while  $i^2 < n$  do
    Statement
end for

```

---

i	$i^2$
0	0
1	1
2	4
3	9
$\vdots$	$\vdots$
k	$k^2$

The loop stops when:

$$\begin{aligned}k^2 &\geq n \\ k &\geq \sqrt{n}\end{aligned}$$

Thus, the time complexity is  $O(\sqrt{n})$ .

## 1.9 Independent loop

---

**Algorithm 9** Independent for loops

---

```

for  $i \leftarrow 0$  to  $n-1$  do
    ....STMT....
end for
for  $j \leftarrow 0$  to  $n-1$  do
    ....STMT....
end for

```

---

Since this is not nested. They are independent loops. They have  $O(n)$  for both the loops adding to  $T(n) = 2n$ .

But **Time complexity:**  $O(n)$ .

## 1.10 One loop dependent on another

---

### Algorithm 10 Independent for loops

---

```

 $p \leftarrow 0$ 
for  $i \leftarrow 0$  while  $i < n$  step  $2 \cdot i$  do
     $p++$ 
end for
for  $j \leftarrow 0$  while  $j < p$  step  $2 \cdot j$  do
    ...STMT...
end for

```

$\triangleright p = \log_2(n)$   
 $\triangleright \log_2 p$

---

Thus this algorithm takes  $O(\log \log n)$

## 2 Analysis of If & While loops

---

### Algorithm 11 If and While loops

---

```

 $i \leftarrow 0$ 
while  $i < n$  do
    Statement
     $i++$ 
end while

```

$\triangleright 1$   
 $\triangleright n + 1$   
 $\triangleright n$   
 $\triangleright n$

---

$$T(n) = 2n + 2$$

**Time complexity:**  $O(n)$

### 2.1 While with two variables

---

#### Algorithm 12 While with two variables

---

```

 $i \leftarrow 1$ 
 $j \leftarrow 1$ 
while  $j < n$  do
    Statement
     $j \leftarrow j + i$ 
     $i \leftarrow i + 1$ 
end while

```

---

Let us trace the values of  $i$  and  $j$ :

$i$	$j$
1	1
2	2
3	$2 + 2$
4	$2 + 2 + 3$
5	$2 + 2 + 3 + 4$
6	$2 + 2 + 3 + 4 + 5$
$\vdots$	$\vdots$
$k$	$2 + 2 + 3 + 4 + \dots + k - 1$
$k+1$	$2 + 2 + 3 + 4 + \dots + k \approx \frac{k \cdot (k+1)}{2}$

The loop stops when:

$$\begin{aligned}\frac{k(k+1)}{2} &\geq n, \\ k^2 &\gtrapprox n, \\ k &\gtrapprox \sqrt{n}.\end{aligned}$$

Thus, the time complexity is  $O(\sqrt{n})$ .

The equivalent for loop is:

---

**Algorithm 13** For loop with two variables

---

```
for  $i \leftarrow 1$  while  $j < n$  do
  Statement
   $j \leftarrow j + i$ 
end for
```

---

## 2.2 Example : GCD of two numbers

---

**Algorithm 14** GCD of two numbers

---

```
while  $a \neq b$  do
  if  $a > b$  then
     $a \leftarrow a - b$ 
  else
     $b \leftarrow b - a$ 
  end if
end while
```

---

Lets trace the values of  $a = 10$  and  $b = 15$ :

$a$	$b$	$a \neq b$
10	15	True
5	15	True
5	10	True
5	5	False

So it took 3 iterations to find the GCD of 10 and 15.

**Time complexity:**  $O(\max(a, b))$  min Time complexity:  $O(1)$  and max Time complexity:  $O(n)$

### 3 Conclusion

---

<b>for</b> $i \leftarrow 0$ <b>to</b> $n - 1$ <b>do</b> Statement <b>end for</b>	$\triangleright \blacktriangleright O(n)$
<b>for</b> $i \leftarrow 0$ <b>to</b> $n - 1$ <b>step</b> $i + 2$ <b>do</b> Statement <b>end for</b>	$\triangleright \frac{n}{2} \blacktriangleright O(n)$
<b>for</b> $i \leftarrow n$ <b>to</b> $1$ <b>do</b> Statement <b>end for</b>	$\triangleright \blacktriangleright O(n)$
<b>for</b> $i \leftarrow 0$ <b>to</b> $n - 1$ <b>step</b> $i \cdot 2$ <b>do</b> Statement <b>end for</b>	$\triangleright \blacktriangleright O(\log_2 n)$
<b>for</b> $i \leftarrow 0$ <b>to</b> $n - 1$ <b>step</b> $i \cdot 3$ <b>do</b> Statement <b>end for</b>	$\triangleright \blacktriangleright O(\log_3 n)$
<b>for</b> $i \leftarrow n$ <b>to</b> $1$ <b>step</b> $\frac{i}{2}$ <b>do</b> Statement <b>end for</b>	$\triangleright \blacktriangleright O(\log_2 n)$

---