```
void f4(int n) {
    int count = 0;
    for(int i = 1; i <= n; i++) {
        for(int j = 1; j <= n; j++) {
            count += 1;
        }
    }
}
```

nested

$\cancel{c} + 3nc$

$i = 1 \rightarrow$ loop $j \rightarrow [1, N] \rightarrow \underline{3nc}$

$i = 2 \rightarrow$ loop $j \rightarrow [1, N] \rightarrow 3nc$

$i = 3 \rightarrow$ loop $j \rightarrow [1, N] \rightarrow 3nc$

$\vdots$

$i = n \rightarrow$ loop $j \rightarrow [1, N] \rightarrow \underline{3nc}$

total ins – $\rightarrow 3nc + 3nc + 3nc \ldots\ldots 3nc$ $\underline{(n \text{ times})}$

$\rightarrow 3n^2c \longrightarrow \underline{O(n^2)}$

for every iteration of i, multiple instructions of loop of j will be executed.

If we sum up no. of instructions executed by j, for every value of i then we can get total instructions

$N=3$

```
void f5(int n) {
    int count = 0;
    for(int i = 1; i ≤ n; i++) {
        for(int j = i; j ≤ n; j++) {
            count += 1;
        }
    }
}
```

nested

$O(N^2)$

$i=1$ loop $j \rightarrow [1, N] \rightarrow N$

$i=2$ loop $j \rightarrow [2, N] \rightarrow N-1$

$i=3$ loop $j \rightarrow [3, N] \rightarrow N-2$

$\vdots$

$i=N$ loop $j \rightarrow [N, N] \rightarrow 1$

Total ins $\rightarrow N + (N-1) + (N-2) + (N-3) \ldots\ldots\ 3+2+1$

$$\rightarrow \frac{N(N+1)}{2} \Rightarrow \underbrace{\frac{N^2}{2} + \frac{N}{2}}_{\downarrow}$$

$$O(N^2)$$

$N=3$

$1+2+3$

$\frac{3(\cancel{4})}{\cancel{2}} \rightarrow 6$

max
3 places

[3, 1, 2, 4, 9, 8]

```
longestConsecutiveSequence(arr) {
    let mp = {};
    for(let i = 0; i < arr.length; i++) {
        mp[arr[i]] = true;
    }
    let ans = -1;
    for(let i = 0; i < arr.length; i++) {
        if(mp[arr[i] - 1]) {
            continue;
        } else {
            let len = 0;
            let x = arr[i];
            while(mp[x]) {
                len++;
                x++;
            }
            ans = Math.max(ans, len);
        }
    }
    return ans;
}
```

→ 3N

~ O(N)

i = 3

i = 0

i = 2

key value

{
3 - true
1 - true
2 - true
4 - true
9 - true
8 - true
}

i = ~~0 1 2 3 4~~ 5

x = ~~1 2 3 4~~ 5

→ O(N)

4 iterations [x = 1 → 2 → 3 → 4]

x = ~~8 9~~ 10

2 iterations [x → 8, 9]

```c
void f6(int n) {
    int count = 0;
    while(n > 0) {
        count++;
        n /= 2;
    }
}
```

$O(\log n)$

total instructions

In every iteration of while loop, we exec

3 ins

Assume total → K iterations

1st iteration	2nd	3rd	4th	......	Kth
$n/1$	$n/2$	$n/4$	$n/8$		$n/2^{k-1} \rightarrow 1$
$n/2^0$	$n/2^1$	$n/2^2$	$n/2^3$		

$$\frac{n}{2^{k+1}} = 1$$

$$n = 2^{k-1}$$ taking $\log_2$ both sides

$$\log_2 n = \log_2 2^{(k-1)}$$

$$\log_2 n = (k-1)\log_2 2$$

$$\log_2 n = k-1$$

$$\boxed{K = \log_2 n + 1}$$

$$\log_3 n \rightarrow \frac{\log_2 n}{\log_2 3}$$

Total ins $\rightarrow 3(\log_2 n + 1)$

$$\downarrow$$

$$O(\log n)$$

```
void f7(int n) {
    int count = 0;
    for(int i = 1; i ⩽ n; i *= 2) {
        count++;
    }
}
```

$O(\log n)$

1st → 3, 2nd → 3, 3rd → 3, ... Kth → 3

lets assume we have K iterations

Total ops → 3K

	1st	2nd	3rd	4th	5th	....	Kth
$i =$	1	2	4	8	16		2^{k-1}
	(2^0)	(2^1)	(2^2)	(2^3)	(2^4)	....	(2^{k-1})

(2^{k-1}) → last value of i viz $\leq n$

$2^{k-1} \leq n$ taking log both sides

$$\log_2 2^{k-1} \leq \log_2 n \longrightarrow (k-1)\underbrace{\log_2 2}_{1} \leq \log_2 n$$

$$k-1 \leq \log_2 n \longrightarrow (k \leq \log_2 n + 1) \longrightarrow O(\log n)$$

```
void f8(int n) {
    int count = 0;
    for(int i = 1; i <= n; i++) {
        for(int j = 1; j <= sqrt(n); j++) {
            count++;
        }
    }
}
```

$i=1 \rightarrow j \Rightarrow [1, \sqrt{n}] \rightarrow \sqrt{n}$

$i=2 \rightarrow j \rightarrow [1, \sqrt{n}] \rightarrow \sqrt{n}$

$i=3 \rightarrow j \rightarrow [1, \sqrt{n}] \rightarrow \sqrt{n}$

$\vdots$

$i=N \rightarrow j \rightarrow [1, \sqrt{N}] \rightarrow \sqrt{N}$

$O(N\sqrt{N})$

```
void f9(int n) {
    int ans = 0;
    for(int i = 0; i < n; i++) {
        ans += i;
    }
    for(int i = 0; i < n; i++) {
        for(int j = 0; j < n; j++) {
            ans += (i+j);
        }
    }
}
```

first

2nd

$O(n + n^2) \rightarrow \underline{O(n^2)}$

$i = 0 \quad j \rightarrow [0, N-1] \rightarrow N$

$i = 1 \quad j \rightarrow [0, N-1] \rightarrow N$

$\vdots$

$i = N-1 \quad j \rightarrow (0, N-1) \rightarrow N$

$\underline{N \times N}$

```c
void f10(int n) {
    int count = 0;
    for(int i = n; i > 0; i /= 3) {
        count++;
    }
}
```

1^{st} $\quad 2^{nd}$ $\quad 3^{rd}$ $\quad \ldots\ldots$ $\quad k$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$n \quad n/3 \quad n/9 \quad n/3^{k-1}$

$$\frac{n}{3^{k-1}} \approx 1$$

$$n \approx 3^{k-1}$$

$$\log_3 n \approx \log_3 3^{k-1}$$

$$\log_3 n \approx k-1$$

$$k \rightarrow O(\log n)$$

```
void f11(int n) {
    int i = 1, s = 1;
    while(s ≤ n) {
        i++;
        s += i;
    }
}
```

$O(\sqrt{n})$

$i = 1 \qquad s = 1$

$i = 2 \qquad s = (1+2)$

$i = 3 \qquad s = (1+2+3)$

$i = 4 \qquad s = (1+2+3+4)$

⋮

if loop terminates here

$i = k \qquad s = (1+2+3+4 \ldots\ldots k)$

$s \leq n \rightarrow (1+2+3+4 \ldots\ldots k) \leq n$

$\frac{k(k+1)}{2} \leq n \rightarrow \approx k^2 \leq n$

$\approx k \leq \sqrt{n}$