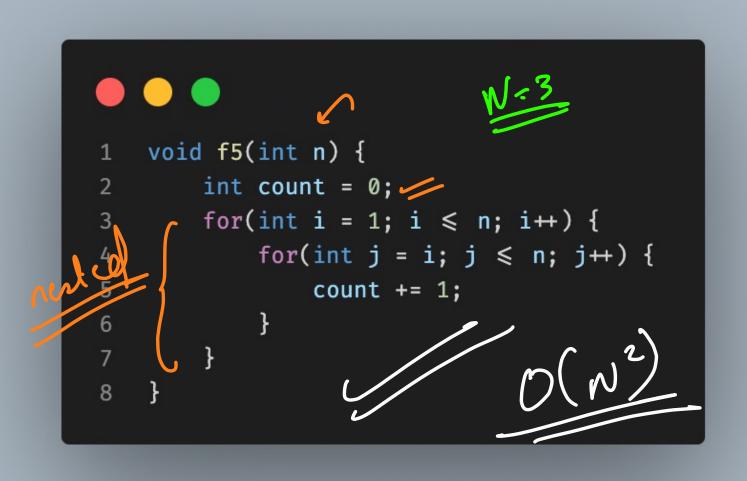
```
1  void f4(int n) {
2    int count = 0;
3    for(int i = 1; i ≤ n; i++) {
        for(int j = 1; j ≤ n; j++) {
            count += 1;
        }
7     }
8  }
```

```
i=1 \longrightarrow loop J \rightarrow [I,N] \rightarrow \frac{3nc}{3nc}
i=2 \longrightarrow loop J \rightarrow [I,N] \rightarrow \frac{3nc}{3nc}
i=n \longrightarrow loop J \rightarrow [I,N] \rightarrow \frac{3nc}{3nc}
```

for every iteration of i, multiple instructions of loop of j well be

Ineculed.

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



$$i = 1$$

$$i = 2$$

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

$$i = 3$$

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

$$\vdots$$

$$i = N$$

$$loop j \rightarrow [N, N]$$

$$\frac{1}{100} \rightarrow N + (N-1) + (N-2) + (N-3) \dots 3+2+1$$

$$\frac{N(N+1)}{2} \rightarrow \frac{N^2}{2} + \frac{N}{2}$$

$$\frac{1+2+1}{3}$$

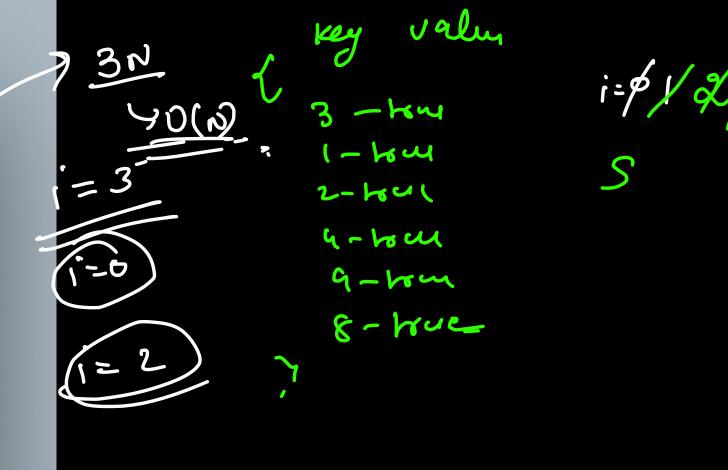
$$\frac{N(N+1)}{3} \rightarrow \frac{N^2}{2} + \frac{N}{2}$$

$$\frac{N(N+1)}{3} \rightarrow \frac{N^2}{2} \rightarrow \frac{N^2}{2} \rightarrow \frac{N}{2}$$

```
3 P/Q W
```

53,1,2,4,9,8

```
longestConsecutiveSequence(arr) {
         let mp = {};
         for(let i = 0; i < arr.length; i++) {</pre>
             mp[arr[i]] = true;
        let ans = -1;
         for(let i = 0; i < arr.length; i++) {</pre>
 8
             if(mp[arr[i] - 1]) {
 9
                 continue;
10
             } else {
11
                 let len = 0;
12
                 let x = arr[i];
13
                 while(mp[x]) {
14
                  ↓ len++;
15
16
17
                 ans = Math.max(ans, len);
18
19
20
21
         return ans;
22
```



x= x 2 3 4 s

4 ituations (x=1-2-7-4)

$$\chi = 8910$$

2 iterations $[n-879]$

```
void f6(int n) {
  int count = 0;
  while(n > 0) {
    count++;
    n ≠ 2;
    6  }
    Clor n
```

Potal instructions

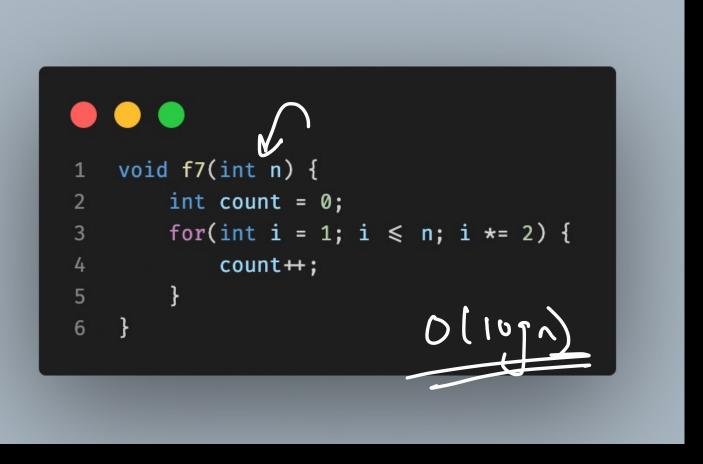
```
In eug iteration of while loop, we exect

3 ins

assume total > K iteration
```

```
1st iteration \frac{1}{2} \frac{1}{2}
```

JOIN THE DARKSIDE



lets assume une home Kiterations

$$i = 1$$
 2 4 8 16 2 (2^k) (2^k) (2^k) (2^k) (2^k) (2^k) (2^k)

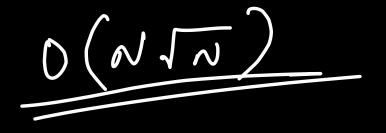
$$2^{k-1} \le n$$

$$(0 \le 2^{k-1} \le \log_2 n)$$

$$(0 \le 2^{k-1} \le$$

```
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 > d=> (1, \( \) \\ i=2 > d > \( \) \\ i=3 > \( \) \\ i=1 \\ i=1
```



```
1  void f9(int n) {
2    int ans = 0;
3    for(int i = 0; i < n; i++) {
        ans += i;
}
6    for(int i = 0; i < n; i++) {
        for(int j = 0; j < n; j++) {
            ans += (i+j);
        }
10        }
11    }
</pre>
```

$$O(n^{2})$$

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

```
1/3 K-1
•
        21
```

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

```
8 = 1
      i = 1
             S=(1+2)
      i= 2
             S= (1+2+3)
     1=3
             S = (1+2+3+4)
           S= (1+2+3+4 .... K)
    i= K
      -> (1+2+3+4....K) 4 n
S < 1
```