Time Complexity

$1^2 + 2^2 + 3^2 + 4^2 + 5^2 - - - + n^2 = \frac{n(n+1)(2n+1)}{6}$

Useful Formulae

$c + c + c + c + c$

$k + k + k + k + k +$ n-times

$\Rightarrow \boxed{nk}$

Formulae

Here is a list of handy formulas which can be helpful when calculating the Time Complexity of an algorithm:

$1+2+3+4+ -$

Summation	Equation
$(\sum_{i=1}^{n} c) = c + c + c + \cdots + c$	cn
$(\sum_{i=1}^{n} i) = 1 + 2 + 3 + \cdots + n$	$\frac{n(n+1)}{2}$
$(\sum_{i=1}^{n} i^2) = 1 + 4 + 9 + \cdots + n^2$ $1^2 + 2^2 + 3^2$	$\frac{n(n+1)(2n+1)}{6}$
$(\sum_{i=0}^{n} r^i) = r^0 + r^1 + r^2 + \cdots + r^n$ $r^0 + r^1 + r^2$	$\frac{(r^{n+1}-1)}{r-1}$

$\frac{n(n+1)}{2}$

$r^0 + r^1 + r^2 + - - - + r^n = \boxed{\frac{r^{n+1} - 1}{r-1}}$

Simple for-loop with an increment of size 1

```
for (int x = 0; x < n; x++) {
    //statement(s) that take constant time
}
```

$5n^3 + n^2 + 5n$

$O(n^3)$

$x=0<5$
$x=1<5$
$x=2<5$
$x=3<5$
$x=4<5$

$\boxed{O(n)}$

$(n+1)$

$n=5$

$n=5$

$x=0<5$ (T)
$x=1<5$ (T)
$x=2<5$ (T)
$x=3<5$ (T)
$x=4<5$ (T)
$x=5<5$ (F)
$x=6<5$

$\boxed{O(n)}$

$(n+1)$ $\boxed{O(n)}$

a) $O(n)$ b) $O(n^2)$ c) $O(\log n)$ d) $O(n^{1/2})$

$O(2)$

$x=0$

$x=0;\ x<n;\ x{+}{+};$

0 1 2 3 4 5

$n+1$

$x+=1$

0, 1, 1+1, 1+1+1

0, k, 2k, 3k, ----

For-loop with increments of size k → natural no.

1, --- ∞

```
for (int x = 0; x < n; x+=k) {
    //statement(s) that take constant time
}
```

① $x -= k$

② $x += k$

$\boxed{O(n)}$

$x+=2$

$x+=2$

$\frac{n}{k}$

$x = 0, k, 2k, 3k, 4k, --$

$k(1, 2, 3, 4, 5 ---)$

$\boxed{nk}$

$\boxed{n\cancel{k}}$

$\boxed{O(n)}$

$\boxed{O(n)}$

$\frac{n \times \cancel{k}}{\cancel{k}}$

$\boxed{\frac{n \times \cancel{k}}{\cancel{k}}}$

$\frac{n \times k}{\cancel{k}}$

$\boxed{\frac{n \times \cancel{k}}{\cancel{k}}}$

0, 2, 4, 6, 8 ---

$2(1, 2, 3, 4)$

$O(n)$

$n = 5$

$m = 6$

Simple nested For-loop

```
for (int i=0; i<n; i++){
    for (int j=0; j<m; j++){
        //Statement(s) that take(s) constant time
    }
}
```

$n = \cancel{m}$

$\cancel{n^2}$

$\cancel{n \times n}$

$i=0 < 5 \rightarrow m$

$i=1 < 5 \rightarrow m$

$i=2 < 5 \rightarrow m$

$i=3 < 5 \rightarrow m$

$i=4 < 5 \rightarrow m$

$\cancel{i=5 < 5} \rightarrow ?\ n$

n times

$m + m + m + m + m \rightarrow n$ times

$O(n * m)$

5×6

$O(n * m)$

i $n=5$

Nested For-loop with dependent variables

```
for (int i=0; i<n; i++){
    for (int j=0; j<i; j++){
        //Statement(s) that take(s) constant time
    }
}
```

$i=0$ $\quad$ $j=0$ $\quad$ $j \to 0$

$i=1$ $\quad$ $j=1 \to 0$

$i=2$ $\quad$ $j=2, \to 0,1$

$i=3$ $\quad$ $j=3, \to 0,1,2$

$i=4$ $\quad$ $j=4, \to 0,1,2,3$

$\vdots$

$i=n-1$ $\quad$ $j=n-1$

$1+2+3+4+5+ \cdots +(n-1)$

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

$$\frac{n^2}{2} - \frac{n}{2}$$

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

$a, \; a+d, \; a+2d$

a for(int i=a; i<n; i*=a) → O(n)

```
i = //constant
n = //constant
k = //constant
while (i < n){
    i*=k;
    // Statement(s) that take(s) constant time
}
```

$\boxed{\log_k n}$

for(int i=1; i<n; i*=2)

k

i=1; i<n; i*=k

1, 2, 4, 8, 16, ... → G.P

$r = k$

$a_n = ar^{n-1}$

$n = 1(2)^{n-1}$ $n-1 \approx x$

$n = 2^x$

$\log$

$\boxed{\log_k n = x}$

$\boxed{T(n) = T(n/2) + k}$

$\log_k n$

$\log k$

$\boxed{O(\log_k n)}$

$n = (k)^{x-1}$ → G.P

$\boxed{\log_k n = x}$

$k = 2$

$a_n = a r^{n-1}$

$n = 1(2)^{n-1}$

$n-1 \approx x$

$n = 2^{n-1}$

$n = 2^x$

$\log$

$\boxed{x = \log_2 n}$

$T(n) = T(n/k) + k$

$T(n/k) = T\left(\frac{n}{k \cdot k}\right) + k$

$T(n$

Note →

int a = 1
b = 2
c = 5 . . . . } → O(1) → O(constant time)

If, elseif, else } → O(1) → O(constant time)

T.C. (O (n^2))

```java
class NestedLoop {
 public static void main(String[] args) {
   int n = 10;
   int sum = 0;
   double pie = 3.14;
   for (int var = 0; var < n; var = var + 3) {
     System.out.println("Pie: " + pie);
     for (int j = 0; j < n; j = j + 2) {
       sum++;
       System.out.println("Sum = " + sum);
     }
   }
 }
}
```

for (var=0; var<n; var+=k) → n

var+=K

(j=0 ; j<n; j+=k) → n

0 → n
1 → n
2 → n
3 → n
⋮
n-1 → n

→ n times

→ (n * n)

O (n^2)

```java
class NestedLoop {
 public static void main(String[] args) {
   int n = 10; // O(time complexity of the called function)
   int sum = 0; //O(1)
   double pie = 3.14; //O(1)

   //O(?)
   for (int var = n; var >= 1; var = var - 3) {
     System.out.println("Pie: " + pie);
     //O(?)
     for (int j = n; j >= 0; j = j - 1) {
       sum++;
     }
   } //end of outer for loop
 System.out.println("Sum: " + sum);//O(1)
 } //end of main
} //end of class
```

var++ = k

var-- = k

var = n ; var >= 1 ; var-- = k → n

→ n

j = n ; j >= 0 ; j-- = k → n

n → n

n−1 → n

n−2 → n

2

1 → n

→ (n * n)

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

5

2

n−3

2

2 + 2

```java
class NestedLoop {
 public static void main(String[] args) {
   int n = 10;     //O(1)
   int sum = 0;   //O(1)
   int j = 1;    //O(1)
   double pie = 3.14;   //O(1)
    //O(?)
   for (int var = 1; var < n; var += 3) {
     System.out.println("Pie: " + pie);
     j = 1;
     while (j < n) { //O(?)
       sum += 1;
       j *= 3;
     }
   }
   System.out.println("Sum: " + sum); //O(1)
 } //end of main
} //end of class
```

$\log_3 n$

$j^x = k$

$O(var=1; var<n; var+=k) \rightarrow n$

$\rightarrow n$

$\rightarrow \log_3 n$

$1 \rightarrow \log_3 n$

$2 \rightarrow \log_3 n$

$3 \rightarrow \log_3 n$

10

$n-1 \rightarrow \log_3 n$

$(n-1)(\log_3 n)$

$\boxed{n \log_3 n} + \log_3 n$

$\boxed{O(n \log_3 n)}$

```java
class NestedLoop {
 public static void main(String[] args) {
   int n = 10;
   int sum = 0;
   double pie = 3.14;

   for (int var = 0; var < n; var++) {
     int j = 1;
     System.out.println("Pie: " + pie);
     while(j < var) {
       sum += 1;
       j *= 2;
     }
   } //end of for loop
   System.out.println("Sum: " + sum);
 } //end of main
} //end of class
```

var=0; var<n ; var++ → n

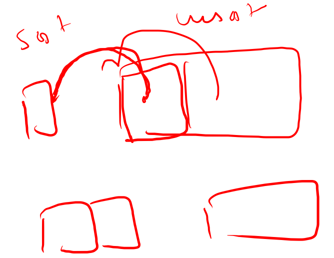
$\log_2 n$

$\log_4 n$

$\log_k n$

$\boxed{n * \log_2 n}$

T.C → $O(n^2)$ | sort ⟷ Insertion sort




```
void sort(int[] input) {
    for (int i = 1; i < input.length; i++) {
        int key = input[i];
        for (int j = i - 1; j >= 0; j--) {
            if (input[j] > key) {
                int tmp = input[j];
                input[j] = key;
                input[j + 1] = tmp;
            }
        }
    }
}
```

$i=1$ → 1 → 0

$i=2$ → 2 → 0, 1

$i=3$ → 3 → 0, 1, 2

$i=n-1$ → $n-1$ → 0, 1, ... $n-2$

$\frac{(n-1)(n)}{2}$ ⟵ $O(n^2)$ ⟵ $(1+2+3+ \cdots\ n-1)$ →

$\frac{n^2}{2} - \frac{n}{2}$

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++) {
    for (j = 2; j <= n; j = j * 2) {
        k = k + n / 2;
    }
}
```

$\frac{n}{2}$ $O(n)$

$\log_2 n$

$\boxed{n \log_2 n}$

Options:

1. O(n)
2. O(N log N)
3. O(n^2)
4. O(n^2Logn)

```
int a = 0, i = N;
while (i > 0) {
    a += i;
    i /= 2;
}
```

Options:

1. O(N)
2. O(Sqrt(N))
3. O(N / 2)
4. O(log N)

$o(\log_2^n)$

N, N/2, N/4, N/8

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

$\log_2^n$