1. hardcore x

generalised

```
for(int i=0 ;i≤7;i++)
```

2.

	Input	Output
TC1	3	0 1 1 2
TC2	0	0
TC3	5	0 1 1 2 3 5

Code

Input → Enter n / The ans is → ~~Enter n~~ ~~The ans is~~ 0 1 1 2

correct output / expected

Your output

wrong answer

```
int fun(int a)
{
  if (a==0) return 0;
}
```

```
void fun(int a)
{

}
main()
{
  int c= fun(a)
}
```

Time Complexity

Experimental

10am Start Time

Code } 3 minutes

10:03am End Time

Asymptotic Analysis

program : input dependent

(n)

linearly : n

quad : n^2

exp : c^n

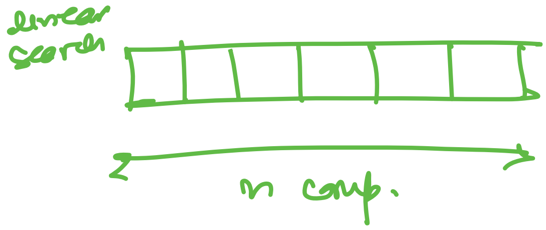
Worst Case

Best Case

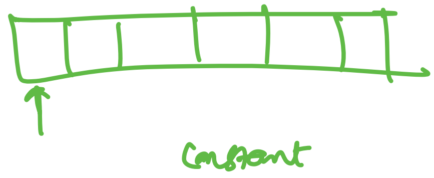
Avg Case

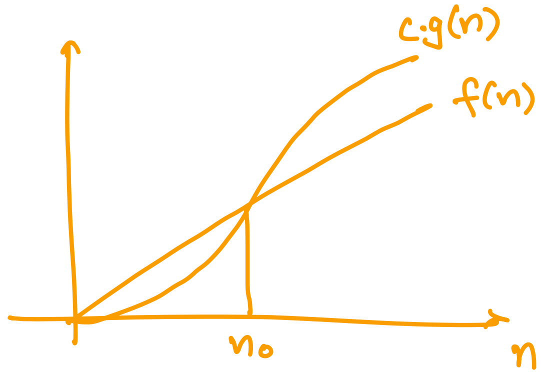
Worst Case (Big oh) O

Worst case time



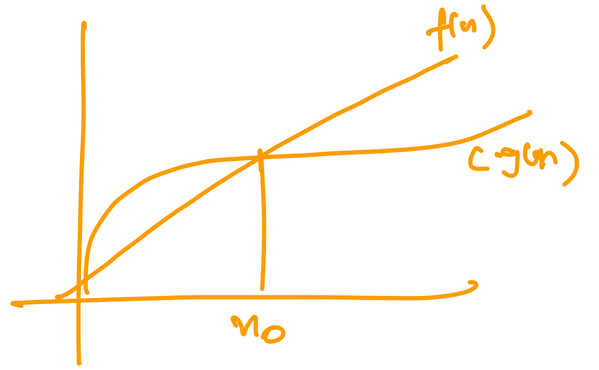
Best Case Ω (Omega)





$f(n) \le c \cdot g(n) \quad \forall\, n \ge n_0$

$f(n) = O(g(n))$



$f(n) \ge c \cdot g(n) \quad \forall\, n \ge n_0$

$f(n) = \Omega(g(n))$

```
for(int i=1 — n)
    pf(hello)
pf(bye)
```

max

i=1	i=2	i=3	i=4	i=5
1≤4	2≤4	3≤4	4≤4	5≤4 ✗
hello	hello	hello	hello	

$3 \cdot n$

$3n + 2 + 1$

$f(n) = 3n + 3$

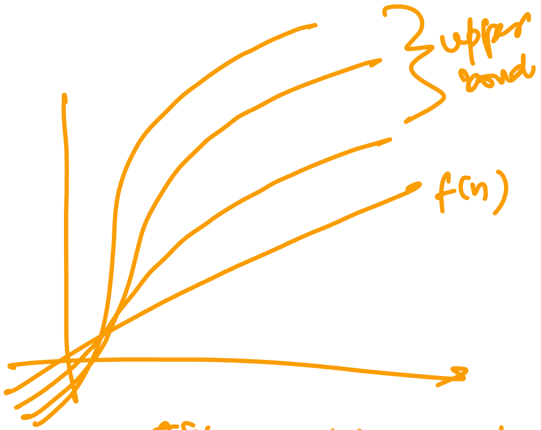
$f(n) \le c \cdot g(n) \quad \forall\, n \ge n_0$

$f(n) = O(g(n))$

$3n + 3 \le \underbrace{5}_{c} \cdot n \qquad n \ge 2$

$3n+3 = O(n)$

$3n+3 \le 5 \cdot n^2$



tighest upper bond

$f(n) = n^2+2n+3$

$O(n^2)$

$$n^2+2n+3 \le \underbrace{5}_{c} \cdot \underbrace{n^2}_{g(n)} \quad \forall\, \underbrace{n \ge 1}_{n_0}$$

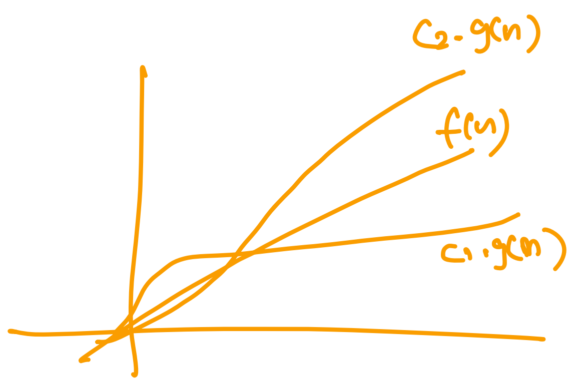
$n^2+2n+3 = O(n^2)$

Average Case:

Θ (Theta)

$c_1 \cdot g(n) \le f(n) \le c_2\, g(n)$

$f(n) = \Theta(g(n))$



Q: constant time

- pf(hello)
- int a
- int a = 200 + 1000 * 300

$\Big\}$ $O(1)$

↳ independent of n

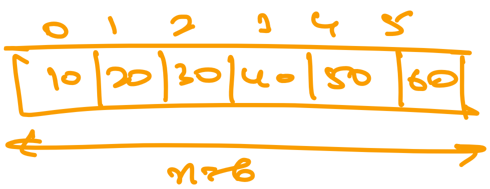
```
int n;
cin >> n;
pf(hello)
```

1
1
1
3

$\cancel{O(3)}$

Output: $O(1)$

Q: Linear Search

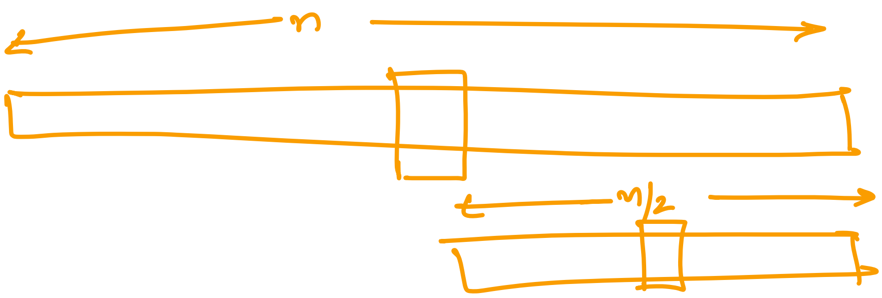


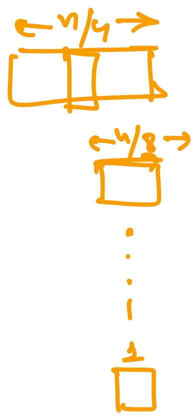
$O(n)$

$n-1 \rightarrow O(n)$

$n-2 \rightarrow O(n)$

Q: Binary Search





$1 \rightarrow n/2 \rightarrow n/2^1$

$2 \rightarrow n/4 \rightarrow n/2^2$

$3 \rightarrow n/8 \rightarrow n/2^3$

$\vdots$

$k \rightarrow 1 \rightarrow n/2^k$

$n = 2^k$

$k = \log_2 n$

Intuition:

$n \to n/2 \to n/4 \to n/8 \to n/16$

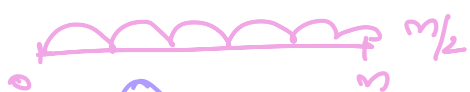
$\log_2 n$

$1 \to 2^1 \to 2^2 \to 2^3 \to \cdots \to n \; (2^{\log_2 n})$ — $\log_2 n$ steps

$\log_2 n + 1 \le c \cdot \log_2 n$

c, $g(n)$

$O(\log_2 n)$



Q: (a)

```
i=0;
while (i<=n)
{
    pf(DTU)
    i++;
}
```

$n+1$

$O(n)$

(b)

```
i=0
while(i<=n)
{
    pf(DTU)
    i=i+2;
```

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

$\frac{n}{2}+1 \le c \cdot g(n)$

$O(n)$

(c)

```
i=0
while(i<=n)
{
    pf(DTU)
    i=i+2;
    i=i+3;
}
```

$\frac{n}{5}$

$O(n)$

(d)

```
while(n>0)
{
    pf(DTU)
    n=n-1;
}
```

$O(n)$

(e)

```
while(n>0)
{
    pf(DTU)
    n=n-2;
}
```

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

(f)

```
while(n>0)
{
    pf(DTU)
    n=n-2;
    n=n-3;
}
```

$\frac{n}{5}$ $O(n)$

(g) while (n > 0)
{
pf(DTU)
n = n/2;
}
$\log_2 n$

(h) while (n > 0)
{
pf(DTU);
n = n/3;
}
$\log_3 n$

(i) while (n > 0)
{
pf(DTU)
n = n/2;
n = n/3;
}
$\log_6 n$

(h) while (2^n >= 1)
{
pf(DTU)
n = n/2;
}

$\log_2 2^n = n$

$2^n \downarrow 2^{n-1} \downarrow 2^{n-2} \downarrow \vdots \; 2^0$ — n+1 times, O(n)

Q:

```
for (int i=1; i<=n; i++)        → n
{
    for (int j=1; j<=n; j++)    → n
    {
        pf(DTU);
    }
}
```

n^2 dependency X

i=1	i=2	i=3	. . .	i=n
n times	n times	n times		n times

$n + n + n + \dots + n = n*n = n^2$

Q:

```
for (int i=1; i<=n; i++)
{
    for (int j=1; j<=i; j++)
    {
        pf(DTU);
```

Dependency : On row

}

}

$i=1$ $j=1$ time | $i=2$ $j=2$ time | $i=3$ $j=3$ time | ———— | $i=n$ $j=n$ time

$$\underbrace{1+2+3+ \text{———} n}_{\frac{n(n+1)}{2} = \frac{n^2}{2}+\frac{n}{2} = O(n^2)}$$

Q:

```
for (i=1 ; i<=n ; i++)
{
    for(j=1; j<=n; j=j+i)
    {
        pf(DTO);
    }
}
```

$i=1$ $j=n$ time | $i=2$ $j=\frac{n}{2}$ time | $i=3$ $\frac{n}{3}$ | $i=n$ $\frac{n}{n}$

$n+\frac{n}{2}+\frac{n}{3}+\cdots\frac{n}{n}$

$$n\underbrace{\left(1+\frac{1}{2}+\frac{1}{3}+\text{———}\frac{1}{n}\right)}_{?}$$

$n \log n$

$$\sum_{i=1}^{n}\frac{1}{i}$$

$$\int_{i=1}^{n}\frac{1}{i}=\log n$$

Q:

```
for(i=1; i<=k; i++)        → k time
{
    for(j=1; j<=n/k; j++)  → n/k time
    {
        pf(DTO);
    }
}
```

$$k \cdot \frac{n}{k} = O(n)$$

Q:

```
i=1, s=0;
while (i<=n)
{
    pf(DTO);
    s=s+i;
    i=i+1;
}
```

$\} \; O(n)$

Q:

```
i=1, s=0;
while (s<=n)
{
    pf(DTO);
    s=s+i;
    i=i+1;
}
```

$i=1,\ s=0$	$i=2$	$i=3$	. . . . . .	$i=k$
$0 \le n$				
$s=0+1$	$s=0+1+2$	$s=0+1+2+3$		$s=0+1+2\cdots k$
				$=\frac{k(k+1)}{2}$

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

$$k^2 \le n$$

$$k \le \sqrt{n}$$

$$O(\sqrt{n})$$

Prime No.

→ 2 factors: 1, no. itself

2
3
5
7
11
13
⋮

n = 7:

A1:

1 | n%1
⋮ | n%2
⋮ | n%3
n | ⋮
  | n%n

} 2 factors

7%1 ✓
7%2 ✗
7%3 ✗
7%4 ✗
7%5 ✗
7%6 ✗
7%7 ✓

} 2

A2:

2 … n-1 } factors = 0   n-2 times   O(n)

A3:

n = 18

n/2 = 9

2 … n/2

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

A4: $\sqrt{n}$

n = 36

$\sqrt{36} = 6$

2×18
3×12
4×9
~~6×6~~
9×4
12×3
18×2

loop 2 … $\sqrt{n}$

$i^2 \le n$

$i \le \sqrt{n}$

$n \longrightarrow \sqrt{n}$ ✓

1 ———— 1000

$1 \longrightarrow \sqrt{n}$

$n \longrightarrow n\sqrt{n}$??

Sieve of Eratosthenes (SOE):

$n = 25$

1: false
0: true

~~0~~	~~1~~	2	3	~~4~~	5
	~~6~~	7	~~8~~	~~9~~	~~10~~
	11	~~12~~	13	~~14~~	~~15~~
	~~16~~	17	~~18~~	19	~~20~~
	~~21~~	~~22~~	23	~~24~~	~~25~~

5

2 table multiples, 2

3 table

4 table ✗

5

$3 \times 2 = 6$

$3 \times 3 = 9$

$3 \times 4 = 12$

$3 \times 5 = 15$

$3 \times 6 = 18$

$3 \times 7 = 21$

$3 \times 8 = 24 \le n$

table — multiplier

$n = 25$

$n=2$: 2, 4, 6, 8, 10, 12, …, 25 $\Big\}\ n/2$

$n=3$: 3, 6, 9, 12, …, 25 $\Big\}\ n/3$

$n=4$: ×

$n=5$: 5, 10, 15, 20, …, 25 $\Big\}\ n/5$ ....

$$\frac{n}{2} + \frac{n}{3} + \frac{n}{5} + \frac{n}{7} + \frac{n}{11} + \frac{n}{13} + \cdots$$

$$n\Big(\underbrace{\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} \text{———}}_{\text{sum of reciprocals of prime}}\Big)$$

log log n

n log log n

$\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots$

log n

$n\sqrt{n} \longrightarrow n \log\log n$

$n = 2^{2^{1000}}$

$\log n = 2^{1000}$

$\log\log n = 1000$