

Algorithms

Algorithm

- ① Written at design time (Design)
- ② Domain Knowledge. A Person writing Algo should have domain knowledge.
- ③ Any language can be used.
- ④ Hardware and OS independent
- ⑤ Analyse. After writing an algo we will analyse it for time and space complexity

Program

- ① Written at Implementation Time (Implementation)
- ② Programming Knowledge. A Person writing Program must know some programming language.
- ③ Written Using Programming Language.
- ④ H/W and OS dependent
- ⑤ After writing a program we simply Test it. (Testing)

Proxi Analysis

- ① It is done on algorithm by studying it into greater details knowing how its working and we get some results.
- ② Independent of language
- ③ Hardware Independent
- ④ Time and space functions

Posterior Testing

- ① Program
- ② Language dependent
- ③ Hardware dependent
- ④ Watch time and Bytes.

characteristics of Algo →

- ① Input — can take 0 or more input
- ② Output — atleast 1 output
- ③ Definiteness — every statement should be clear, unambiguous and precise.
- ④ Finiteness
- ⑤ Effectiveness

How to write an Algo →

Algorithm swap (a, b)	⇒	Algo swap (a, b)
{		begin
temp = a		temp ← a
a = b		a ← b
b = temp		b ← temp
}		end.

How to Analyze an Algo →

Criteria on which we can analyze an algorithm.

- ① Time : An algo should be time efficient. After analyzing we get a time function.
- ② Space : How much memory space a Algorithm requires.
- ③ N/w Consumption : Most applications these days are cloud based, web based. So amount of data transferred can be one criteria.
- ④ Power Consumption.
- ⑤ CPU Registers. : If you are developing some device drivers then one criteria can be how many CPU registers it is using.

Algo: swap (a, b)

ϵ temp = a — 1
 $a = b$ — 1
 $b = temp$ — 1

3

$$f(n) = 3$$

constant time $\Rightarrow O(1)$ $O(1)$

space

a

b

temp

$$S(n) = 3 \text{ words constant.}$$

$$x = 5 * a + 6 * b \rightarrow 1 \text{ unit time}$$

In algo analysis we don't get watch time but we get time function.

Frequency Count Method \rightarrow

①

Algorithm Sum(A, n)

space

A [0 1 2 3 4]
[8 3 9 7 2]

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

ϵ s = 0 — 1
 For ($i = 0, i < n, i++$) — $n+1$

ϵ s = s + A[i] — n

3

return s.

3

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

$$O(n)$$

A $\rightarrow n$

n — 1

s — 1

i — 1

$$S(n) = n + 3$$

$$O(n)$$

②

Algo Add(A, B, n)

Time

space

ϵ For ($i = 0, i < n, i++$) — $n+1$
 ϵ For ($j = 0, j < n, j++$) — $n \times n+1$
 ϵ $c[i][j] = a[i][j] + b[i][j]$ — $n \times n$

3

3

$$f(n) = 2n^2 + 2n + 1$$

$$O(n^2)$$

$$3n^2 + 3$$

$$O(n^2)$$

$$A - n^2$$

$$B - n^2$$

$$C - n^2$$

$$n - 1$$

$$i - 1$$

$$j - 1$$

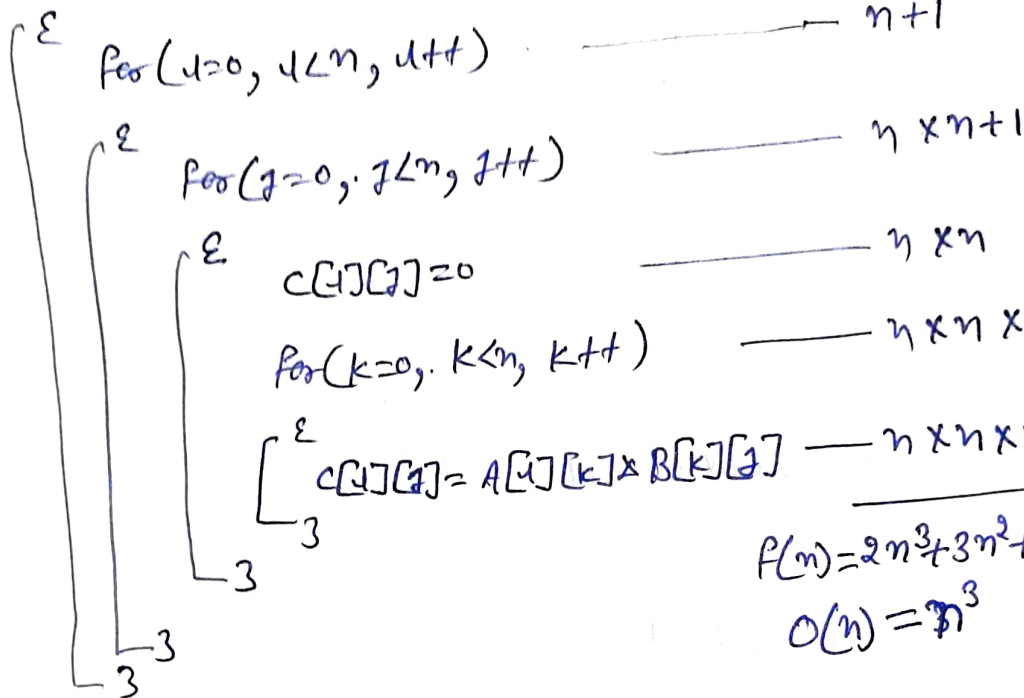
$$S(n) =$$

③

Algo Multiply (A, B, n)

Time

Space (B)

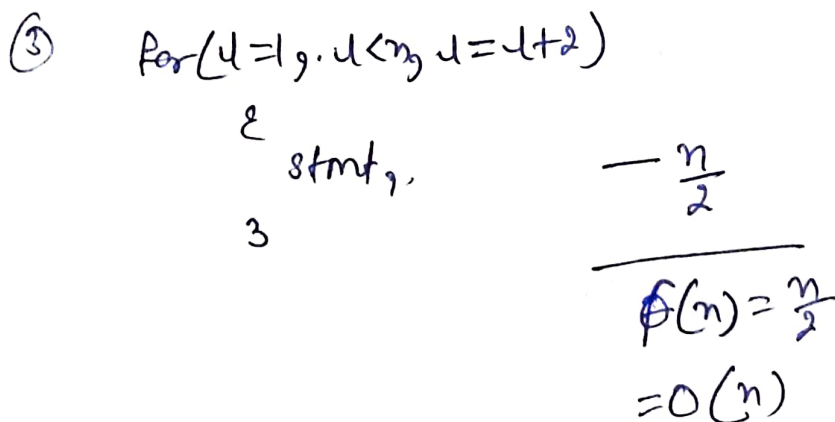
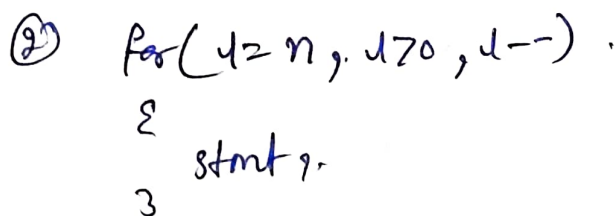
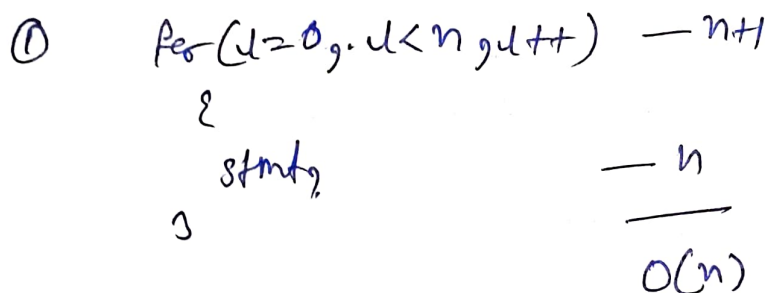


$s(n) = 3n^2 + 4$
 $O(n^2)$

$$T(n) = 2n^3 + 3n^2 + 2n + 1$$

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

#



base.
A n x n
n x n
n x n

(3)

For ($i=0, i < n, i++$)

$$\left[\begin{array}{l} \text{For } (j=0, j < i, j++) \\ \left[\begin{array}{l} \text{stmt}_1 \end{array} \right]_3 \end{array} \right]_3$$

(3)

i	j	#no of time stmt execute
0	0	0
1	0✓ 1x	1
2	0✓ 1✓ 2x	2
3	0✓ 1✓ 2✓ 3x	3
⋮		
n		n times

$$1+2+3+\dots+n = \frac{n(n+1)}{2}$$

$$P(n) = \frac{n^2+n}{2}$$

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

(4)

$P=0$

For ($i=1, P \leq n, i++$)

$$\left[\begin{array}{l} P = P + i \end{array} \right]_3$$

Assume P has become $\geq n$

$$\therefore P = \frac{k(k+1)}{2}$$

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

$$k^2 \geq n$$

$$k \geq \sqrt{n}$$

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

i	P
1	0+1=1
2	1+2=3
3	1+2+3
4	1+2+3+4
⋮	
k	1+2+3+4+...+k

⑤ For ($u=1, u \leq n, u = u \times 2$)

{
3 stmts

Assume u became 2^n

$$i.e. u \geq n$$

$$\therefore u = 2^k$$

$$\therefore 2^k \geq n$$

$$2^k = n$$

$$k = \log_2 n$$

$$O(\log_2 n)$$

$$u = 1 \times 2 \times 2 \times 2 \times \dots \times n$$

$$2^k = n$$

$$k = \log_2 n$$

$\log_2 n$ may give decimal values

Let $n=8$

$n=10$

$$\begin{array}{c} u \\ 1 \\ 2 \\ 4 \\ 8 \end{array} \left. \vphantom{\begin{array}{c} u \\ 1 \\ 2 \\ 4 \\ 8 \end{array}} \right\} 3 \text{ times}$$

$$\log 8 = 3$$

$$\log 2^3 = 3 \log_2 2 = 3$$

$$\lceil \log_2 n \rceil$$

$$\begin{array}{c} u \\ 1 \\ 2 \\ 4 \\ 8 \\ 16 \end{array} \left. \vphantom{\begin{array}{c} u \\ 1 \\ 2 \\ 4 \\ 8 \\ 16 \end{array}} \right\} 4 \text{ times}$$

$$\log 10 = 3.2$$

u

1

$$1 \times 2 = 2$$

$$2 \times 2 = 2^2$$

$$2^2 \times 2 = 2^3$$

\vdots
 2^k

For ($u=1, u \leq n, u++$)

{

stmts

}

$$u = 1 + 1 + 1 + 1 + 1 + \dots + 1 = n$$

$$k = n$$

⑥ For ($u=n, u \geq 1, u = u/2$)

{
3 stmts

Assume $u < n$

$$\therefore \frac{n}{2^k} < 1$$

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

$$n = 2^k$$

$$k = \log_2 n$$

$$O(\log_2 n)$$

$$\begin{array}{c} u \\ \frac{n}{2} \\ \frac{n}{2} \\ \frac{n}{2^2} \\ \frac{n}{2^3} \\ \vdots \\ \frac{n}{2^k} \end{array}$$

+

⑦

for($u=0, u < n, u++$)

{ stmt,

3.

$u * u < n$

$u * u \geq n$

$u^2 = n$

$u = \sqrt{n}$

$O(\sqrt{n})$

⑧

for($u=0, u < n, u++$)

{
 { stmt, }
 3 } $\rightarrow n$

for($j=0, j < n, j++$)

{
 { stmt
 3 } $\rightarrow n$

~~$O(n^2)$~~ $2n$

$O(n)$

9

$p=0$

for($u=1, u < n, u = u * 2$)

{
 {
 { stmt }
 3 } $\rightarrow p = \log n$

for($j=1, j < p, j = j * 2$)

{
 { stmt
 3 } $\rightarrow \log p$

$O(\log \log n)$

10

for($u=0, u < n, u++$) $\rightarrow n$

{
 {
 for($j=1, j < n, j = j * 2$) $\rightarrow n \times \log n$

 {
 { stmt
 3 } $\rightarrow n \times \log n$

 } $\rightarrow 2n \log n + n$

$O(n \log n)$

$$\text{For}(i=0, i < n, i++) \rightarrow O(n)$$

$$\text{For}(i=0, i < n, i=i+2) \rightarrow \frac{n}{2} O(n)$$

$$\text{For}(i=n, i > 1, i--) \rightarrow O(n)$$

$$\text{For}(i=1, i < n, i=i \times 2) \rightarrow O(\log_2 n)$$

$$\text{For}(i=1, i < n, i=i \times 3) \rightarrow O(\log_3 n)$$

$$\text{For}(i=n, i > 1, i=i/2) \rightarrow O(\log_2 n)$$

Q

```

A( )
{
    int i, j, k, n
    For(i=1, i <= n, i++)
    {
        For(j=1, j <= i, j++)
        {
            For(k=1, k <= 100, k++)
            {
                Print("Ravi");
            }
        }
    }
}
    
```

i=1	i=2	i=3	i=4	i=5	...	i=n
j=1 times	j=2 times	j=3	j=4 times	j=5 times	...	j=n
k=100 times	k=2x100 times	k=3x100 times	k=4x100 times	k=5x100	...	k=nx100

$$100 + 2 \times 100 + 3 \times 100 + \dots + n \times 100$$

$$100(1 + 2 + 3 + \dots + n)$$

$$= 100 \left(\frac{n(n+1)}{2} \right)$$

$$= O(n^2) \quad \text{Ans}$$

A()

S = 1 3 6 10 15 21 ... n
u = 1 2 3 4 5 6 ... k

$\frac{k(k+1)}{2}$ ⑦

u=1, s=1

while (s <= n)

u++

s = s + u

Printf("Ravi"),

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

$\frac{k^2 + k}{2} > n$

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

Q

A()

int u, j, k, n

for (u=1; u <= n; u++)

for (j=1; j <= u^2; j++)

for (k=1; k <= $\frac{n}{j}$; k++)

Printf("ABC"),

u=1 u=2 u=3 ... u=n

j = 1 time, j=4 j=9 j=n^2

k = $\frac{n}{j} \times 1$ k = $\frac{n}{j} \times 4$ k = $\frac{n}{j} \times 9$ k = $\frac{n}{j} \times n^2$

$\frac{n}{j} \times 1 + \frac{n}{j} \times 4 + \frac{n}{j} \times 9 + \dots + \frac{n}{j} \times n^2$

$\frac{n}{j} (1 + 4 + 9 + \dots + n^2)$

$\frac{n}{j} \left(\frac{n(n+1)(2n+1)}{6} \right)$

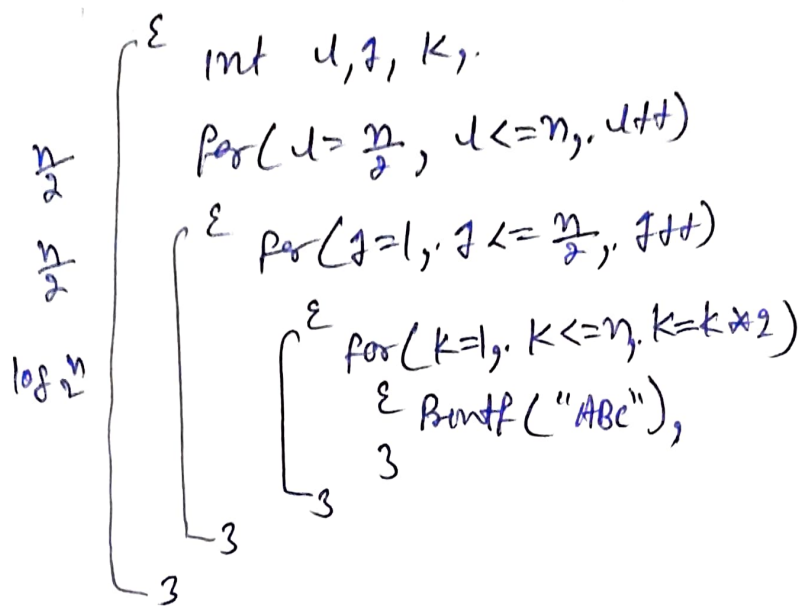
$O(n^4)$

sum of squares of first n numbers

Q

A ()

Q3



$$\frac{n}{2} \times \frac{n}{2} \times \log_2 n$$

$$O(n^2 \log_2 n)$$

Analysis of if and while →

①

u = 0

①

while (u < n)

{ stmt,
u++,
}

n+1

n

n

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

for (u = 0, u < n, u++) → n+1

{

stmt,
u++

}

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

for (u = 1, u < n, u = u * 2)

{ stmt
}

②

a = 1

while (a < b)

{ stmt
a = a * 2
}

a b

1

1 * 2 = 2

2 * 2 = 4

4 * 2 = 8

...

2^k

Terminated when a ≥ b

$$2^k \geq b$$

$$2^k = b$$

$$k = \log_2 b$$

$$O(\log_2 b)$$

③

```

d = n
while (d > 1)
{
    stmt
    d = d/2
}

```

$O(\log n)$

④

```

d = 1
k = 1
while (k < n)
{
    stmt
    k = k + d
    d++,
}

```

d	k
1	1
2	1+1=2
3	2+2=4
4	2+2+3=7
5	2+2+3+4
!	!
m	2+2+3+4+...+m

or Using For

```

for (k=1, d=1, k < n, d++)
{
    stmt
    k = k + d;
}

```

Assume $k \geq m$

$$\frac{m(m+1)}{2} \geq n.$$

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

$$\frac{m^2 + m}{2} \geq n$$

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

⑤

```

while (m != n)
{
    if (m > n)
        m = m - n
    else
        n = n - m
}

```

①

m	n
6	3
3	3

1 time.

②

m	n
5	5

0 times

③

m	n
16	2
14	2
12	2
10	2
8	2
6	2
4	2
2	2

7 times

$\frac{14 \text{ times}}{2}$

$O(n)$

Algo Test(n)

```

{
  if (n < 5)
    print n. — 1    O(1) Best
  else
    for (i = 0, i < n, i++)
      {
        print i — n
      }
}

```

$O(n)$ worst