Algorithms —

Algorithm	Program
① Written at design time (Design)	① Written at Implementation Time (Implementation)
② Domain Knowledge. A Person writing Algo should have domain knowledge.	② Programming Knowledge. A Person writing Program must know some Programming language.
③ Any Language can be used.	③ Written Using Programming Language.
④ Hardware and OS independent	④ H/w and OS dependent
⑤ Analyse. After writing an algo we will analyse it for time and space complexity	⑤ After writing a program we simply Test it. (Testing)

Priori Analysis	Posteriory Testing
① It is done on algorithm by studying it into greater details knowing how its working and we get some results.	① Program
② Independent of Language	② Language dependent
③ Hardware Independent	③ Hardware dependent
④ Time and space functions	④ Watch time and Byte.

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)
{
    temp = a
    a = b
    b = temp
}
```

⇒

```
Algo swap (a,b)
Begin
    temp ← a
    a ← b
    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 analysing we get a time function.
2. space : How much memory space a Algorithm requires.
3. N/w Consuption: Most applications these days are cloud based, web based. So amount of data transfered 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)

```
{
    temp = a   — 1
    a = b      — 1
    b = temp   — 1
}
3
```

space
a
b
temp

$f(n) = 3$

$S(n) = 3$ words constant

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

$x = 5*a + 6*5 \rightarrow$ 1 Unit time

In algo analysis we dont get watch time but we get time function.

Frequency Count Method →

① Algorithm Sum(A, n)

	0	1	2	3	4
A	8	3	9	7	2

```
{
    s = 0                                   — 1
    For(i=0, i<n, i++)                      — n+1
    {
        s = s + A[i]                        — n
    }
    return s.                               — 1
}
```

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

$f(n) = 2n + 3$

$O(n)$

space
A → n
n — 1
s — 1
i — 1

$S(n) = n + 3$

$O(n)$

② Algo Add (A, B, n)

Time

```
{
    For(i=0, i<n, i++)                      — n+1
    {
        For(j=0, j<n, j++)                  — n × n+1
        {
            c[i][j] = a[i][j] + b[i][j]     — n × n
        }
    }
}
```

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

$O(n^2)$

space
A — n^2
B — n^2
C — n^2
n — 1
i — 1
j — 1

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

$O(n^2)$

③ Algo Multiply (A, B, n)

```
{                                                    Time
  For(i=0, i<n, i++)                                 n+1
  {
    For(j=0, j<n, j++)                               n x n+1
    {
      c[i][j]=0                                      n x n
      For(k=0, k<n, k++)                             n x n x n+1
      [ c[i][j]= A[i][k]*B[k][j]                     n x n x n
    }
  }
```

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

$O(n) = n^3$

Space ⑤

A	n
B	n
C	n
n	1
i	1
j	1
k	1

$S(n) = 3n^2 + 4$

$O(n^2)$

①
```
For(i=0, i<n, i++)        — n+1
{
  stmt;                   — n
}
```
$O(n)$

②
```
For(i=n, i>0, i--)
{
  stmt;
}
```
$O(n)$

③
```
For(i=1, i<n, i=i+2)
{
  stmt;                   — n/2
}
```
$O(n) = \frac{n}{2}$

$= O(n)$

③

```
For(i=0; i<n; i++)
{
    for (j=0; j<i; j++)
    {
        stmt;
    }
}
```

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

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

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

$O(n) = n^2$

④

```
P=0
For(i=1; P<=n; i++)
{
    P = P+i;
}
```

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

Assume P has became $> n$

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

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

$k^2 > n$

$k = \sqrt{n}$

$O(\sqrt{n})$

⑤ `for(i=1; i<n; i=i*2)`

```
{
  stmt;
}
```

Assume i became $i>=n$

i.e $i>=n$

$\because i = 2^k$

$\because 2^k >= n$

$2^k = n$

$k = \log_2 n$

$O(\log_2 n)$

i
1
$1\times2=2$
$2\times2=2^2$
$2^2\times2=2^3$
⋮
2^k

`for(i=1; i<=n; i++)`

```
{
  stmt;
}
```

$i = 1+1+1+1+1+1+\cdots+1 = n$

$k = n$

$i = 1\times2\times2\times2\times2\cdots n$

$2^k = n$

$k = \log_2 n$

$\log_2 n$ may give decimal values

Let $n=8$

i
1
2
4
8 ✗

3 time

$\log 8 = 3$

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

Let $n=10$

i
1
2
4
8
16 ✗

4 time

$\log 10 = 3.2$

$\lceil \log_2 n \rceil$

⑥ `for(i=n; i>=1; i=i/2)`

```
{
  stmt;
}
```

Assume $i<n$

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

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

$n = 2^k$

$k = \log_2 n$

$O(\log_2 n)$

i
n
$\frac{n}{2}$
$\frac{n}{2^2}$
$\frac{n}{2^3}$
⋮
$\frac{n}{2^k}$

⑦

```
for(i=0; i*i<n; i++)
{
    stmt;
}
```

$i*i < n$

$i*i > n$

$i^2 = n$

$i = \sqrt{n}$

$O(\sqrt{n})$

⑧

```
for(i=0; i<n; i++)
{
    stmt;          —— n
}

for(j=0; j<n; j++)
{
    stmt           —— n
}
```

$2n$

$O(n)$

9

```
p=0
for(i=1; i<n; i=i*2)
{
    p++;           —— p = log n
}
for(j=1; j<p; j=j*2)
{
    stmt           —— log p
}
```

$O(\log \log n)$

10

```
for(i=0; i<n; i++)                  —— n
{
    for(j=1; j<n; j=j*2)            —— n x log n
    {
        stmt                        —— n x log n
    }
}
```

$2n \log n + n$

$O(n \log n)$

```
For(i=0; i<n; i++)  ⟶ O(n)
For(i=0; i<n; i=i+2) ⟶ n/2 O(n)
For(i=n; i>1; i--)  ⟶ O(n)
For(i=1; i<n; i=i*2) ⟶ O(log2 n)
For(i=1; i<n; i=i*3) ⟶ O(log3 n)
For(i=n; i>1; i=i/2) ⟶ O(log2 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++)
      {
        Printf("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=2×100 times	k=3×100 times	k=4×100 times	k=5×100		k=n×100

$$100 + 2\times100 + 3\times100 + \cdots n\times100$$

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

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

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

```
A()
{
  i=1, s=1
  while (s<=n)
  {
    i++,
    s=s+i
    Printf("Ravi");
  }
}
```

s =	1	3	6	10	15	21	...	n $\left(\frac{k(k+1)}{2}\right)$
i =	1	2	3	4	5	6	...	k

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

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

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

(8)

```
A()
{
  int i, j, k, n
  For(i=1; i<n; i++)
  {
    For(j=1; j<=i^2; j++)
    {
      For(k=1; k<=n/2; k++)
      {
        Printf("ABC");
      }
    }
  }
}
```

i = 1	i = 2	i = 3	...	i = n
j = 1 times	j = 4	j = 9		$j = n^2$
$k = n/2 \times 1$	$k = \frac{n}{2} \times 4$	$k = \frac{n}{2} \times 9$		$k = \frac{n}{2} \times n^2$

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

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

squares of first n

$\frac{n}{2}\left(\frac{n(n+1)(2n+1)}{6}\right)$ sum of squares of natural numbers

$O(n^4)$

(4) A( )

```
{
  int i, j, k;
  for(i = n/2; i<=n; i++)          // n/2
  {
    for(j=1; j<=n/2; j++)          // n/2
    {
      for(k=1; k<=n; k=k*2)        // log2 n
      {
        printf("ABC");
      }
    }
}
```

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

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

Analysis of if and while →

(1)

```
i=0                          ——— 1
while(i<n)                   ——— n+1
{
  stmt;                      ——— n
  i++;                       ——— n
}
```

$f(n) = 3n+2$

```
for(i=0; i<n; i++)           → n+1
{
  stmt;                      → n
}
```

$f(n) = 2n+2$

```
for(i=1; i<n; i=i*2)
{
  stmt
}
```

(2)

```
a=1
while(a<b)
{
  stmt
  a=a*2
}
```

a	b
1	
1×2=2	
2×2=2²	
2²×2=2³	
⋮	
2^k	

Terminate when $a \geq b$

$2^k \geq b$

$2^k = b$

$k = \log_2 b$

$O(\log_2 b)$

③

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

$O(\log n)$

④

```
i=1
k=1
while(k<n)
{
  stmt
  k=k+i
  i++;
}
```

i	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

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

Assume $k \geq n$

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

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

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

Or Using For

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

⑤

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

(a)

m	n
6	3
3	3

1 time.

(b)

m	n
5	5

0 times

(c)

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

7 times

$\frac{16}{2}$ times

$O(n)$

```
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
    }
  }
}
```