→ Algorithms → sequence of steps on data using efficient data structures to solve a given problem.

→ Algorithm	Program
• Design	• Implementation
• Person with Domain knowledge writes Algorithm.	• Programmer writes the program.
• Any language	• Programming language
• Hardware and Operating System Independent.	• Hardware and Operating System Dependent.
• Analyse	• Testing

→ Priori Analysis	Posteriori Testing
• Algorithm	• Program
• Independent of language	• language Dependent
• Hardware Independent	• Hardware Dependent
• Time and Space Function	• Watch Time and Bytes.

Date
Page

→ Characteristics of Algorithm

1. Input - 0 or more
2. Output - atleast one output
3. Definiteness
4. Finiteness
5. Effectiveness

→ How to write an Algorithm

• Algorithm Swap (a, b)

start
    temp = a;
    a = b;
    b = temp;
end

→ How to Analyze an Algorithm

1. Time 2. Space 3. Network

3. Power 5. CPU Registers

Date |
Page |

	Time	Space
start		
temp = a;	– 1	a - 1
a = b;	– 1	b - 1
b = temp;	– 1	temp - 1
end	f(n) = 3	S(n) = 3 word

--- * --- * --- * --- * ---

→ Frequency count Method

- Sum of Array — A is Array, where n is number

Algorithm sum (A, n) — A = | 8 | 5 | 4 | 2 | 9 |

start — n = 5

```
    S=0;                          — 1
    for(i=0; i<n; i++) {          — n+1
        S = S+A[i];               — n
    }
    returns S;                    — 1
end
```

$f(n) = 2n+3$

$O(n)$

space: A = n, S = 1, n = 1, i = 1

$S(n) = n + 3$

- Sum of two square matrices — $n \times n \Rightarrow$ Matrix

Algorithm Add(A, B, n)

Start

```
    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 - $A = n^2$, $B = n^2$, $C = n^2$, $i = 1$, $n = 1$, $j = 1$ = $S(n) = 3(n^2+3)$

$O(n^2)$

- Multiplication of two square matrices

```
Algorithm Multiply (A, B, n)
start
    for(i=0; i<n; i++)          - n+1
    {
        for(j=0; j<n; j++)      - n(n+1)
        {
            c[i,j]=0;           - n×n
            for(k=0; k<n; k++)  - (n+1)×n×n
            {
                c[i,j] = c[i,j] + A[i,k]*B[k,j];
                                    - n×n×n
            }
        }
    }
end
```

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

$O(n^3)$

space :

A	$-n^2$
B	$-n^2$
C	$-n^2$
n	-1
j	-1
i	-1
k	-1

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

$O(n^2)$

(1) → Time complexity :

1. start

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

2. start

```
    for ( i=n; i>0; i-- ) {      → O(n)
        stmt ;
    }
end
```

3. start

```
    for ( i=1 ; i<n; i=i+2)
    {                            ↳ If this is i = i + 20
        stmt;     - n/2                    - n/20
    }
end
```

$f(n) = \frac{n}{2}$ still order will be same

$O(n)$