



# Design & Analysis of Algorithms



GeeksforGeeks

**Topic Name: Analysis of Algorithms**



# KNOW YOUR MENTOR

## Dr. Khaleel Khan Sir

- 30 Years of Teaching Experience
- Ph.D in Computer Science
- Expert of OS and DSA



My Official Telegram Channel  
**<https://t.me/KhaleelSirGate>**



# Topics To be covered

- 💡 Algorithm Concept
- 💡 Need for Analysis
- 💡 Methodology of Analysis
- 💡 **Apriori Analysis**





GeeksforGeeks





# Schedule

## 1. Analysis of Algorithms

1.1 Algorithm Concept and Lifecycle

1.2 Analysis of Algorithms - Need

1.3 Methodology & Types of Analysis ; W.C ; B.C ; A.C

★ ★ 1.4 Asymptotic Notations ( ASN )

★ 1.5 Framework for Analyzing Recursive Algorithms

★ 1.6 Apriori analysis of Non-Recursive Algorithms

★ 1.7 Loop Complexities

1.8 Space Complexity

1.9 Mathematical Background ★

# Design Strategies

## 2. Divide & Conquer

2.1 General Method

★ 2.2 Max-Min Problem

★★ 2.3 Merge Sort

★ 2.4 Binary Search

★★ 2.5 Quick Sort

2.6 Matrix Multiplication

2.7 Long Integer Multiplication (LIM)

★ 2.8 Master Method for D and C Recurrences

★ 2.9 Recursion Tree



### 3. Greedy Method

3.1 General Method

★ 3.2 Knapsack Problem

3.3 Job Sequencing with Deadlines

3.4 Optimal Merge Patterns

★ 3.4.1 Huffman Coding

3.5 Minimum Cost Spanning Trees

★★

3.5.1 Prims Method

3.5.2 Kruskal's Method

★ 3.6 Dijkstras Shortest Paths Problem

(Graph Theory)

Single Source Shortest Paths  
(SSSP)



## 4. Dynamic Programming (DP)

- 4.1 The Method
- 4.2 Difference between DP, Greedy Method and DandC
- 4.3 Multistage Graphs
- 4.4 Travelling Salesperson Problem
- 4.5 Binary Knapsack Problem
- ★ 4.6 All Pairs Shortest Paths : Floyd-Warshall's Algo
- 4.7 Bellman-Ford Single Source Shortest Paths
- ★ 4.8 Longest Common Subsequence (LCS)
- ★ 4.9 Matrix Chain Multiplication (MCP)
- ★ 4.10 Sum of Subsets
- 4.11 Reliable System Design
- ★ 4.12 Optimal Cost Binary Search Tree



## 5. Graph Algorithms

5.1 Representation of Graphs

5.2 Graph Traversals

### DFS

5.2.1 Undirected Connected Graphs

5.2.2 Undirected Disjoint Graphs: DFT

5.2.3 Directed Graphs & Types of Edges

5.2.4 DAG (Topological Sort)

### BFS

5.2.5 FIFO BFS

5.2.6 LIFO BFS

5.2.7 LC BFS

5.3 Parenthesization Theorem

5.4 Components

→ Connected Components

→ Strongly Connected Components  
KOSARAJU

→ Biconnected Components

→ Articulation Points



(Algo)

## 6. Heap Algorithms

- 6.1 Operations : Create, Insert, Delete, Modify
- 6.2 Applications : Heapsort

## 7. Sets

- 7.1 Representations
- 7.2 Operations → (UNION & FIND)

## 8. Sorting Algorithms

- 8.1 Basic terminology
- 8.2 Methods
  - 8.2.1 Bubble Sort
  - 8.2.2 Selection Sort
  - 8.2.3 Insertion Sort
  - 8.2.4 Radix Sort

## 9. Backtracking & Branch and Bound – An Overview

## Text – Books

1. Introduction to Algorithms – Cormen ( CLRS Book )
2. Fundamentals of Algorithms – Horowitz and Sahni



# Pre-Requisites

## (i) Math Background

- ★ → In's
- Algebra
- Calculus
- ★ → Summation Series
- ★ → Logarithms
- ★ → Exponentials
- Combinatorics

→ Probability

★ → Recurrences

→ Data Structures

→ Programming exposure





● Muhammad ibn Mūsā al-Khwārizmī

Shutterstock / German Visuals

Persian mathematician Muḥammad ibn Māsā al-Khwārizmī, sometimes known as the father of algebra, was one of the most influential thinkers of all time. He revolutionised algebra and his seminal works in mathematics, astronomy and geography have proved to be the keystone for centuries of advances across the world.

Al-Khwarizmi was born in about AD 780, and although his birthplace isn't known for sure, the al-Khwārizmī in his name can mean "the native of Khwarazm", which at the time was part of Greater Iran, but is now part of Turkmenistan and Uzbekistan, so many believe he grew up in that area.

Al-Khwārizmī worked at and then became director of the [House of Wisdom](#) in Baghdad, in modern-day Iraq, which was the capital of the Islamic empire at the time. At this centre of scientific research and teaching, he oversaw the translation of many major Greek and Indian mathematical and astronomy works into Arabic. He also produced original work that had a lasting influence on the advance of Muslim and European mathematics.



## The founder of algebra

The terms algebra and **algorithm** are derived from al-Khwārizmī's name and his work. A Latinisation of his name as *Algoritmi* led to the term "algorithm". And the word algebra comes from *al-jabr* in the title of a landmark book he wrote in about AD 820, *al-Kitāb al-Mukhtaṣar fī Ḥisāb al-Jabr wal-Muqābalah*, or *The Compendious Book on Calculation by Completion and Balancing*. The book introduced fundamental methods for solving equations and established the discipline of algebra.



# Introduction to Algorithm

*Algorithms* are the threads that tie together most of the subfields of *Computer Science*.

Something magically beautiful happens when a sequence of commands and decisions is able to marshal a collection of data into organized patterns or to discover hidden structure.

**Donald Knuth**

Defn:

Consists of Finite Set of Steps Statements to solve a  
given Problem;

Problem

↓  
Soln (Algorithm)

(Computer Algo)

- Algo May accept 0/more Inputs
- But must Produce atleast one OP;

Step | Stmt  
→ 1/more basic (Fund.  
operation)



# Algorithm Test

{

1.  $x \leftarrow y + 8;$

2. for  $i \leftarrow 1$  to  $n$   
 $a = b * c;$

3. for  $i \leftarrow 1$  to  $n$   
for  $j \leftarrow 1$  to  $n$   
 $y = y * x;$

4. Push( $s, x$ )

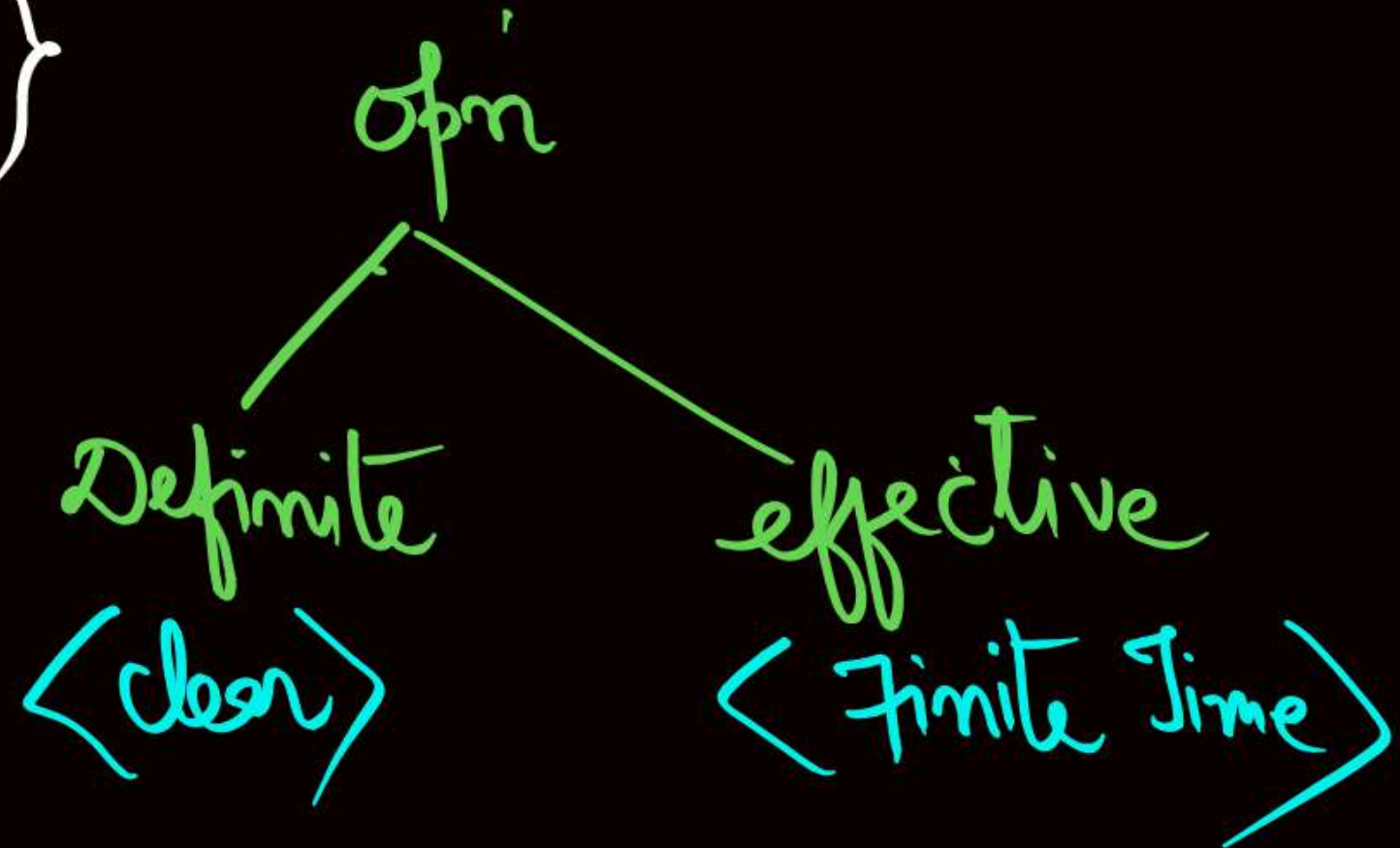
5. fn( $x, y, z$ );

Add; Assignment; Load

6. Read( $x$ );

7. write( $y$ );

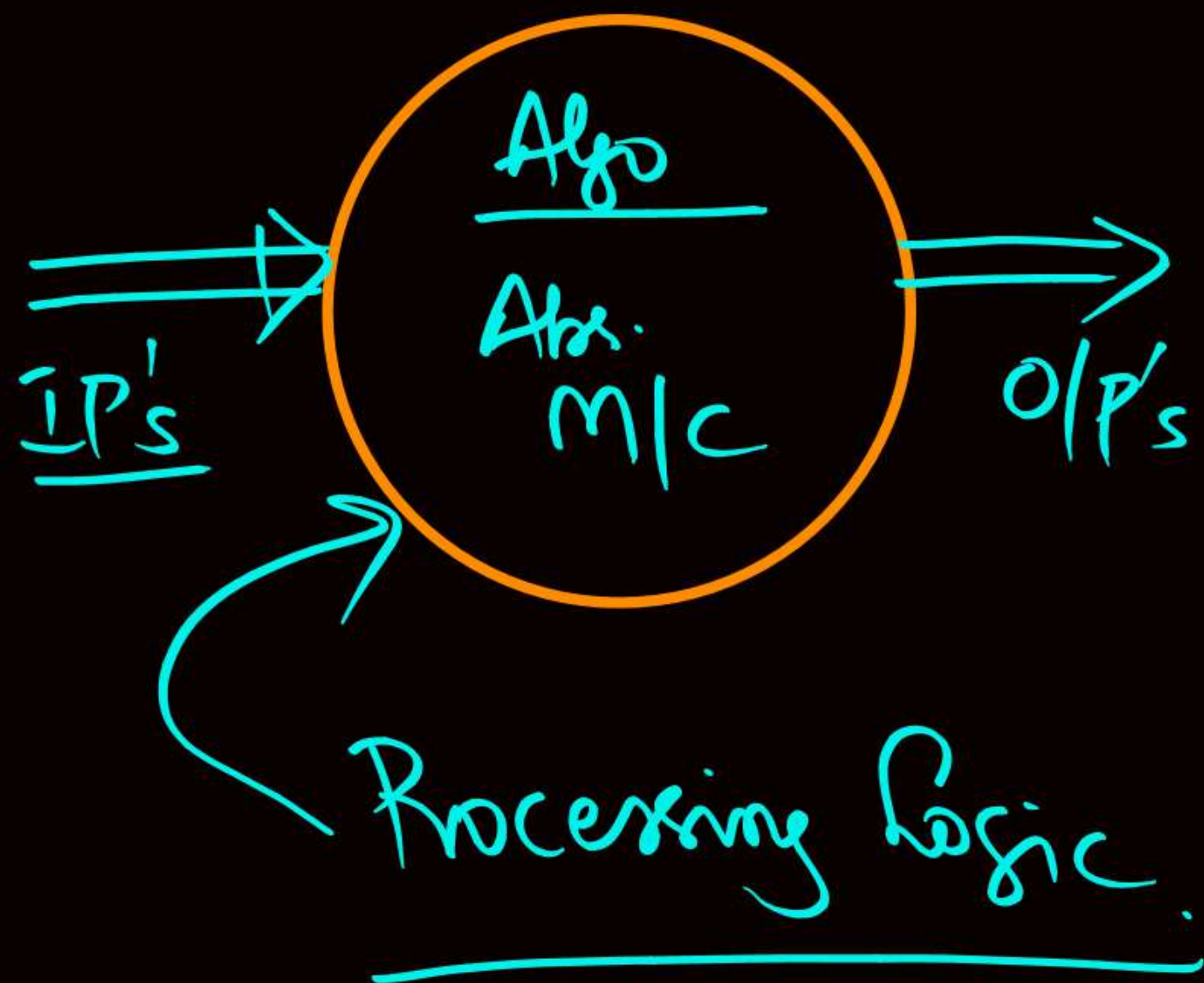
}



## Ex's 9 Basic / Fund. opns

- (i) Arithmetic:  $+, *, -, /, \%$
- (ii) Assignment:  $x = y$
- (iii) Relational, :  $x == y; x < y$   
opns
- (iv) I/O opns: read/write





## **Effective method (or Procedure)**

# **Definition**

A procedure that reduces the solution of some class of problems to a series of rote steps which, if followed to the letter, and as far as may be necessary, is bound to:

always give some answer rather than ever give no answer.

- ❖ always give the right answer and never give a wrong answer.
- ❖ always be completed in a finite number of steps, rather than in an infinite number
- ❖ work for all instances of problems of the class.

## **Algorithm**

An effective method expressed as a finite list of well-defined instructions for calculating a function.



Informally, an *Algorithm* is any well-defined computational procedure that takes some value, or set of values, as *input* and produces some value, or set of values, as *output* in a finite amount of time. An algorithm is thus a sequence of computational steps that transform the input into the output.

One can also view an Algorithm as a tool for solving a well-specified *computational problem*. The statement of the problem specifies in general terms the desired input/output relationship for problem instances, typically of arbitrarily large size. The algorithm describes a specific computational procedure for achieving that input/output relationship for all problem instances.

*CLRS Book ...*



An Algorithm for a computational problem is **correct** if, for every problem instance provided as input, it **halts** – finishes its computing in finite time – and outputs the correct solution to the problem instance.

A **correct Algorithm solves** the given computational problem.

An **incorrect Algorithm** might not halt at all on some input instances, or it might halt with an incorrect answer.

.....



# Properties of an Algorithm

An algorithm must possess the following properties:

**Finiteness** : The algorithm must always terminate after a finite number of steps.

**Definiteness** : Each step must be precisely defined; the actions to be carried out must be rigorously and unambiguously specified for each case.

**Input** : An algorithm has zero or more inputs, taken from a specified set of objects.

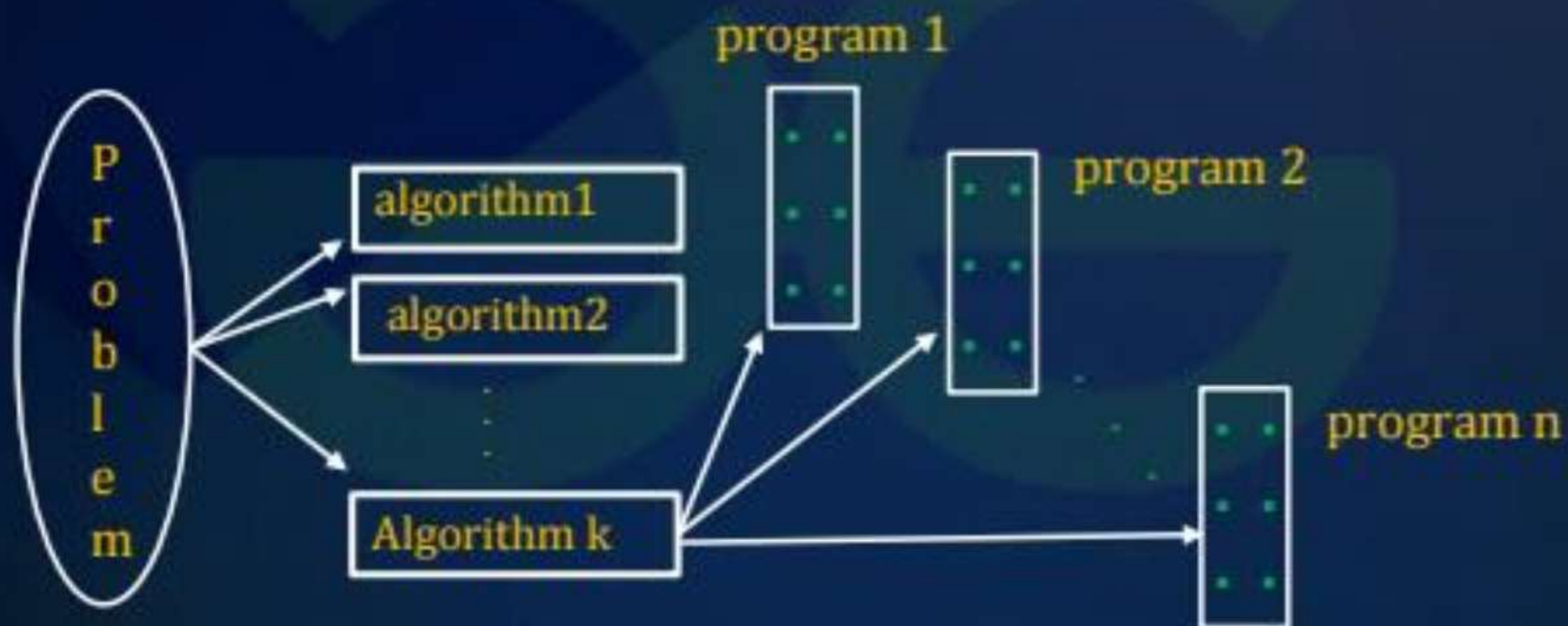
**Output** : An algorithm has one or more outputs, which have a specified relation to the inputs.

**Effectiveness** : All operations to be performed must be sufficiently basic that they can be done exactly and in finite length.



# Problems Vs Algorithms Vs Programs

For each problem or class of problems there may be many different algorithms.  
For each algorithms, there may be many different implementations (programs).





# Expressing Algorithms

An algorithm may be expressed in a number of ways, including:

**Natural Language** : usually verbose and ambiguous

**Flow Charts** : avoid most (if not all) issues of ambiguity; difficult to modify w/o specialized tools; largely standardized

**Pseudo-Code** : also avoids most issues of ambiguity; vaguely resembles common elements of programming languages; no particular agreement on syntax

**Programming Language** : tend to require expressing low-level details that are not necessary for a high-level understanding

.. SPARK



# Common Element of Algorithms

## *acquire data (input)*

some means of reading values from an external source; most algorithms require data values to define the specific problem (e.g., coefficients of a polynomial)

## *Computation*

some means of performing arithmetic computations, comparisons, testing logical conditions, and so forth...

## *Selection*

some means of choosing among two or more possible courses of action, based upon initial data, user input and/or computed results

*iteration* some means of repeatedly executing a collection of instructions, for a fixed number of times or until some logical condition holds

## *report results (output)*

some means of reporting computed results to the user, or requesting additional data from the user



## Problems Solved by Algorithms

The Human Genome Project has made great progress toward the goals of identifying all the roughly 30,000 genes in human DNA, determining the sequences of the roughly 3 billion chemical base pairs that make up human DNA, storing this information in databases, and developing tools for data analysis. Each of these steps requires sophisticated algorithms, **Dynamic Programming**.

- ❖ The Internet enables people all around the world to quickly access and retrieve large amounts of information. With the aid of clever algorithms, sites on the internet are able to manage and manipulate this large volume of data. Examples of problems that makes essential use of algorithms include **finding good routes** on which the data travels, and using a **search engine** to quickly find pages on which particular information resides.



- ❖ Electronic commerce enables goods and services to be negotiated and exchanged electronically, and it depends on the privacy of personal information such as credit card numbers, passwords, and bank statements. The core technologies used in electronic commerce include public-key cryptography and digital signatures which are based on **numerical algorithms and number theory**.
- ❖ You need to compress a large file containing text so that it occupies less space. Many ways to do so are known, including "LZW compression, "which looks for repeating character sequences- **Huffman Coding**



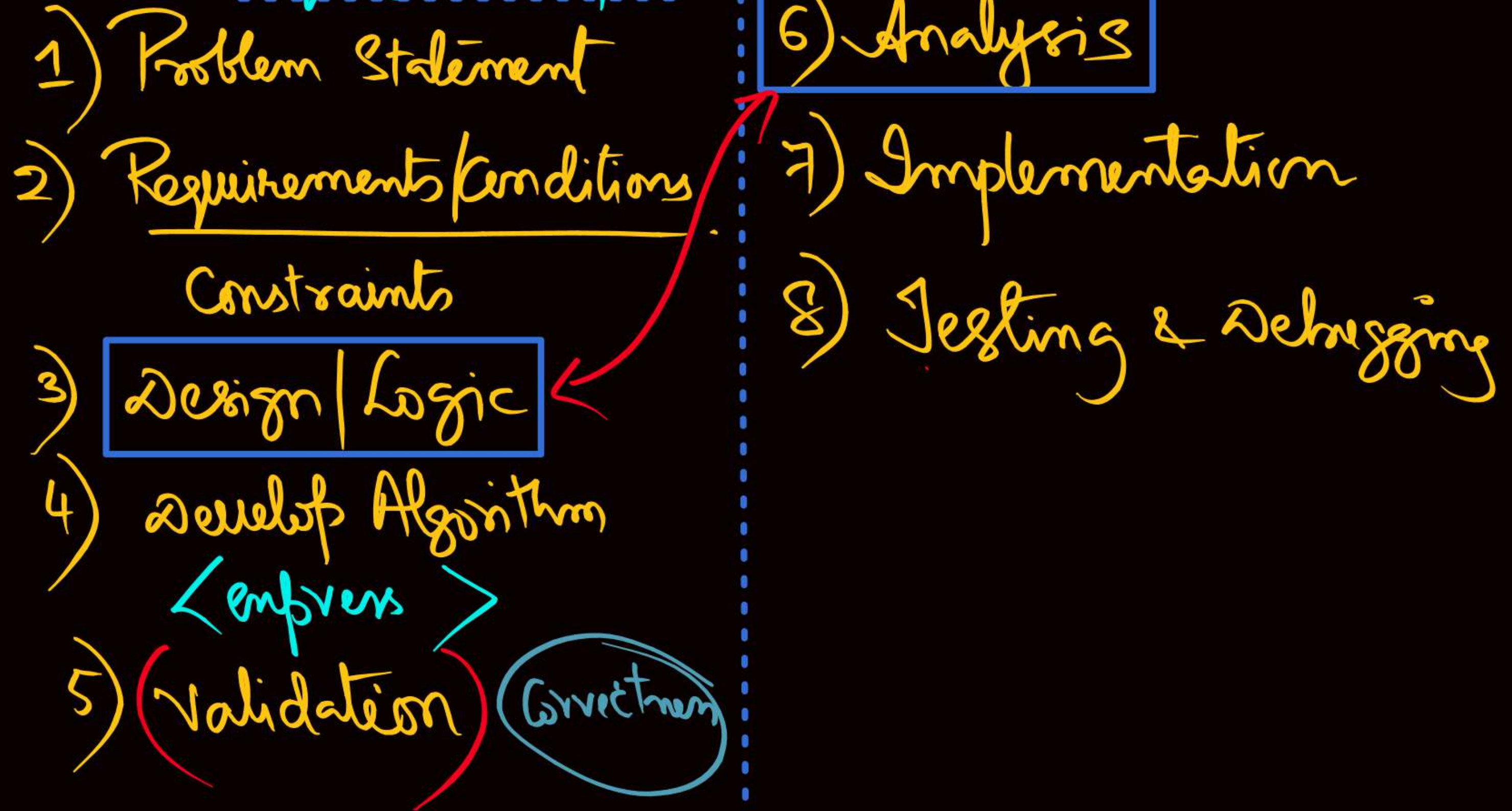
- ❖ Manufacturing and other commercial enterprises often need to allocate scarce resources in the most beneficial way. An oil company might wish to know where to place its wells in order to maximize its expected profit.
- ❖ A political candidate might want to determine where to spend money buying campaign advertising in order to maximize the chances of winning an election.
- ❖ An airline might wish to assign crews to flights in the least expensive way possible, making sure that each flight is covered and that government regulations regarding crew scheduling are met.
- ❖ An internet service provider might wish to determine where to place additional resources in order to serve its customers more effectively. **All of these are examples of problems that can be solved by modeling them as linear programming – Resource Allocation**



- ❖ You have a road map on which the distance between each pair of adjacent intersections is marked, and you wish to determine the shortest route from one intersection to another. The number of possible routes can be huge, even if you disallow routes that cross over themselves. How can you choose which of all possible routes is the shortest?
- ❖ A doctor needs to determine whether an image represents a cancerous tumor or a benign one. The doctor has available images of many other tumors, some of which are known to be cancerous and some of which are known to be benign. A cancerous tumor is likely to be more similar to other cancerous tumors than to benign tumors, and a benign tumor is more likely to be similar to other benign tumors.



# Lifecycle Steps

- 1) Problem Statement
  - 2) Requirements/Conditions  
Constraints
  - 3) Design/Logic
  - 4) develop Algorithm  
<empvers>
  - 5) (Validation) Correctness
  - 6) Analysis
  - 7) Implementation
  - 8) Testing & Debugging
- 



# Need for Analysis

Distributed Computing

Problem

Mobile Computing

Perf. Metrics

Architecture +  
Comp. Environment

T+S

Alg<sub>1</sub>

Alg<sub>2</sub>

Alg<sub>3</sub>

Alg<sub>4</sub>

Who is  
Best?

①

Performance Comparison

②

Performance Metrics : (Time + Space)  
(Resource Consumption) (Resources)

(Efficient)



What? < Time & Space >

Why? < Perfo. Comparison >

↓  
(efficient Sol'n)

How? < Methodology >



## Analysis of Algorithms - Need

Analyzing algorithms involves thinking about how their resource requirements-the amount of time and space they use-will scale with increasing input size.

**Proposed Definition of Efficiency (1):** An algorithm is efficient if, when implemented, it runs quickly on real input instances.

**Proposed Definition of Efficiency (2):** An algorithm is efficient if it achieves qualitatively better worst-case performance, at an analytical level, than brute-force search.

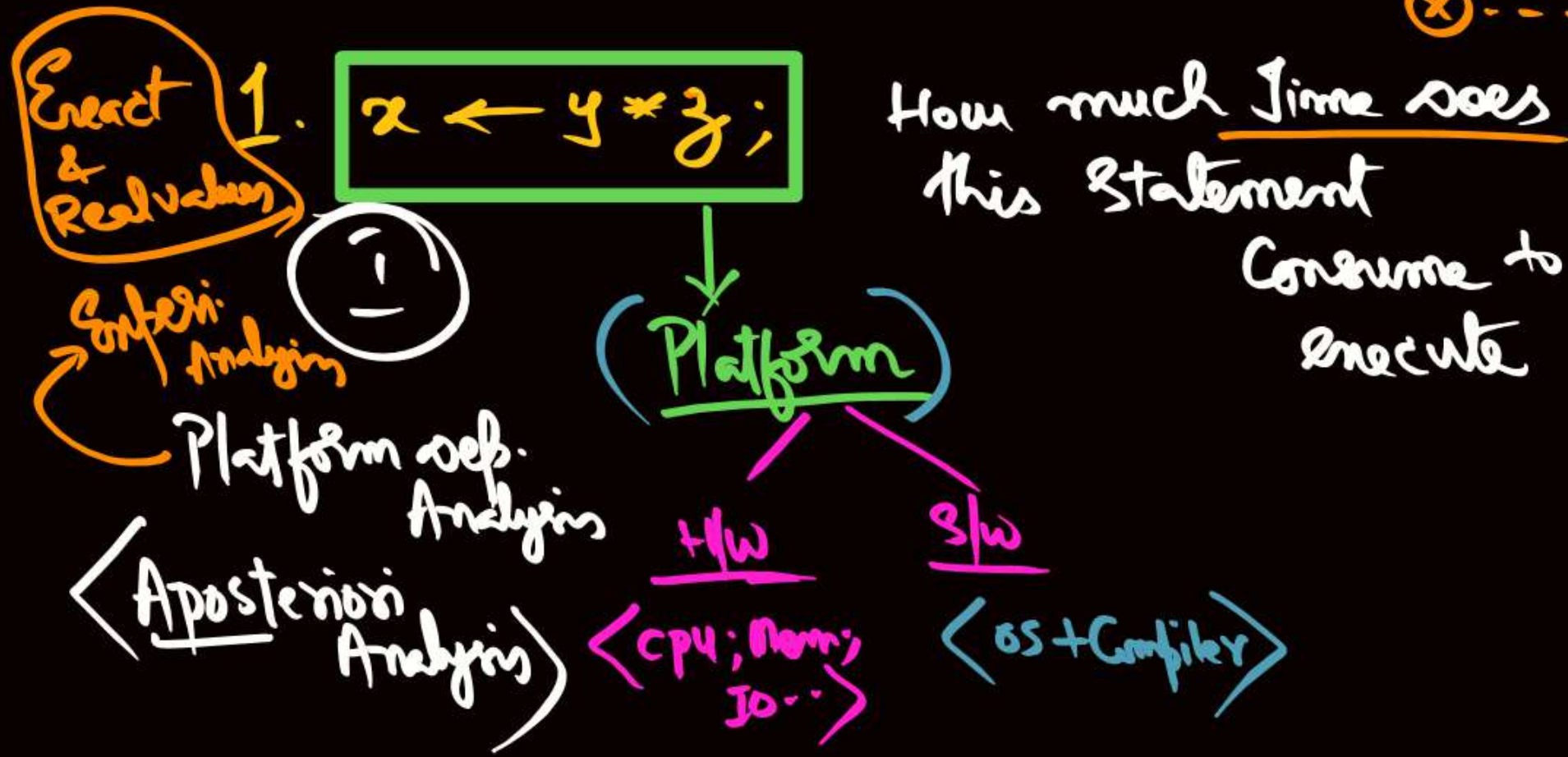
**Proposed Definition of Efficiency (3):** An algorithm is efficient if it has a polynomial running time.

Time



## How to Analyze

## Methodology



## Limitations of Aposteriori Analysis

- 1) Difficult
- 2) Non-uniformity of Platform makes Perf. Comparison difficult;





# Thank You



[t.me/KhaleelSirGate](https://t.me/KhaleelSirGate)

