CS & IT ENGINEERING

Algorithm

Analysis of Algorithms

Lecture No.- 01

By- Aditya sir

Topics to be Covered

Topic Schedule

Topic Outcomes

Topic Intro to Algorithms

A → 1 or 2 → 9%

B → 3rd → 27%

C → 4th yr → 29%

D → Job → 16%

E → preparing 2nd + time

↳ 19%

Topic : Lecture Schedule

DAA

↓

Design & Analysis of Algorithms

1. Analysis of Algorithms

1. Algorithm Concept and Lifecycle
2. Analysis of Algorithms
3. Methodology & Types of Analysis
4. ☆☆ Asymptotic Notations
5. Framework for Analysing Recursive Algorithms
6. Apriori analysis of Non-Recursive Algorithms
7. ☆☆ ← Analysing Loops
8. Space Complexity
9. Mathematical Background

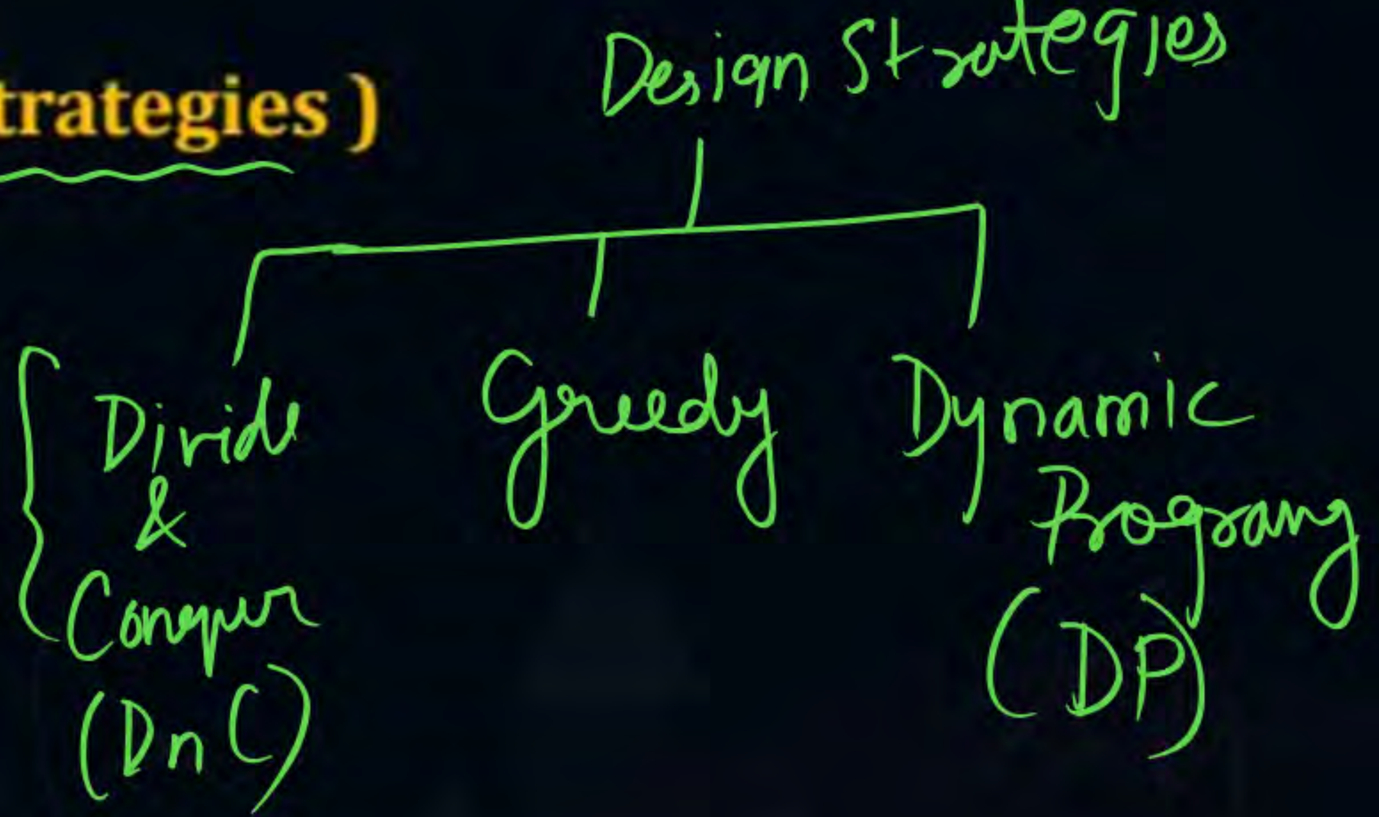
Topic : Lecture Schedule

2. Divide & Conquer (Design Strategies)

1. General Method (Framework)
2. Max-Min Problem
3. Merge Sort
4. Binary Search
5. Quick Sort
6. Matrix Multiplication
7. Long Integer Multiplication (LIM)
8. Master Method for D and C Recurrences
9. Recursion Tree

Sorting: 3, 5

Time Complexity Analysis: 7, 8, 9



Topic : Lecture Schedule

3. Greedy Method

1. General Method (Framework)
2. Knapsack Problem
3. Job Sequencing with Deadlines [JSWD]
4. Optimal Merge Patterns [OMP]

   4.1 Huffman Coding

5. ☆☆☆ Minimum Cost Spanning Trees
   1. Prims Method
   2. Kruskal's Method → Set + Heap
6. Dijkstra's Shortest Paths Problem ← Shortest Path Algo.

Topic : Lecture Schedule

4. Dynamic Programming (DP) → [Interviews + GATE + Job]

1. The Framework

Applications ? ← 2. Difference between DP, Greedy Method and Divide and Conquer

3. Multistage Graphs

4. Travelling Salesperson Problem [TSP]

☆ 5. Binary Knapsack Problem (0/1 knapsack)

6. All Pairs Shortest Paths (APSP)

[mcm]

7. Bellman-Ford Single Source Shortest Paths (SSSP)

8. Longest Common Subsequence [LCS]

10. Matrix Chain Multiplication sum of Subsets → (SOS)

Bonus ← 11. Optimal Cost Binary Search Tree

Topic : Lecture Schedule

5. Graph Algorithms

1. Representation of Graphs { Directed, Undirected, properties }
2. Graph Traversals

Applications

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 Sorting

BFS
- 5.2.5 FIFO BFS
- 5.2.6 LIFO BFS

Topic : Lecture Schedule

6. Heap Algorithms

(Data Structure) 1. Operations : Create, Insert, Delete, Modify

2. Applications : Heapsort

Time Complexity Analysis

+

Algo

Sorting Technique

Topic : Lecture Schedule

7. **Sets**

1. Representations
2. Operations → (Union & Find)

Applications

1) Kruskal Algo for MCST

Topic : Lecture Schedule

8. **Sorting Algorithms**
    1. Basic terminology →
    2. Methods
        1. Bubble Sort
        2. Selection Sort
        3. Insertion Sort
        4. Radix Sort

Topic : Lecture Schedule

(Bonus) 9. Backtracking & Branch- Bound

↳ Application

N- Queens Problem

Reference Books :

1) Introduction to Algorithms → Cormen

2) Fundamentals of Algorithms → Sahni

Scope / outcomes :

- → Semester exams (College)
- → GATE, TIFR, ISRO, BARC ......
- → Placements → Product / Service based
  - Product → Google, Microsoft
  - Service → Infosys, WIPRO, Accenture
- → Coding Tests

Algorithms

30th Jan ✓
↓
Amazon coding Test

3rd Feb
↓
GATE ✓

5th Feb
↓
Amazon Interview ✓

Pre-requisites:

① Data Structures Fundamentals

↳ [Stacks, Queue, Tree - - -]

② Programming Fundamentals

↳ if-else

↳ loops

↳

③ Basic Maths

↳ Series (AP, GP...)

PnC, Log,

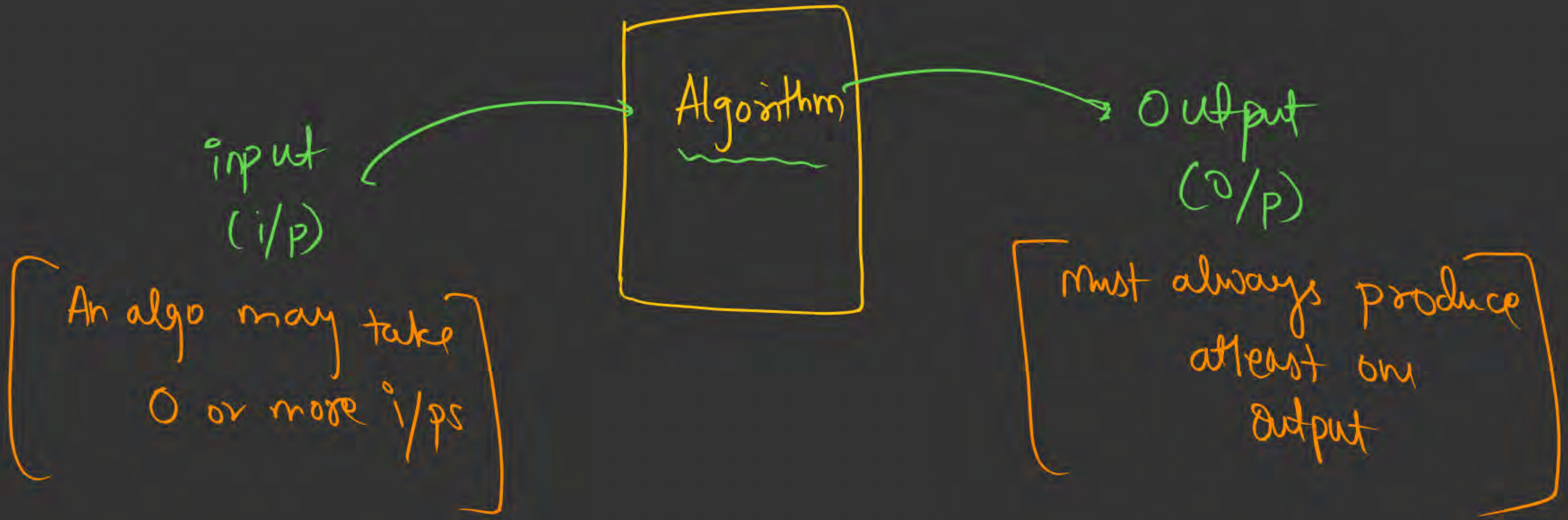
Algorithm:- An algorithm is a collection of finite number of instructions to Solve a given problem.

↓ statements

→ These instructions are fundamental and should follow a proper sequence.

$x = y + z$

↓ fundamental opr



An algo may take 0 or more i/ps

must always produce atleast one output

```
Python :- for i in range(0,n+1)

C++ :     For(i=0; i<=n; i++)
```

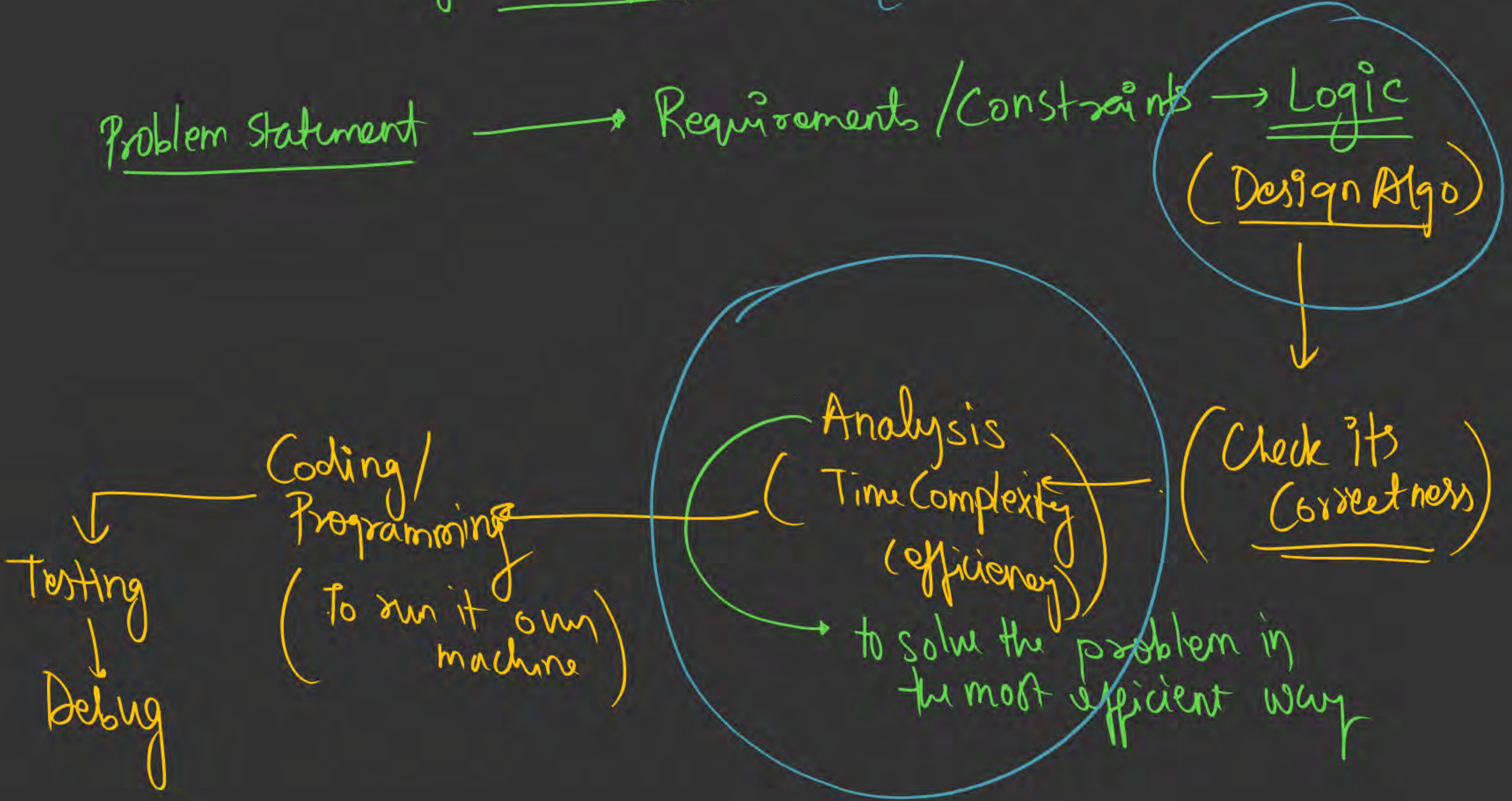
Program → Algorithm implemented using some programming language

Algorithm → Pseudo Code → for i → 0 : n
for i = 0 to n } For i :(0 → n)

(set of sequential Rules/statements/instructions)

Algorithm Lifecycle

[Gate algo Syllabus]



Problem Statement → Requirements / Constraints → Logic

(Design Algo)

(Check its Correctness)

Analysis

(Time Complexity

(efficiency))

to solve the problem in the most efficient way

Coding / Programming

(To run it own machine)

Testing

Debug

0 inputs

```
Algo AJSir( )
{
    print("Hello Students!")
}
```

```
Algo AJSir2( )
{
    return 100
}
```

★ Analysis of Algorithm :

✓ 1) Why to analyse? → Compare two Algos

✓ 2) What to analyse? → on the basis of Time / Space { Resource consumption }

★★ 3) How to analyse?

(Q.1) Why to analyse?

↳ To Compare and decide which solution is the best among different available solutions (Algos)

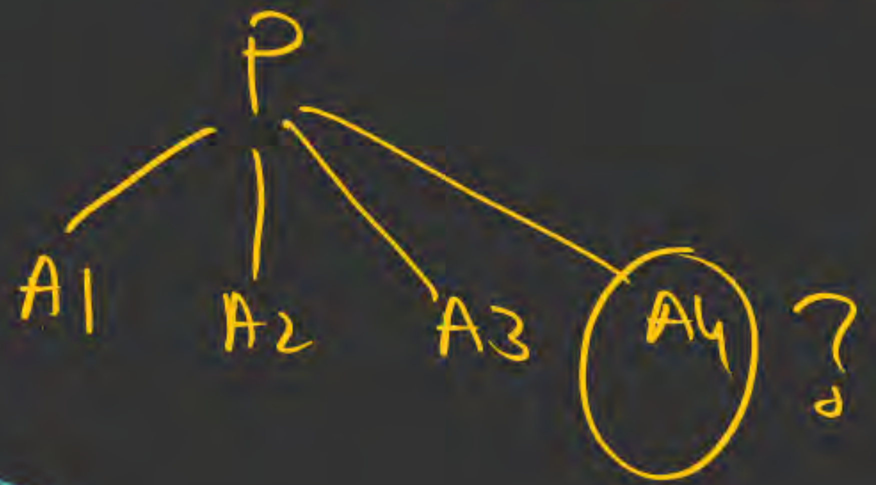
Problem:- Nagpur → Delhi (P)

→ 1) Walk → Soln (A1)

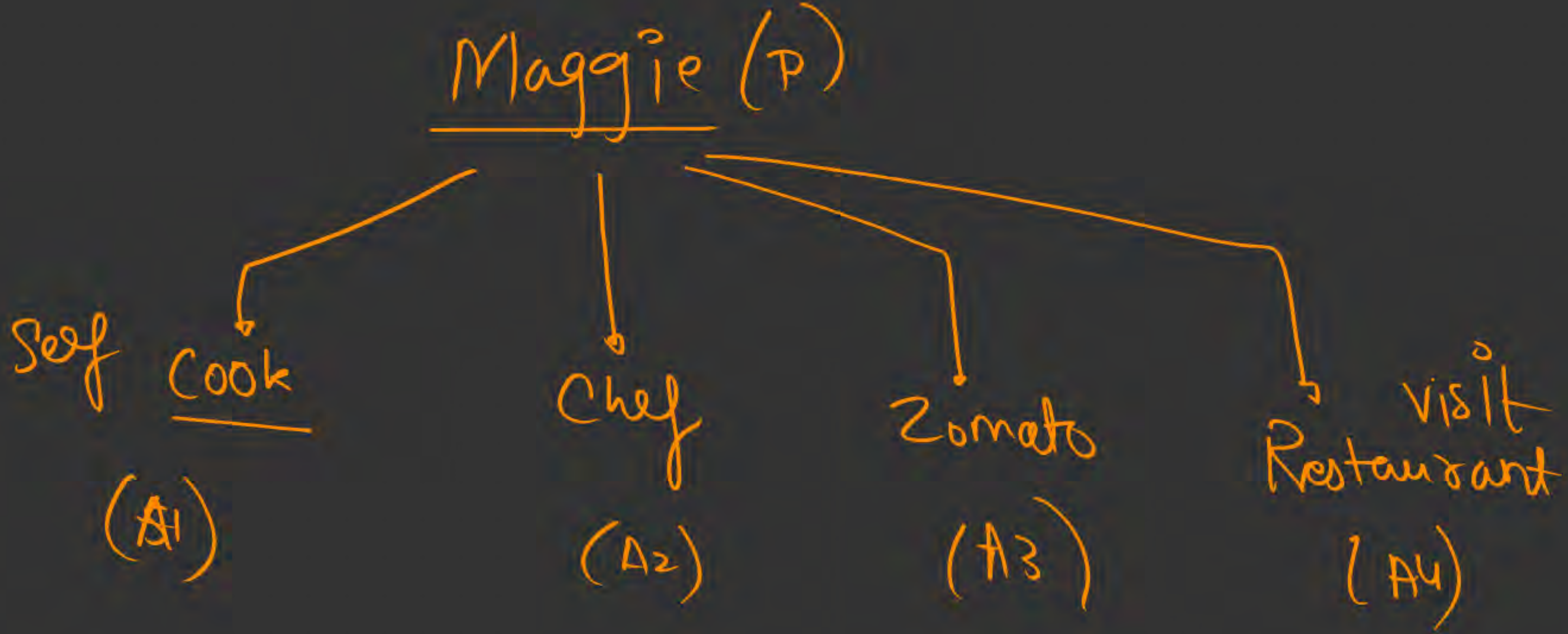
2) Bike/Car → A2

3) Train → A3

4) Flight → A4



Time ↓ cost ↑ (money)


Maggie (P)
Self Cook
(A1)
Chef
(A2)
Zomato
(A3)
visit Restaurant
(A4)

(Q.2) What to analyse ?

↳ Analyse the consumption of Resources.

Resources: Time, Space/Memory, → GATE

Money, Internet Bandwidth, etc.

No of programmers.

(9.3) <u>How to analyse?</u> (Framework/ methods)

1. Aposteriori analysis

   Analyzing After implementation of Algo.

2. Apriori Analysis

   Analyzing <u>Before</u> implementation of Algo.

① Apostoriori Analysis

a=b+c

Advantages

→ gives the exact measurement of time units.

High level lang → User/Programmer friendly

Low level lang → machine friendly

Disadvantages

→ Platform dependent

- ↳ Software → OS (Linux, Windows, MAC)
- ↳ Hardware → (Processor/CPU, memory (RAM))

→ Programming language dependent

2 mins Summary

Topic Schedule ✓

Topic Books ✓

Topic Prerequisite ✓

Topic Outcomes ✓

Intro to Algo

THANK - YOU