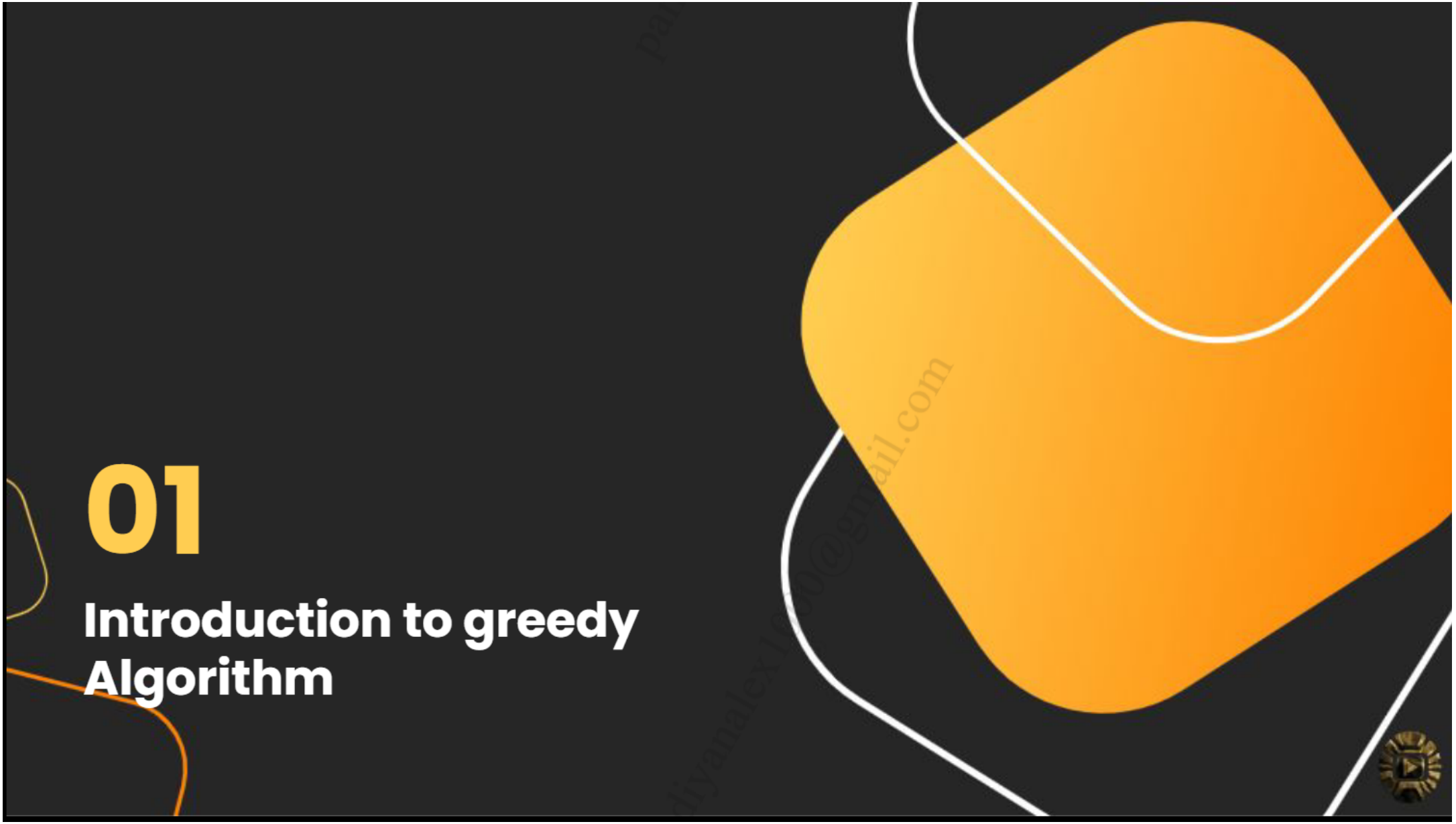
Session – 15

Greedy Algorithm

CONTENTS

01

Introduction to greedy Algorithm



Introduction to Greedy Algorithm

Definition:

- A **Greedy Algorithm** solves problems by consistently selecting the **best immediate option** available at each step.
- As the name implies, it makes decisions that are **locally optimal**, with the hope that these choices will eventually lead to a **globally optimal solution**.

How Does it Work?

- At every decision point, the algorithm evaluates the situation using an **objective function** which it aims to either **maximize or minimize** and selects the option that provides the most favorable outcome at that moment.
- Greedy algorithms **do not backtrack** or revisit past choices. They make a single pass through the problem space, committing to the current best decision without reconsideration.

02

When and How?

When and How?

When to Apply:

- Greedy algorithms are best suited for problems where **making locally optimal choices at each step leads to a globally optimal solution**.
- These problems typically exhibit:
  - **Greedy Choice Property**
  - **Optimal Substructure**

How to Apply:

- Begin with the **smallest possible sub-solutions** and build the final solution incrementally
- At each step, make the **best local (greedy) choice** available
- Maintain additional state if necessary (e.g., track max product so far, count of negative numbers, etc.)

03

Characteristics

Characteristics of Greedy Algorithm

- Makes a **locally optimal** choice at each step
- Doesn't reconsider past decisions
- **No backtracking**
- **Fast & efficient**, often used for optimization problems
- Works only when **greedy choice property** and **optimal substructure** hold

04

Greedy Vs Dynamic

Greedy Algorithm Vs Dynamic Programming

Feature	Greedy	Dynamic Programming
Decision-making	Local optimal	Global optimal
Overlapping Subproblems	❌ No	✅ Yes
Optimal Substructure	✅ Required	✅ Required
Backtracking	❌ No	✅ Sometimes
Time Efficiency	Faster (usually O(NlogN))	Slower (usually $O(N^2)$ or more)

05

Advantages & Disadvantages

Advantages and Disadvantages

Advantages:

- Simple to implement
- Fast and efficient
- Good for real-time systems

Disadvantages:

- Doesn't always give optimal solution
- Only works for specific types of problems
- Requires proof of correctness

Real -Life Applications

- File compression (Huffman coding)
- Job Scheduling
- Network routing (shortest path)
- Resource allocation
- Cache optimization

06

Activity

Activity Graph

- Largest Number
- Gas Station
- Merge Intervals
- Meeting Rooms