# LAB PROJECT REPORT

## Group Members with Roll number and Section:

Sarvesh Khawale (A5-B1-01)  
Tarun Vyas (A5-B1-16)   
Devyansh faye (A5-B3-40)

## TITLE:

Optimal Resource Allocation System Using Greedy Method

## Objectives:

* To schedule multiple jobs to maximize total profit using deadlines.
* To optimally select resources under cost constraints using a greedy approach.
* To compare greedy results with brute-force methods for validation.
* To analyze time complexity and demonstrate efficiency.

## Introduction:

In real-world systems, resources such as time, money, or machines must be allocated efficiently to achieve maximum output. The Greedy Method is a powerful algorithmic strategy that makes locally optimal choices at each step to achieve a globally optimal result.  
  
This project implements an Optimal Resource Allocation System that uses greedy algorithms for two major problems:  
1. Job Scheduling with Deadlines and Profits – selecting the most profitable jobs to execute within limited time slots.  
2. Resource Selection under Constraints – choosing resources with the highest value-to-cost ratio, similar to the fractional knapsack problem.  
  
A brute-force approach is also implemented for both modules to verify the correctness and optimality of the greedy solution.

## Algorithms/Technique used:

1. Job Scheduling Algorithm (Greedy):  
 Step 1: Sort all jobs in descending order of profit.  
 Step 2: For each job, assign it to the latest available slot before its deadline.  
 Step 3: If no slot is available, skip the job.  
 Step 4: Continue until all jobs are scheduled or all slots are filled.  
  
 Explanation: Ensures the most profitable jobs are scheduled first without violating deadlines.  
 Diagram (Conceptual): Jobs → Sort by Profit → Assign to Slots (Before Deadline)  
  
2. Resource Selection Algorithm (Greedy – Knapsack Type):  
 Step 1: Compute value-to-cost ratio for each resource.  
 Step 2: Sort resources in descending order of this ratio.  
 Step 3: Pick resources until the total cost reaches the capacity.  
  
 Explanation: Prioritizes resources providing the highest return per unit cost.

## Time Complexity and its explanation:

Job Scheduling (Greedy): O(n log n) – due to sorting by profit.  
Job Scheduling (Brute Force): O(2ⁿ) – tests all combinations.  
Resource Selection (Greedy): O(n log n) – sorting by ratio dominates runtime.  
Resource Selection (Brute Force): O(2ⁿ) – tests all subsets.  
  
Greedy algorithms provide results close to optimal while significantly reducing computation time.

## Results:

Job Scheduling Output:  
Job Order (Greedy): ['C', 'A', 'E']  
Total Profit: 142  
  
Job Order (Brute Force): ['A', 'C', 'E']  
Total Profit: 142  
  
Resource Selection Output:  
Selected Resources (Greedy): [4, 3, 0]  
Total Value: 660  
  
Selected Resources (Brute Force): (4, 2)  
Total Value: 660  
  
Discussion: Both methods yielded the same total value, demonstrating that the Greedy algorithm achieved an optimal solution with less computational effort. The efficiency ratio between Greedy and Brute Force was 100%, confirming correctness.

## Conclusion and future scope:

The Optimal Resource Allocation System successfully demonstrates how the Greedy Method can be applied to scheduling and selection problems for faster and efficient decision-making.  
  
Future Scope:  
- Integrate with real-time systems for task scheduling (e.g., CPU scheduling, cloud resource allocation).  
- Extend to dynamic or multi-constraint optimization problems.  
- Build a graphical interface for better visualization of scheduling and selection processes.