Task 3: Job Sequencing Problem
Define a class Job with properties int Id, int Deadline, and int Profit. Then implement a function List<Job> JobSequencing(List<Job> jobs) that takes a list of jobs and returns the maximum profit sequence of jobs that can be done before the deadlines. Use the greedy method to solve this problem.
ANS:

```
package com.Day19;
import java.util.*;
class Job {
    int Id;
    int Deadline;
    int Profit;
    Job(int id, int deadline, int profit) {
        this.Id = id;
        this.Deadline = deadline;
        this.Profit = profit;
    }
    @Override
    public String toString() {
        return "Job{" +
                "Id=" + Id +
                ", Deadline=" + Deadline +
                ", Profit=" + Profit +
                '}';
    }
}
public class JobSequencingProblem {
    public static List<Job> JobSequencing(List<Job> jobs) {
        // Sort jobs by profit in descending order
        jobs.sort((a, b) -> b.Profit - a.Profit);
        // Find the maximum deadline
        int maxDeadline = jobs.stream().mapToInt(job ->
job.Deadline).max().orElse(0);
        // Initialize the result array to store the sequence of jobs
        Job[] result = new Job[maxDeadline];
        boolean[] slot = new boolean[maxDeadline];
        // Iterate through all jobs to schedule them
        for (Job job : jobs) {
```

```
            // Find a slot for the job, starting from the last possible slot
            for (int j = Math.min(maxDeadline - 1, job.Deadline - 1); j >= 0;
j--) {
                if (!slot[j]) {
                    slot[j] = true;
                    result[j] = job;
                    break;
                }
            }
        }
        // Collect the jobs that have been scheduled
        List<Job> scheduledJobs = new ArrayList<>();
        for (Job job : result) {
            if (job != null) {
                scheduledJobs.add(job);
            }
        }
        return scheduledJobs;
    }
    public static void main(String[] args) {
        List<Job> jobs = Arrays.asList(
                new Job(1, 4, 20),
                new Job(2, 1, 10),
                new Job(3, 1, 40),
                new Job(4, 1, 30)
        );
        List<Job> jobSequence = JobSequencing(jobs);
        System.out.println("The maximum profit sequence of jobs is:");
        for (Job job : jobSequence) {
            System.out.println(job);
        }
    }
}
```

OUTPUT:

```
The maximum profit sequence of jobs is:
Job{Id=3, Deadline=1, Profit=40}
Job{Id=1, Deadline=4, Profit=20}
```