



BHARATIYA VIDYA BHAVAN'S  
**SARDAR PATEL INSTITUTE OF TECHNOLOGY**  
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.  
(Autonomous College Affiliated to University of Mumbai)  
**MASTER OF COMPUTER APPLICATIONS**

**Class : F.Y.MCA Semester : II Academic Year : 2024-25**

**Course Name : Design and Analysis of Algorithm MC507**

**Subject Incharge : Prof.Nikhita Mangaonkar**

**UCID: 2024510001 BATCH: A NAME: Atharva Vasant Angre**

**EXPERIMENT NO: 03**

**EXPERIMENT TITLE:** To implement Greedy Algorithm.

3.1 To Implement Greedy Algorithm (Job Sequence).

**Objective:**

1.To Implement Greedy Algorithm (Job Sequence).



**BHARATIYA VIDYA BHAVAN'S**  
**SARDAR PATEL INSTITUTE OF TECHNOLOGY**  
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.  
(Autonomous College Affiliated to University of Mumbai)  
**MASTER OF COMPUTER APPLICATIONS**

**Class : F.Y.MCA Semester : II Academic Year : 2024-25**

**Course Name : Design and Analysis of Algorithm MC507**

**Subject Incharge : Prof.Nikhita Mangaonkar**

**Program code: -**

**Job Sequence**

```
import java.util.Arrays;

public class JobSequence {
    public static void main(String[] args) {
        Job[] jobs = {
            new Job(1, 10, 10, 10),
            new Job(2, 10, 10, 10),
            new Job(3, 10, 10, 10),
            new Job(4, 10, 10, 10)
        };

        int n = jobs.length;
        jobSequence(jobs, n);

        static void jobSequence(Job[] jobs, int n) {
            Arrays.sort(jobs, (Job a, Job b) -> a.profit > b.profit);

            int maxDeadline = 0;
            for (Job job : jobs) {
                maxDeadline = Math.max(maxDeadline, job.deadline);
            }

            int[] result = new int[maxDeadline + 1];
            boolean[] slot = new boolean[maxDeadline + 1];

            Arrays.fill(slot, false);

            int maxProfit = 0;
            int count = 0;
        }
    }
}
```

```
public class JobSequence {
    public static void main(String[] args) {
        Job[] jobs = {
            new Job(1, 10, 10, 10),
            new Job(2, 10, 10, 10),
            new Job(3, 10, 10, 10),
            new Job(4, 10, 10, 10)
        };

        int n = jobs.length;
        jobSequence(jobs, n);

        static void jobSequence(Job[] jobs, int n) {
            Arrays.sort(jobs, (Job a, Job b) -> a.profit > b.profit);

            int maxDeadline = 0;
            for (Job job : jobs) {
                maxDeadline = Math.max(maxDeadline, job.deadline);
            }

            int[] result = new int[maxDeadline + 1];
            boolean[] slot = new boolean[maxDeadline + 1];

            Arrays.fill(slot, false);

            int maxProfit = 0;
            int count = 0;

            for (int i = 1; i <= maxDeadline; i++) {
                if (slot[i]) {
                    result[i] = job.profit;
                    slot[i] = true;
                    maxProfit += job.profit;
                    jobIndex++;
                    break;
                }
            }

            System.out.println("Selected jobs: ");
            for (int i = 1; i <= maxDeadline; i++) {
                if (slot[i]) {
                    System.out.print(jobIndex + " ");
                    jobIndex++;
                }
            }

            System.out.println("\nTotal Profit: " + maxProfit);
        }
    }
}
```

```
public class JobSequence {
    public static void main(String[] args) {
        Job[] jobs = {
            new Job(1, 10, 10, 10),
            new Job(2, 10, 10, 10),
            new Job(3, 10, 10, 10),
            new Job(4, 10, 10, 10)
        };

        int n = jobs.length;
        jobSequence(jobs, n);

        static void jobSequence(Job[] jobs, int n) {
            Arrays.sort(jobs, (Job a, Job b) -> a.profit > b.profit);

            int maxDeadline = 0;
            for (Job job : jobs) {
                maxDeadline = Math.max(maxDeadline, job.deadline);
            }

            int[] result = new int[maxDeadline + 1];
            boolean[] slot = new boolean[maxDeadline + 1];

            Arrays.fill(slot, false);

            int maxProfit = 0;
            int count = 0;

            for (int i = 1; i <= maxDeadline; i++) {
                if (slot[i]) {
                    result[i] = job.profit;
                    slot[i] = true;
                    maxProfit += job.profit;
                    jobIndex++;
                    break;
                }
            }

            System.out.println("Selected jobs: ");
            for (int i = 1; i <= maxDeadline; i++) {
                if (slot[i]) {
                    System.out.print(jobIndex + " ");
                    jobIndex++;
                }
            }

            System.out.println("\nTotal Profit: " + maxProfit);
        }
    }
}
```



**BHARATIYA VIDYA BHAVAN'S**  
**SARDAR PATEL INSTITUTE OF TECHNOLOGY**  
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.  
(Autonomous College Affiliated to University of Mumbai)  
**MASTER OF COMPUTER APPLICATIONS**

**Class : F.Y.MCA Semester : II Academic Year : 2024-25**

**Course Name : Design and Analysis of Algorithm MC507**

**Subject Incharge : Prof.Nikhita Mangaonkar**

**Output:**

```
Project -> JobSequencing
+ JobSequencing.java
+ JobSequencingTest.java
+ JobSequencingTest.java

Run -> JobSequencingTest

Output:
C:\Program Files\Java\jdk-17\bin\java.exe -classpath C:\Program
SequencingTest
Job 1 Job 2
Total Profit: 300
Process finished with exit code 0
```

**Conclusion:**

The Greedy Algorithm for Job Sequencing helps in selecting jobs to maximize profit while meeting deadlines. It works by choosing the most profitable jobs first, ensuring an efficient schedule. This method is fast and useful for real-world scheduling problems. However, it may not always give the best solution in complex cases. Overall, it is a simple and effective approach for job sequencing.