Name : Manthan Gopal Dhole
Id : 201080020
Branch : IT
Date :05-03-2022

DAA Assignment 6

Aim :

We have a set of jobs with the amount of time they need to complete, $j_1, j_2, \ldots, j_N$, the deadline they need to be completed by, $d_1, d_2, \ldots, d_N$, and a profit incurred if the job is completed by the deadline, $p_1, p_2, \ldots, p_N$.
The optimization problem attempts to order the jobs so as to get the ***maximum profit***. The decision problem is to find a schedule that gives maximum profit.

Jobs	J1	J2	J3	J4	J5	J6
Deadlines	5	3	3	2	4	2
Profits	200	180	190	300	120	100

- Design a greedy method for job scheduling problems
- Write complexity analysis of the proposed algorithm
- Write a program for the same

Theory :

Given an array of jobs where every job has a deadline and associated profit if the job is finished before the deadline. It is also given that every job takes a single unit of time, so the minimum possible deadline for any job is 1. How to maximize total profit if only one job can be scheduled at a time.

A Simple Solution is to generate all subsets of a given set of jobs and check individual subsets for the feasibility of jobs in that subset. Keep track of maximum profit among all feasible subsets. The time complexity of this solution is exponential.

Greedy Approach :

So , We need to define HashMap===> To connect profit to its deadline ,PriorityQueue ===> To get max profit element first out and an Array ===> to store the attendance of the job.

1]Here we can use priority queue to store the profits of the jobs.To get max profit out first from the queue while removing elements from the queue one by one. So, for these operations we need O(n) time complexity and O(n) space complexity.

2]Now, we can link the profit of Job j^th to its deadline through HashMap technique. For all these we need O(n) time and O(n) space complexity.
3]Now, we can get the max profit , first out from queue , then through max profit we can access it's deadline using HashMap and if there is a vacancy we will mark J^th job attendance in getting max profit. For the task up to marking the attendance of the job we need O(n^2)

In short :

1) Sort all jobs in decreasing order of profit.

2) Iterate on jobs in decreasing order of profit.For each job , do the following :

a)Find a time slot i, such that slot is empty and i < deadline and i is greatest.Put the job in

this slot and mark this slot filled.

b)If no such i exists, then ignore the job.

Time complexity And Analysis:

We need to get the max element from the queue and then we need to mark it's attendance if possible on the Done[] array for these we need to iterate the array.
So, there is a nested loop Time complexity will be O(n^2).

Let n be the max deadline in the given jobs.

First 1 , 2, 3, 4 ....n operations to get values from the queue and for each operation we have to check nearly in this pattern n, n-1, n-2 ..... 1 .

So, Total operations nearly Operations = 1(n-0) + 2(n-1) + 3(n-2)..... n(n-(n-1))
So, the complexity will be coming O(n^2).

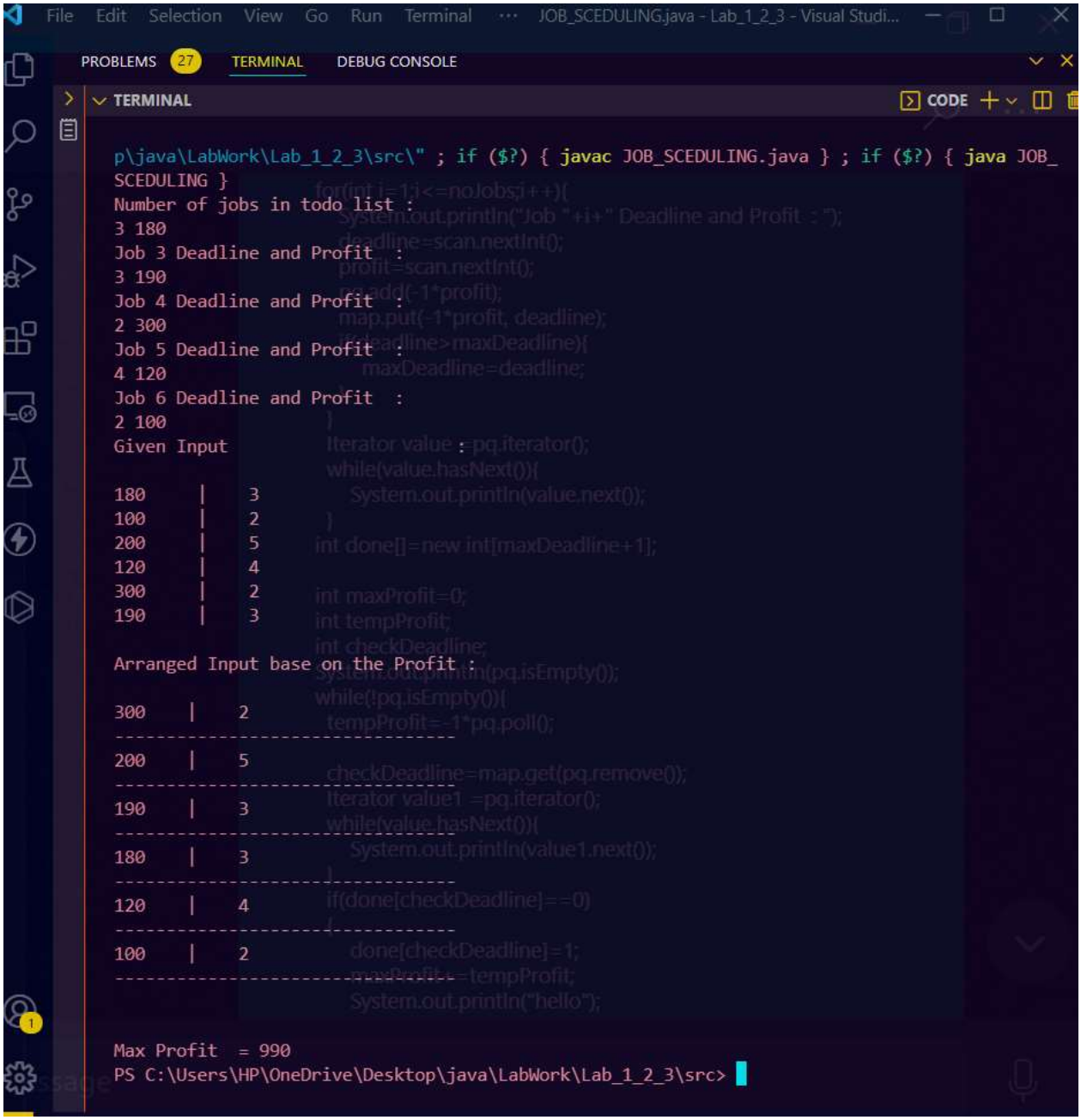
Code :

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.PriorityQueue;
import java.util.Scanner;

public class JOB_SCEDULING {
//ID 201080020 Manthan Dhole
    public static void main(String[] args) {
        PriorityQueue<Integer>pq=new PriorityQueue<Integer>();
        HashMap<Integer, Integer>map=new HashMap<Integer, Integer>();
        //Defining the hashmap to connect the profits of the jobs to its
deadline

          System.out.println("Number of jobs in todo list :  ");
          Scanner scan =new Scanner(System.in);
          int noJobs=scan.nextInt();// Number of jobs

          int deadline=0;
          int profit=0;
          int maxDeadline=0;

          for(int i=1;i<=noJobs;i++){
              System.out.println("Job "+i+" Deadline and Profit  : ");
              deadline=scan.nextInt();
```

```
            profit=scan.nextInt();
            pq.add(-1*profit);// PriorityQueue in java are giving small
first
            map.put(-1*profit, deadline);
            if(deadline>maxDeadline){
                maxDeadline=deadline;
            }
          }
          System.out.println("Given Input                          :\n");
          map.forEach((key,value)->
          System.out.println(-1*key + "      |     "+value)
          );

          System.out.println("\nArranged Input base on the Profit :
\n");
          Iterator pqueue= pq.iterator();
          while(pqueue.hasNext())
          {
            int temp=(int) pqueue.next();
            System.out.println(-1*temp+"     |     "+map.get(temp));
            System.out.println("-----------------------------------");
          }
        int done[]=new int[maxDeadline+1];
          done[0]=1;
        int maxProfit=0;
        int tempProfit;
        int checkDeadline;

        while(!pq.isEmpty()){
          // This loop will run till the queue is not empty
          tempProfit=-1*pq.peek();

          checkDeadline=map.get(pq.remove());
          if(done[checkDeadline]==0)
          {
              done[checkDeadline]=1;
              maxProfit+=tempProfit;
          }
          else{
              int i=checkDeadline-1;
```

```
                while(i>=0)
                {
                    if(done[i]==0){
                        done[i]=1;
                        maxProfit+=tempProfit;
                        break;
                    }
                   i--;
                }
              }
           }
           System.out.println("\n\nMax Profit  = "+maxProfit);




    }
}
```

OUTPUT :


File Edit Selection View Go Run Terminal ··· JOB_SCEDULING.java - Lab_1_2_3 - Visual Studi...
PROBLEMS 27
TERMINAL
DEBUG CONSOLE
TERMINAL
CODE
p\java\LabWork\Lab_1_2_3\src\" ; if ($?) { javac JOB_SCEDULING.java } ; if ($?) { java JOB_
SCEDULING }
Number of jobs in todo list :
3 180
Job 3 Deadline and Profit :
3 190
Job 4 Deadline and Profit :
2 300
Job 5 Deadline and Profit :
4 120
Job 6 Deadline and Profit :
2 100
Given Input
180 | 3
100 | 2
200 | 5
120 | 4
300 | 2
190 | 3
Arranged Input base on the Profit :
300 | 2
200 | 5
190 | 3
180 | 3
120 | 4
100 | 2
Max Profit = 990
PS C:\Users\HP\OneDrive\Desktop\java\LabWork\Lab_1_2_3\src>

Conclusion :

1] Here we implemented the job scheduling problem using a greedy algorithm.
2] Here we use the priority Queue data structure to store the data in the form of max heap. To get the Max profit job first out.
3]Also we use hashmap to connect the profit to its deadline and to get the deadline from the hashmap .
4] In this way by applying the proper algo we got the max profit output.