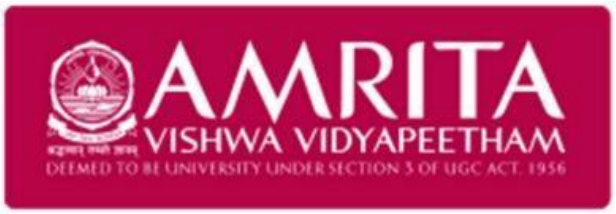

SCHOOL OF COMPUTING

DESIGN AND ANALYSIS OF ALGORITHMS

LAB WORKBOOK

WEEK - 7

NAME : Reddy Manish

ROLL NUMBER : CH.SC.U4CSE24129

SECTION : CSE-B

Question 1: To implement the Greedy algorithm for the Job Sequencing with Deadlines problem and determine the optimal sequence of jobs that maximizes total profit.

Consider a set of 14 jobs, where each job requires one unit of time for completion. Each job has an associated profit and deadline. A job must be completed on or before its deadline to earn the profit.

The profits of the jobs are:

{22, 19, 29, 28, 30, 21, 27, 25, 24, 26, 14, 27, 19, 11}

The deadlines of the jobs are:

{3, 3, 8, 6, 7, 5, 10, 4, 6, 12, 13, 2, 14, 1}

CODE:

```
//CH.SC.U4CSE24129
#include <stdio.h>
#include <stdlib.h>

struct Job{
   int id;
   int p;
   int d;
};

void sort(struct Job j[], int n){
   int i, k;
   struct Job temp;

   for(i = 0; i < n-1; i++){
      for(k = 0; k < n-i-1; k++){
         if(j[k].p < j[k+1].p){
            temp = j[k];
            j[k] = j[k+1];
            j[k+1] = temp;
         }
      }
   }
}

int main(){
   int n = 0;
   int i, k;
   int max = 0;
   int totalProfit = 0;
```

```
    printf("Enter no of Jobs:\n");
    scanf("%d", &n);

    struct Job *j = (struct Job*) malloc(n * sizeof(struct Job));

    for (i = 0; i < n; i++){
        printf("Enter job %d profit and deadline:\n", i+1);
        j[i].id = i+1;
        scanf("%d %d", &j[i].p, &j[i].d);
    }

    sort(j, n);

    for(i = 0; i < n; i++){
        if(j[i].d > max)
            max = j[i].d;
    }

    int *slot = (int*) malloc((max+1) * sizeof(int));

    for(i = 0; i <= max; i++)
        slot[i] = -1;

    for(i = 0; i < n; i++){
        for(k = j[i].d; k > 0; k--){
            if(slot[k] == -1){
                slot[k] = j[i].id;
                totalProfit += j[i].p;
                break;
            }
        }
    }
    printf("\nSelected Jobs:\n");
    for(i = 1; i <= max; i++){
        if(slot[i] != -1)
            printf("Job %d\n", slot[i]);
        else
            printf("-\n");
    }
    printf("Total Profit = %d\n", totalProfit);
    free(j);
    free(slot);

    return 0;
}
```

OUTPUT:

```
Enter no of Jobs:
14
Enter job 1 profit and deadline:
22 3
Enter job 2 profit and deadline:
19 3
Enter job 3 profit and deadline:
29 8
Enter job 4 profit and deadline:
28 6
Enter job 5 profit and deadline:
30 7
Enter job 6 profit and deadline:
21 5
Enter job 7 profit and deadline:
27 10
Enter job 8 profit and deadline:
25 4
Enter job 9 profit and deadline:
24 6
Enter job 10 profit and deadline:
26 12
Enter job 11 profit and deadline:
14 13
Enter job 12 profit and deadline:
27 2
Enter job 13 profit and deadline:
19 14
Enter job 14 profit and deadline:
11 1

Selected Jobs:
Job 6
Job 12
Job 1
Job 8
Job 9
Job 4
Job 5
Job 3
-
Job 7
-
Job 10
```

WORKING:

(Q1) Job - scheduling.

P- {22, 19, 20, 28, 30, 21, 27, 25, 24, 26, 14, 27, 19, 11}

D- {3, 3, 8, 6, 7, 5, 10, 4, 6, 12, 13, 2, 14, 1}

Sort

J5	30	7
J3	29	8
J4	28	6
J7	27	10
J12	27	2
J10	26	12
J8	25	4
J9	24	6
J1	22	3
J6	21	5
J2	19	3
J13	19	14
J11	14	13
J14	11	1

Algorithm :-

1) sort descending with Respect to Profits.

2) Assign Jobs

Assign :- n = 14 - Add Jobs one by one.

J6	J12	J1	J8	J9	J4	J5	J3	-	J7	-	J10	J11	J13
1	2	3	4	5	6	7	8	9	10	11	12	13	14