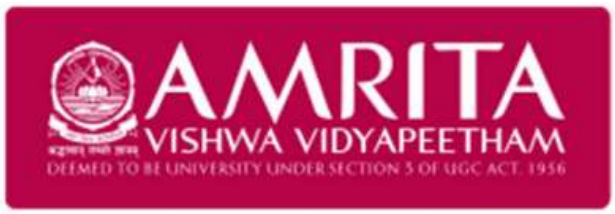


SCHOOL OF COMPUTING

DESIGN AND ANALYSIS OF ALGORITHMS

LAB WORKBOOK

WEEK - 7

NAME : Konda Bhavani

ROLL NO : CH.SC.U4CSE24120

CLASS : CSE-B

Question 1: Let there be 14 jobs with the profit of
22,19,29,28,30,21,27,25,24,26,14,27,19,11 with deadlines
3,3,8,6,7,5,10,4,6,12,13,2,14,1

Implement the greedy algorithm for the Job Sequencing with Deadlines and determine the optimal sequence of jobs that maximizes total profit.

Job sequencing

Let there be 14 Jobs with the profit of [22, 19, 29, 28, 30, 21, 27, 25, 24, 26, 14, 27, 19, 11] and job completion times [3, 3, 8, 6, 7, 5, 10, 4, 6, 12, 13, 2, 14, 1]

Hoffman Coding

Data ANALYTICS AND INTELLIGENCE LABORATORY

D - 2	Y - 2	G - 1
A - 7	I - 3	B - 1
T - 4	C - 2	O - 2
N - 4	S - 1	R - 2
L - 4	E - 3	

Job Sequencing

Q) Let there be 14 Jobs with the profit of [22, 19, 29, 28, 30, 21, 27, 25, 24, 26, 14, 27, 19, 11] with job completion time [3, 3, 8, 6, 7, 5, 10, 4, 6, 12, 13, 2, 14, 1].

Given,

No. of jobs (N) = 14

Profits = $(P_1$ to $P_{14}) = (22, 19, 29, 28, 30, 21, 27, 25, 24, 26, 14, 27, 19, 11]$

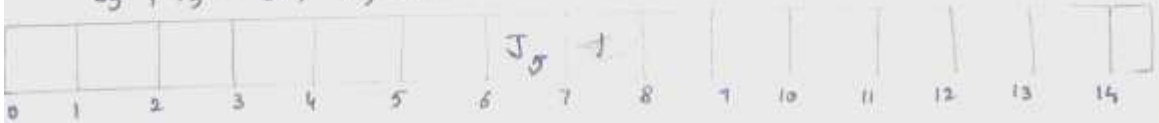
Deadlines = $(D_1$ to $D_{14}) = (3, 3, 8, 6, 7, 5, 10, 4, 6, 12, 13, 2, 14, 1)$.

Step 1:- Arrange the jobs in descending order based on profits and write corresponding deadlines.

30	29	28	27	27	26	25	24	22	21	19	19	14	11
7	8	6	10	2	12	4	6	3	5	3	14	13	1
J_5	J_3	J_4	J_7	J_{12}	J_{10}	J_8	J_9	J_1	J_6	J_2	J_{13}	J_{11}	J_{14}

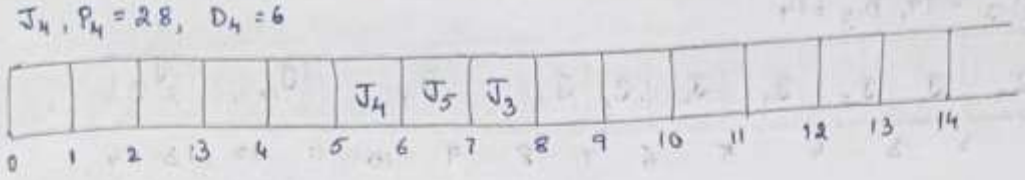
Step 2:- Create slots and Assign jobs

J_5, $P_5 = 30$, $D_5 = 7$.

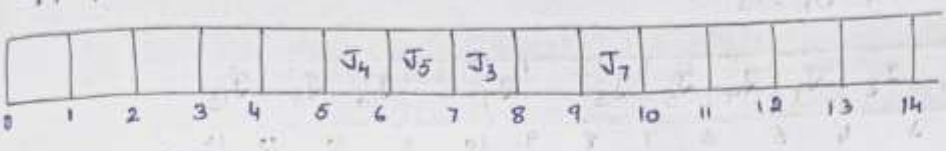


J_3, $P_3 = 29$, $D_3 = 8$

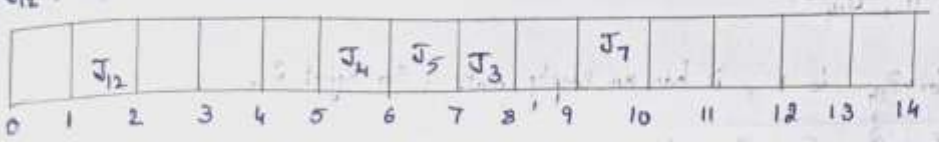
J_4, $P_4 = 28$, $D_4 = 6$



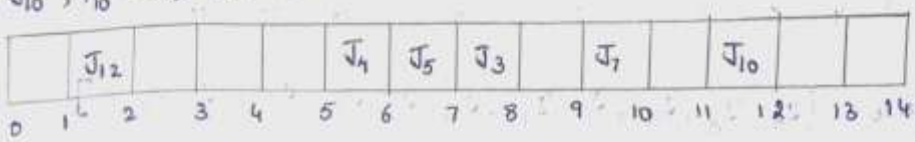
J_7, $P_7 = 27$, $D_7 = 10$



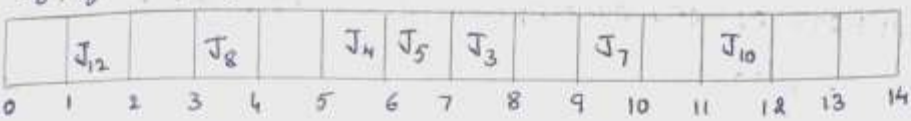
J_{12}, $P_{12} = 27$, $D_{12} = 2$



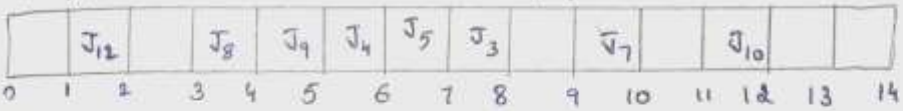
J_{10}, $P_{10} = 26$, $D_{10} = 12$



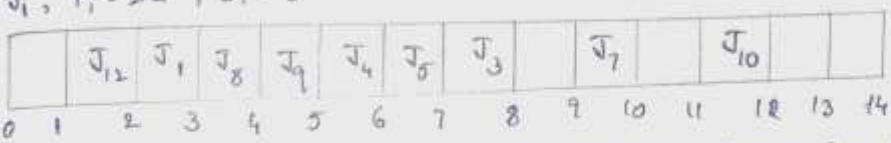
J_8, $P_8 = 25$, $D_8 = 4$



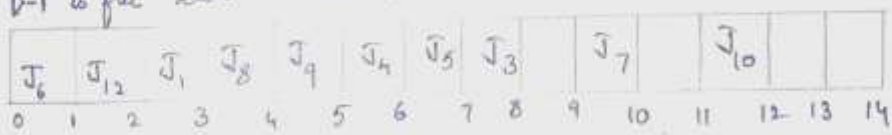
J_9, $P_9 = 24$, $D_9 = 6$, As slot 5-6 is filled check 4-5, As it is empty alot it with J_9.



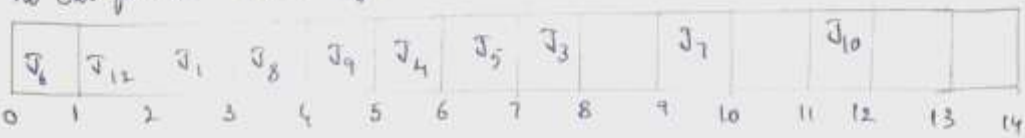
J_1, $P_1 = 22$, $D_1 = 3$



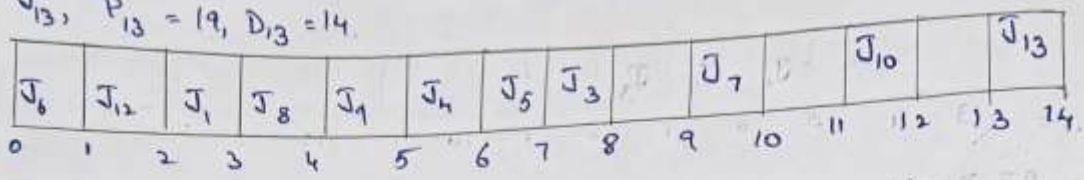
J_6, $P_6 = 21$, $D_6 = 5$, As 5-4 slot is already filled, check previous slots, As only 0-1 is free alot it with J_6



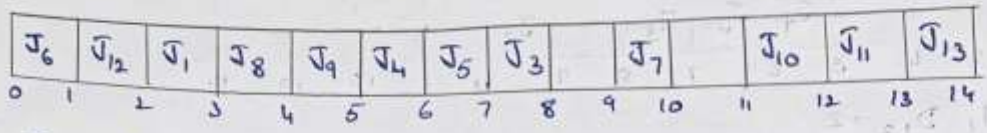
J_2, $P_2 = 19$, $D_2 = 3$. All the slots before deadline i.e, 3 are alloted already. So no slot for J_2 So, we reject J_2.



J_{13}, $P_{13} = 19$, $D_{13} = 14$



Place J_{11}, $P_{11} = 14$, $D_{11} = 13$



J_{14}, $P_{14} = 11$, $D_{14} = 1$

as deadline for P_{14} is 1, it has no empty slots. So Reject J_{14}.

Final Job sequence :- $[J_5, J_3, J_4, J_7, J_{12}, J_{10}, J_8, J_9, J_1, J_6, J_{13}, J_{11}]$

Total profit = 21 + 27 + 22 + 25 + 24 + 28 + 30 + 29 + 27 + 26 + 14 + 19

= 292

CODE:

```
//CH.SC.U4CSE24120
#include <stdio.h>
#define MAX 100
struct Job
{
    int id;
    int profit;
    int deadline;
};
void sortJobs(struct Job jobs[], int n)
{
    int i, j;
    struct Job temp;

    for(i = 0; i < n - 1; i++)
    {
        for(j = 0; j < n - i - 1; j++)
        {
            if(jobs[j].profit < jobs[j + 1].profit)
            {
                temp = jobs[j];
                jobs[j] = jobs[j + 1];
                jobs[j + 1] = temp;
            }
        }
    }
}
int findMaxDeadline(struct Job jobs[], int n)
{
    int i, max = jobs[0].deadline;

    for(i = 1; i < n; i++)
    {
        if(jobs[i].deadline > max)
        {
            max = jobs[i].deadline;
        }
    }
    return max;
```

```
}
int main()
{
    struct Job jobs[MAX];
    int n, i, j;

    printf("Enter number of jobs: ");
    scanf("%d", &n);
    printf("Enter profits:\n");
    for(i = 0; i < n; i++)
    {
        jobs[i].id = i + 1;
        scanf("%d", &jobs[i].profit);
    }
    printf("Enter deadlines:\n");
    for(i = 0; i < n; i++)
    {
        scanf("%d", &jobs[i].deadline);
    }
    sortJobs(jobs, n);
    int maxDeadline = findMaxDeadline(jobs, n);
    int slot[MAX];
    for(i = 1; i <= maxDeadline; i++)
    {
        slot[i] = -1;
    }
    int totalProfit = 0;
    for(i = 0; i < n; i++)
    {
        for(j = jobs[i].deadline; j >= 1; j--)
        {
            if(slot[j] == -1)
            {
                slot[j] = jobs[i].id;
                totalProfit += jobs[i].profit;
                break;
            }
        }
    }
```

```
    printf("\nSlot Arrangement:\n");
    for(i = 1; i <= maxDeadline; i++)
    {
        if(slot[i] == -1)
            printf("Slot %d : _\n", i);
        else
            printf("Slot %d : J%d\n", i, slot[i]);
    }
    printf("\nMaximum Profit = %d\n", totalProfit);
    return 0;
}
```

OUTPUT:

```
Enter number of jobs: 14
Enter profits:
22 19 29 28 30 21 27 25 24 26 14 27 19 11
Enter deadlines:
3 3 8 6 7 5 10 4 6 12 13 2 14 1

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

Maximum Profit = 292
```

Time Complexity:

1. Sorting the jobs by profit

We used Bubble Sort in the program.

Time complexity:$O(n^2)$

2. Finding maximum deadline

We check all jobs once.

Time complexity:$O(n)$

3. Assigning jobs to slots

For each job, we may check up to d slots.$O(n^2)$

Total Time Complexity

$\mathbf{O(n^2) + O(n) + O(n^2)}$**=** $\mathbf{O(n^2)}$

Space Complexity

We use:

- Job array → O(n)
- Slot array → O(d)

Total Space: O(n)