



SCHOOL OF
COMPUTING

Vishal M.D.

CH.SC.U4CSE24150

Week - 6

Date - 12/02/2026

Design and Analysis of Algorithm(23CSE211)

1. Job Scheduling

Code:

```
// CH.SC.U4CSE24150
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int id;
    int deadline;
    int profit;
} Job;
void jobScheduling(Job jobs[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (jobs[j].profit < jobs[j + 1].profit) {
                Job temp = jobs[j];
                jobs[j] = jobs[j + 1];
                jobs[j + 1] = temp;
            }
        }
    }
    int maxDeadline = 0;
    for (int i = 0; i < n; i++) {
        if (jobs[i].deadline > maxDeadline) {
            maxDeadline = jobs[i].deadline;
        }
    }
    int *result = (int *)malloc(maxDeadline * sizeof(int));
    int *slot = (int *)malloc(maxDeadline * sizeof(int));
    int totalProfit = 0;
    for (int i = 0; i < maxDeadline; i++) {
        slot[i] = 0;
    }
    for (int i = 0; i < n; i++) {
        for (int j = jobs[i].deadline - 1; j >= 0; j--) {
            if (slot[j] == 0) {
                result[j] = i;
                slot[j] = 1;
                totalProfit += jobs[i].profit;
                break;
            }
        }
    }
    printf("\nJob Sequence: ");
    for (int i = 0; i < maxDeadline; i++) {
        if (slot[i]) {
            printf("%d ", jobs[result[i]].id);
        }
    }
    printf("\nTotal Profit: %d\n", totalProfit);
    free(result);
    free(slot);
}
int main() {
```

```

printf("CH.SC.U4CSE24150\n");
int n;
printf("Enter number of jobs: ");
scanf("%d", &n);
Job jobs[n];
for (int i = 0; i < n; i++) {
    jobs[i].id = i + 1;
    printf("Enter profit and deadline for Job %d: ", jobs[i].id);
    scanf("%d %d", &jobs[i].profit, &jobs[i].deadline);
}
jobScheduling(jobs, n);
return 0;
}

```

Output:

```

● (base) → DAA gcc Job_Scheduling.c -o Job_scheduling
● (base) → DAA ./Job_scheduling
CH.SC.U4CSE24150
Enter number of jobs: 8
Enter profit and deadline for Job 1: 45 2
Enter profit and deadline for Job 2: 57 1
Enter profit and deadline for Job 3: 3557 4
Enter profit and deadline for Job 4: 57 0
Enter profit and deadline for Job 5: 57 5
Enter profit and deadline for Job 6: 54 0
Enter profit and deadline for Job 7: 574 7
Enter profit and deadline for Job 8: 54 78

Job Sequence: 2 1 3 5 7 8
Total Profit: 4344
○ (base) → DAA

```

Time Complexity: $O(n^2)$

Justification: The algorithm first uses a nested Bubble Sort to sort jobs based on profit, which takes $O(n^2)$. Following this, there is another nested loop structure where, for each of the n jobs, the algorithm scans through the slot array (up to the maximum deadline d) to find an available time slot. If the maximum deadline d is proportional to n , the total time complexity remains $O(n^2)$.

Space Complexity: $O(n+d)$

Justification: The algorithm allocates two dynamic arrays, `result` and `slot`, both of size `maxDeadline` (d). Additionally, it stores an array of n `Job` structures. Since it uses an iterative approach rather than recursion, there is no function call stack overhead. The total auxiliary space is determined by the number of jobs and the range of the deadlines.