LABORATORY REPORT

Algorithm Laboratory (CS-39001)

B.Tech Program in ECSc

Submitted By

Name:- SANNIDHI DEB

Roll No: 2330044

Kalinga Institute of Industrial Technology

(Deemed to be University) Bhubaneswar, India

Autumn, 2025

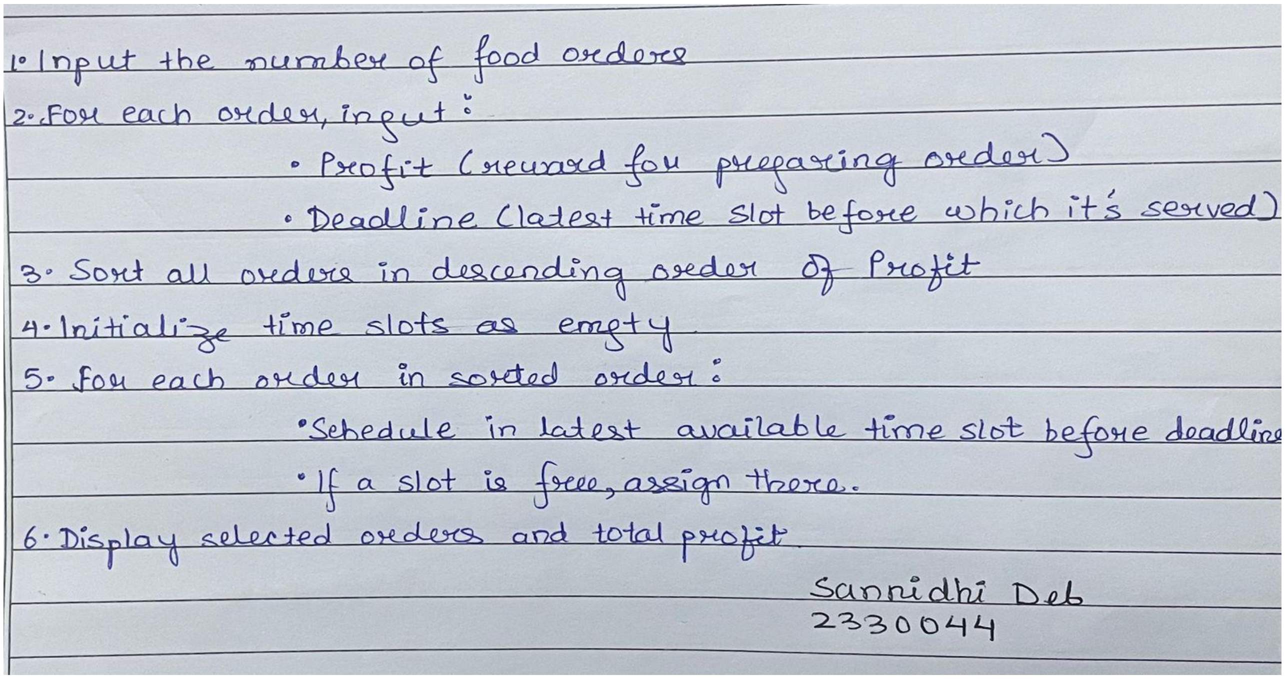
Table of Contents

<table>
<tr><td><b>Experiment Number</b></td><td>10 (Open Ended Experiment)</td></tr>
<tr><td><b>Experiment Title</b></td><td>To design and implement a Greedy algorithm that schedules restaurant orders to achieve maximum total profit, ensuring that all accepted orders are completed before their deadlines.

Number of orders: 5

<table>
<tr><td>Order</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
<tr><td>Deadline</td><td>2</td><td>1</td><td>2</td><td>1</td><td>3</td></tr>
<tr><td>Profit</td><td>100</td><td>19</td><td>27</td><td>25</td><td>15</td></tr>
</table>
</td></tr>
<tr><td><b>Date of Experiment</b></td><td>23/10/2025</td></tr>
<tr><td><b>Date of Submission</b></td><td>30/10/2025</td></tr>
</table>

1.Algorithm:-



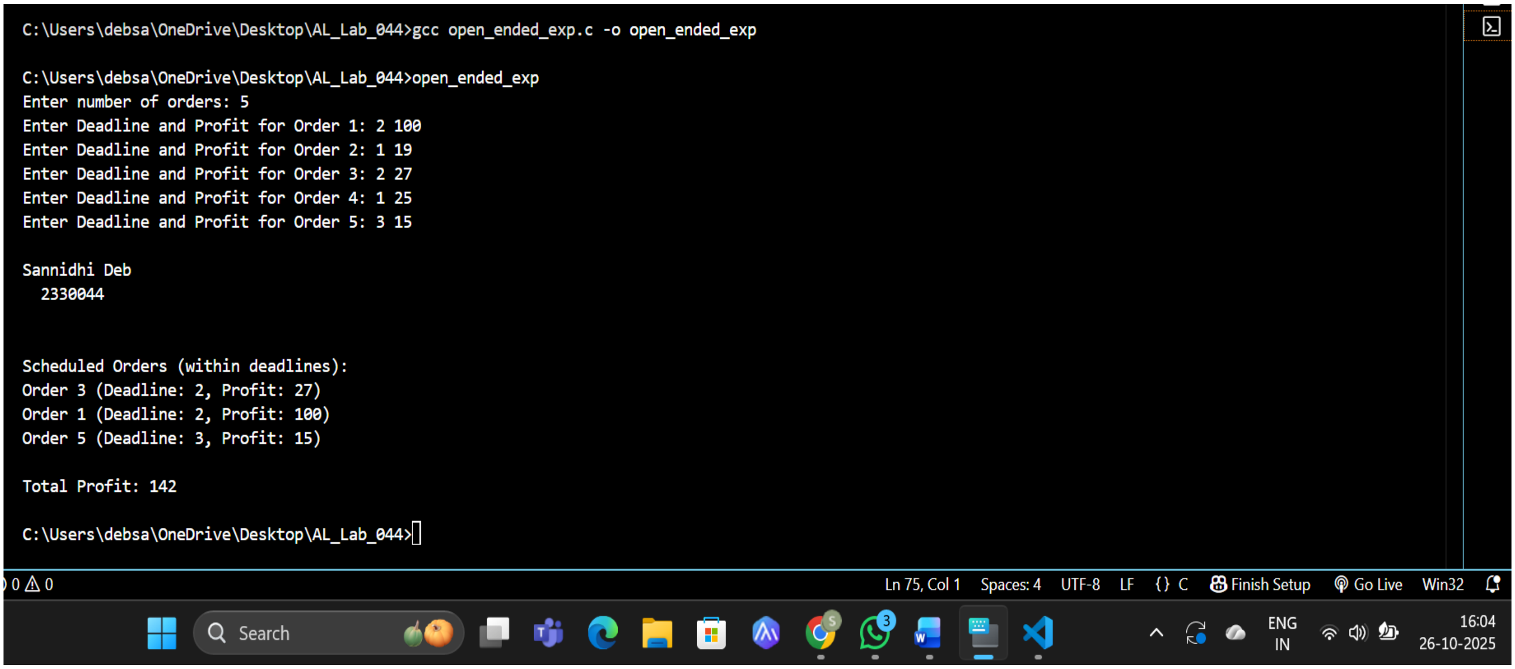
1. Input the number of food orders
2. For each order, input:
   - Profit (reward for preparing order)
   - Deadline (latest time slot before which it's served)
3. Sort all orders in descending order of Profit
4. Initialize time slots as empty.
5. For each order in sorted order:
   - Schedule in latest available time slot before deadline
   - If a slot is free, assign there.
6. Display selected orders and total profit

Sannidhi Deb
2330044

2. Code:-

```
#include <stdio.h>

struct Order {
    int id;
    int deadline;
    int profit;
};

void sortOrders(struct Order orders[], int n) {
    struct Order temp;
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (orders[j].profit < orders[j + 1].profit) {
                temp = orders[j];
                orders[j] = orders[j + 1];
                orders[j + 1] = temp;
            }
        }
    }
}
int findMaxDeadline(struct Order orders[], int n) {
    int max = orders[0].deadline;
    for (int i = 1; i < n; i++) {
        if (orders[i].deadline > max)
            max = orders[i].deadline;
    }
    return max;
}

void scheduleOrders(struct Order orders[], int n) {
    int maxDeadline = findMaxDeadline(orders, n);
    int slots[maxDeadline + 1];
    for (int i = 0; i <= maxDeadline; i++)
        slots[i] = -1;

    int totalProfit = 0;

    for (int i = 0; i < n; i++) {
        for (int j = orders[i].deadline; j > 0; j--) {
            if (slots[j] == -1) {
                slots[j] = i;
                totalProfit += orders[i].profit;
                break;
            }
        }
    }
    printf("\nSannidhi Deb\n  2330044\n\n");
    printf("\nScheduled Orders (within deadlines):\n");
    for (int i = 1; i <= maxDeadline; i++) {
        if (slots[i] != -1) {
            int idx = slots[i];
            printf("Order %d (Deadline: %d, Profit: %d)\n",
                orders[idx].id, orders[idx].deadline, orders[idx].profit);
        }
    }
    printf("\nTotal Profit: %d\n", totalProfit);
}

int main() {
    int n;
    printf("Enter number of orders: ");
    scanf("%d", &n);

    struct Order orders[n];
    for (int i = 0; i < n; i++) {
        orders[i].id = i + 1;
        printf("Enter Deadline and Profit for Order %d: ", i + 1);
        scanf("%d %d", &orders[i].deadline, &orders[i].profit);
    }

    sortOrders(orders, n);
    scheduleOrders(orders, n);
    return 0;
}
```

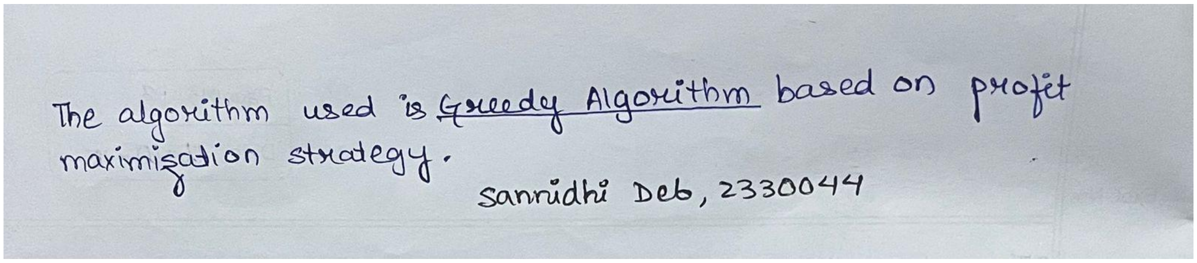
3.Results/Output:- Entire Screen Shot including Date & Time:-



```
C:\Users\debsa\OneDrive\Desktop\AL_Lab_044>gcc open_ended_exp.c -o open_ended_exp

C:\Users\debsa\OneDrive\Desktop\AL_Lab_044>open_ended_exp
Enter number of orders: 5
Enter Deadline and Profit for Order 1: 2 100
Enter Deadline and Profit for Order 2: 1 19
Enter Deadline and Profit for Order 3: 2 27
Enter Deadline and Profit for Order 4: 1 25
Enter Deadline and Profit for Order 5: 3 15

Sannidhi Deb
  2330044


Scheduled Orders (within deadlines):
Order 3 (Deadline: 2, Profit: 27)
Order 1 (Deadline: 2, Profit: 100)
Order 5 (Deadline: 3, Profit: 15)

Total Profit: 142

C:\Users\debsa\OneDrive\Desktop\AL_Lab_044>
```

4. Remarks:-

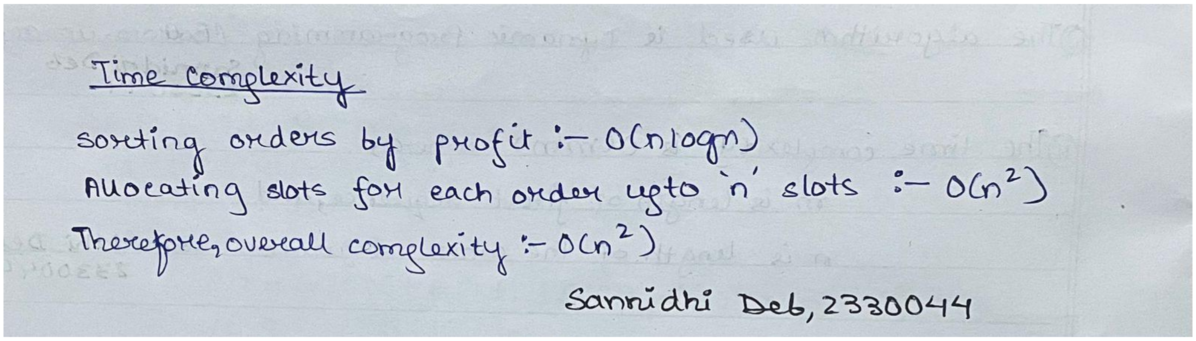
1. What type of algorithm is used?



The algorithm used is Greedy Algorithm based on profit maximisation strategy.

Sannidhi Deb, 2330044

2. Analyze the complexity of your algorithm.



Time complexity

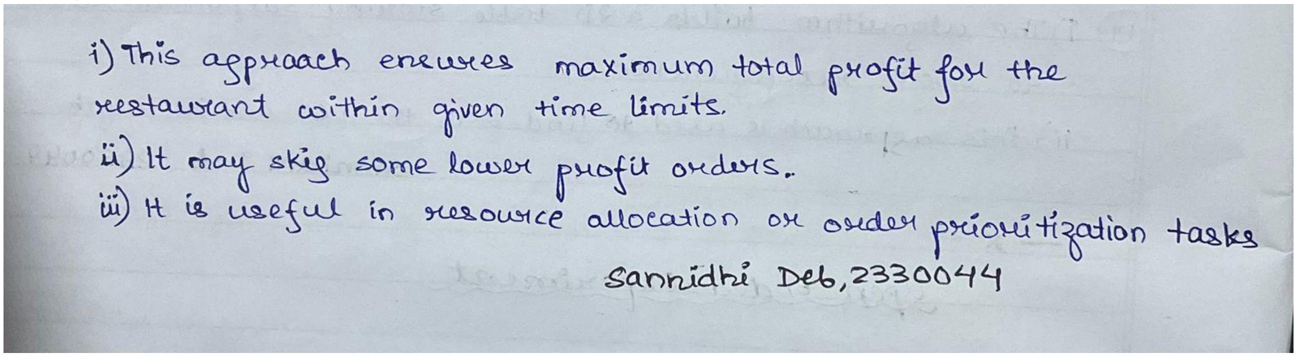
sorting orders by profit :- $O(nlogn)$

Allocating slots for each order upto 'n' slots :- $O(n^2)$

Therefore, overall complexity :- $O(n^2)$

Sannidhi Deb, 2330044

3.Any other observations?



i) This approach ensures maximum total profit for the restaurant within given time limits.
ii) It may skip some lower profit orders.
iii) It is useful in resource allocation or order prioritization tasks

Sannidhi Deb, 2330044

5. Conclusion:- The Restaurant Auto Scheduling System implemented using the Job Sequencing with Deadlines (Greedy Algorithm) approach efficiently schedules restaurant orders to maximize total profit while ensuring deadlines are met. This experiment demonstrates how greedy strategies can be applied to real-world restaurant operations and other scheduling tasks involving limited time and high-profit optimization.

Sannidhi Deb
(2330044)

Signature of the FIC

Sannidhi Deb

(Name of the FIC)