**Restaurant Ordering System - BSCS Operating System Lab Project Documentation**

**Project Title:**

Restaurant Ordering System using Shortest Job First (SJF) Scheduling Algorithm

**Course:**

Operating Systems Lab (FA-2022-BSCS)

**Submitted To:**

Miss Safoora Siddiqi

**Team Members:**

* Musharib Rehman (032)
* Abdullah Arsalan Ali (305)
* Muhammad Asif Ali (216)
* Muhammad Uzair (097)
* Mohid Chishti (179)

**Submission Date:**

Jan 06, 2025

**1. Project Overview:**

The Restaurant Ordering System is a simulation that models the ordering process in a restaurant using the Shortest Job First (SJF) scheduling algorithm. The system allows multiple customers to place their orders, calculates their bill, and processes the orders based on burst time (time taken for each order). The system displays a menu, accepts customer orders with quantities, calculates total prices, and determines the shortest order (in terms of burst time) to be processed first.

**Objective:**

To simulate a restaurant's ordering system with an efficient scheduling algorithm (Shortest Job First - SJF) for processing orders based on burst times.

**2. Functional Requirements:**

* **Display Menu:** The system displays a list of menu items, along with their burst time and price.
* **Place Order:** The customer can select multiple items from the menu, specify the quantity, and add them to the order.
* **Order Summary:** After the order is placed, the system generates a summary displaying the selected items, their burst times, and total price.
* **Payment Handling:** The customer proceeds to payment after confirming the order.
* **Order Processing (SJF):** After multiple orders are placed, the system sorts the orders based on burst time and processes them in the order of the shortest burst time.

**3. System Design:**

**3.1 Data Structures:**

The system uses the following data structure:

**Order Structure:**

typedef struct {

int customerId;

int arrivalTime;

int burstTime; // Total burst time for the order

int totalBill;

char \*items[10]; // Array of item names

int quantities[10]; // Quantities of each item

int itemCount; // Number of items in the order

} Order;

This structure holds the following data:

* **customerId:** Unique ID for each customer.
* **arrivalTime:** The time the customer arrives (indexed by the order).
* **burstTime:** The total time needed to process the customer's order.
* **totalBill:** The total cost of the order.
* **items:** An array of items ordered.
* **quantities:** The quantity of each item ordered.
* **itemCount:** The number of items in the current order.

**3.2 Functions:**

* **displayMenu:** Displays the restaurant's menu with item numbers, names, burst times, and prices.
* **handleOrder:** Handles the order placement by the customer, calculates the burst time, and updates the order's total bill.
* **displayOrderSummary:** Displays a detailed summary of the items ordered, burst times, and total cost.
* **handlePayment:** Processes the payment and thanks the customer.
* **processOrders:** Sorts the orders using the Shortest Job First (SJF) algorithm based on burst times and displays the processing order.

**4. Implementation Details:**

**4.1 Algorithm:**

The system uses the **Shortest Job First (SJF)** scheduling algorithm to process orders. This algorithm selects the order with the shortest burst time for processing. If multiple orders have the same burst time, the system processes them in the order they were placed.

**4.2 Menu Items and Prices:**

The following items are available in the menu:

| **No.** | **Item Name** | **Burst Time** | **Price (Rs.)** |
| --- | --- | --- | --- |
| 1 | Chicken Biryani | 3 | 300 |
| 2 | Nihari | 3 | 400 |
| 3 | Seekh Kebabs | 2 | 250 |
| 4 | Karahi | 4 | 600 |
| 5 | Peshawari Chapli Kebabs | 3 | 500 |
| 6 | Paratha Roll | 2 | 150 |
| 7 | Naan | 1 | 50 |
| 8 | Sada Roti | 1 | 20 |
| 9 | Lassi | 2 | 100 |

* **Burst Time:** Represents the processing time for each item.
* **Price:** The cost of each item in Pakistani Rupees.

**4.3 Process:**

1. **Customer Order:**
   * The customer selects items from the menu.
   * The system calculates the total burst time and total cost.
2. **Order Summary:**
   * The system generates an order summary showing the items ordered, burst times, and total cost.
3. **Payment:**
   * The system processes the payment and thanks the customer.
4. **Order Scheduling (SJF):**
   * After all customers have placed their orders, the system processes the orders by sorting them based on burst time.

**5. Testing and Results:**

**5.1 Testing Procedure:**

The following steps were followed to test the system:

1. Test for a single customer ordering multiple items.
2. Test for multiple customers ordering different quantities of the same item.
3. Test for different combinations of items and quantities.
4. Validate that the orders are processed correctly by burst time.
5. Ensure correct calculation of prices and burst times.

**5.2 Sample Output:**

**Menu:**

-----------------------------------------------------------

| No. | Item Name | Burst Time | Price (Rs.) |

-----------------------------------------------------------

| 1 | Chicken Biryani | 3 | 300 |

| 2 | Nihari | 3 | 400 |

| 3 | Seekh Kebabs | 2 | 250 |

| 4 | Karahi | 4 | 600 |

| 5 | Peshawari Chapli Kebabs | 3 | 500 |

| 6 | Paratha Roll | 2 | 150 |

| 7 | Naan | 1 | 50 |

| 8 | Sada Roti | 1 | 20 |

| 9 | Lassi | 2 | 100 |

-----------------------------------------------------------

**Order Summary:**

Your selected items:

-----------------------------------------------------------

| No. | Item Name | Burst Time | Price (Rs.) |

-----------------------------------------------------------

| 1 | Chicken Biryani | 3 | Rs. 600 |

| 2 | Naan | 1 | Rs. 100 |

-----------------------------------------------------------

Total Bill: Rs. 700

Total Burst Time: 4

**Order Processing (SJF):**

Processing orders based on Shortest Job First (SJF):

Order #1 (Customer 1) will be served.

Order #2 (Customer 2) will be served.

**6. Dry Run:**

**Scenario:**

Let’s consider two customers who place orders. Customer 1 orders **Chicken Biryani (3)** and **Naan (1)**, and Customer 2 orders **Seekh Kebabs (2)** and **Naan (1)**.

**Step 1: Customer 1 Places Order**

1. Customer 1 selects **Chicken Biryani (3)** and **Naan (1)**.
   * **Chicken Biryani:** Burst time = 3 \* 1 = 3, Price = Rs. 300
   * **Naan:** Burst time = 1 \* 1 = 1, Price = Rs. 50
   * Total Bill = Rs. 300 + Rs. 50 = Rs. 350
   * Total Burst Time = 3 + 1 = 4

**Order Summary for Customer 1:**

Your selected items:

-----------------------------------------------------------

| No. | Item Name | Burst Time | Price (Rs.) |

-----------------------------------------------------------

| 1 | Chicken Biryani | 3 | Rs. 300 |

| 2 | Naan | 1 | Rs. 50 |

-----------------------------------------------------------

Total Bill: Rs. 350

Total Burst Time: 4

**Step 2: Customer 2 Places Order**

1. Customer 2 selects **Seekh Kebabs (2)** and **Naan (1)**.
   * **Seekh Kebabs:** Burst time = 2 \* 1 = 2, Price = Rs. 250
   * **Naan:** Burst time = 1 \* 1 = 1, Price = Rs. 50
   * Total Bill = Rs. 250 + Rs. 50 = Rs. 300
   * Total Burst Time = 2 + 1 = 3

**Order Summary for Customer 2:**

Your selected items:

-----------------------------------------------------------

| No. | Item Name | Burst Time | Price (Rs.) |

-----------------------------------------------------------

| 1 | Seekh Kebabs | 2 | Rs. 250 |

| 2 | Naan | 1 | Rs. 50 |

-----------------------------------------------------------

Total Bill: Rs. 300

Total Burst Time: 3

**Step 3: Sorting Orders (SJF)**

* Customer 2 has a total burst time of **3**, which is shorter than Customer 1's **4**.
* The orders are sorted, and Customer 2 is processed first.

**Processing Order:**

Processing orders based on Shortest Job First (SJF):

Order #1 (Customer 2) will be served.

Order #2 (Customer 1) will be served.

**6.1 Algorithm solution:**

### ****Step 1: Given Information****

#### Customers' Orders:

| **Customer** | **Items Ordered** | **Burst Time (BT)** | **Total Price (Rs.)** |
| --- | --- | --- | --- |
| 1 | Chicken Biryani (3), Naan (1) | 4 | 350 |
| 2 | Seekh Kebabs (2), Naan (1) | 3 | 300 |

**Step 2: Apply SJF Scheduling**

SJF prioritizes tasks based on their burst time (shortest first). The customers' orders are sorted based on their total burst times.

**Sorted Burst Times:**

1. Customer 2: Burst Time = 3
2. Customer 1: Burst Time = 4

### ****Step 3: Calculate Turnaround Time (TAT) and Waiting Time (WT)****

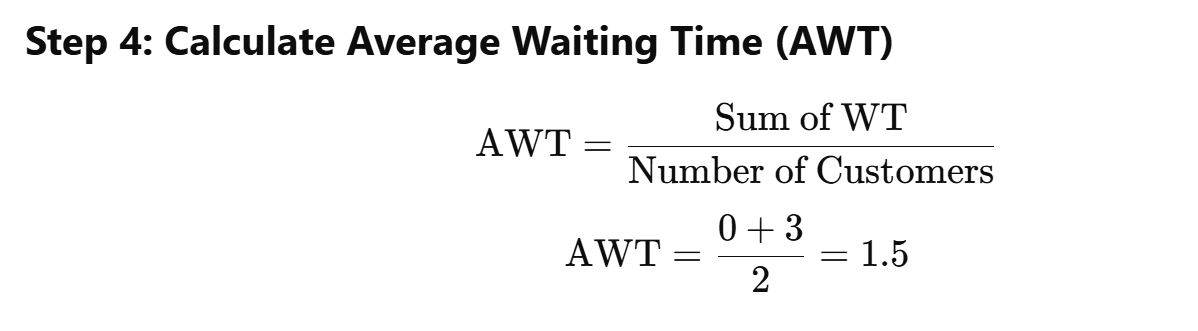
#### Formula:

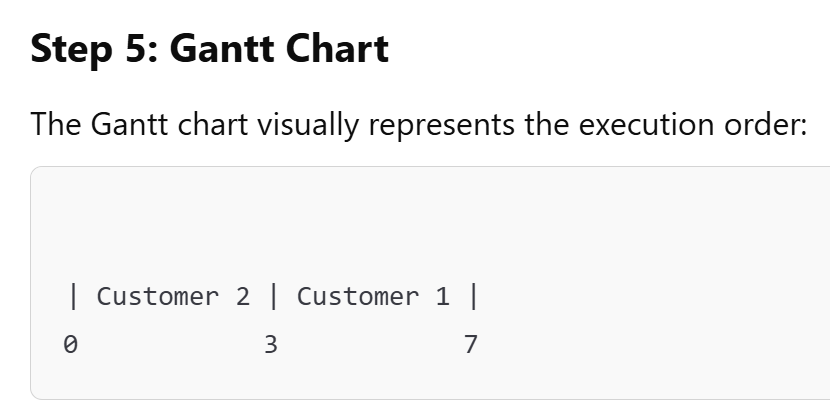
1. **Turnaround Time (TAT)** = Completion Time (CT) - Arrival Time (AT)
2. **Waiting Time (WT)** = TAT - Burst Time (BT)

Assume both customers arrive at time t=0t = 0t=0.

**Execution Order:**

| **Order** | **Customer** | **Burst Time (BT)** | **Completion Time (CT)** | **Turnaround Time (TAT)** | **Waiting Time (WT)** |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 3 | 3 | 0 |
| 2 | 1 | 4 | 7 | 7 | 3 |

****

****

**7. Conclusion:**

This project successfully demonstrates the implementation of a restaurant ordering system using the Shortest Job First (SJF) scheduling algorithm. The system allows customers to place orders, calculates the total price and burst time, and processes orders efficiently based on burst time. The use of burst time for scheduling ensures that smaller orders are processed first, making the system more efficient and fair.

**8. References:**

* Operating Systems Concepts by Miss Safoora Siddiqia
* C Programming Language
* Artificial intelligence