

Name: Bhargav Dhole
Roll No: 150096724014
Batch: Jensen Huang
Case Study No: 27

Problem Statement: Bank ATM Scheduling

Scenario: Customers arrive at an ATM randomly. VIP customers are prioritized to ensure faster access, but regular customers must also be served efficiently.

Proposed Solution:

1. First Come First Serve (FCFS)

Explanation:

Customers are served strictly in the order they arrive, regardless of priority.

Proposed Solution:

- Maintain a single queue sorted by arrival time.
- The ATM serves one customer completely before moving to the next.

2. Priority Scheduling (Preemptive)

Explanation:

VIP customers (priority = 0) are served before regular customers (priority = 1).

In the **preemptive version**, if a VIP arrives while a regular customer is being served, the regular customer's transaction is interrupted.

Proposed Solution:

- Use a priority queue.
- Always select the customer with the highest priority (lowest number).
- If a VIP arrives, pause the regular customer and resume them later.

3. Round Robin (Time-Sharing)

Explanation:

Each customer gets a fixed **time slice (quantum)** to use the ATM. If the transaction is not completed, the customer goes back to the end of the queue.

Proposed Solution:

- Maintain a circular queue of customers.

- Allocate a fixed time quantum to each customer.
- Repeat until all transactions finish.

Screenshots:

```
C atm_scheduling.c ×  atm
C atm_scheduling.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX_CUSTOMERS 50
5
6  typedef struct
7  {
8      int customer_id;
9      int arrival_time;
10     int transaction_time;
11     int priority;
12     int remaining_time;
13     int waiting_time;
14     int turnaround_time;
15 } Customer;
16
17 void fcfs(Customer customers[], int n)
18 {
19     int current_time = 0;
20     float total_waiting_time = 0;
21     float total_turnaround_time = 0;
22
23     printf("\n--- FCFS SCHEDULING ---\n");
24     printf("ID\tArrival\tBurst\tWaiting\tTurnaround\n");
25
26     for (int i = 0; i < n; i++)
27     {
28         if (current_time < customers[i].arrival_time)
29         {
30             current_time = customers[i].arrival_time;
31         }
32
33         customers[i].waiting_time = current_time - customers[i].arrival_time;
34         customers[i].turnaround_time =
35             customers[i].waiting_time + customers[i].transaction_time;
36
37         current_time += customers[i].transaction_time;
38
39         total_waiting_time += customers[i].waiting_time;
40         total_turnaround_time += customers[i].turnaround_time;
41
42         printf("%d\t%d\t%d\t%d\t%d\n",
43                customers[i].customer_id,
44                customers[i].arrival_time,
45                customers[i].transaction_time,
46                customers[i].waiting_time,
47                customers[i].turnaround_time);
48     }
49
50     printf("\nAverage Waiting Time: %.2f\n",
51           total_waiting_time / n);
52     printf("Average Turnaround Time: %.2f\n",
53           total_turnaround_time / n);
54 }
55
56 void priority_preemptive(Customer customers[], int n)
57 {
58     int time = 0, completed = 0;
59     int highest_priority_index = -1;
60     int min_priority;
61     int is_completed[MAX_CUSTOMERS] = {0};
62
63     float total_waiting_time = 0;
64     float total_turnaround_time = 0;
65
66     printf("\n--- PREEMPTIVE PRIORITY SCHEDULING (VIP FIRST) ---\n");
67
68     while (completed < n)
69     {
70         min_priority = 999;
71         highest_priority_index = -1;
72
73         for (int i = 0; i < n; i++)
74         {
75             if (customers[i].arrival_time <= time &&
```

===== BANK ATM SCHEDULING SYSTEM =====

1. Enter Customer Details
2. FCFS Scheduling
3. Priority Scheduling (Preemptive)
4. Round Robin Scheduling
5. Exit

Enter your choice: 3

--- PREEMPTIVE PRIORITY SCHEDULING (VIP FIRST) ---

ID	Priority	Waiting	Turnaround
1	1	0	1
2	1	0	2
3	0	0	3

Average Waiting Time: 0.00

Average Turnaround Time: 2.00

===== BANK ATM SCHEDULING SYSTEM =====

1. Enter Customer Details
2. FCFS Scheduling
3. Priority Scheduling (Preemptive)
4. Round Robin Scheduling
5. Exit

Enter your choice: 4

Enter Time Quantum: 1

--- ROUND ROBIN SCHEDULING ---

ID	Waiting	Turnaround
1	0	1
2	0	2
3	0	3

Average Waiting Time: 0.00

Average Turnaround Time: 2.00

===== BANK ATM SCHEDULING SYSTEM =====

1. Enter Customer Details
2. FCFS Scheduling
3. Priority Scheduling (Preemptive)
4. Round Robin Scheduling
5. Exit

Enter your choice: 5

Exiting program.

bhargavdhole@Bhargavs-MacBook-Air ATM_Scheduling_OS %

```

83         }
84     }
85
86     if (highest_priority_index == -1)
87     {
88         time++;
89         continue;
90     }
91
92     customers[highest_priority_index].remaining_time--;
93     time++;
94
95     if (customers[highest_priority_index].remaining_time == 0)
96     {
97         is_completed[highest_priority_index] = 1;
98         completed++;
99
100        customers[highest_priority_index].turnaround_time =
101            time - customers[highest_priority_index].arrival_time;
102
103        customers[highest_priority_index].waiting_time =
104            customers[highest_priority_index].turnaround_time -
105            customers[highest_priority_index].transaction_time;
106
107        total_waiting_time += customers[highest_priority_index].waiting_time;
108        total_turnaround_time += customers[highest_priority_index].turnaround_time;
109    }
110}
111
112 printf("ID\tPriority\tWaiting\tTurnaround\n");
113 for (int i = 0; i < n; i++)
114 {
115     printf("%d\t%d\t%d\t%d\n",
116           customers[i].customer_id,
117           customers[i].priority,
118           customers[i].waiting_time,
119           customers[i].turnaround_time);
120 }
121
122 printf("\nAverage Waiting Time: %.2f\n",
123       total_waiting_time / n);
124 printf("Average Turnaround Time: %.2f\n",
125       total_turnaround_time / n);
126 }
127
Windsurf: Refactor | Explain | Generate Function Comment | X
128 void round_robin(Customer customers[], int n, int time_quantum)
129 {
130     int time = 0;
131     int completed = 0;
132     int queue[MAX_CUSTOMERS];
133     int front = 0, rear = 0;
134
135     int visited[MAX_CUSTOMERS] = {0};
136
137     float total_waiting_time = 0;
138     float total_turnaround_time = 0;
139
140     // Initially add arrived customers at time 0
141     for (int i = 0; i < n; i++)
142     {
143         if (customers[i].arrival_time == 0)
144         {
145             queue[rear++] = i;
146             visited[i] = 1;
147         }
148     }
149
150     printf("\n--- ROUND ROBIN SCHEDULING ---\n");
151

```

```
270         break;
271
272     case 3:
273         for (int i = 0; i < n; i++)
274         {
275             customers[i].remaining_time = customers[i].transaction_time;
276         }
277         priority_preemptive(customers, n);
278         break;
279
280     case 4:
281     {
282         int time_quantum;
283         printf("Enter Time Quantum: ");
284         scanf("%d", &time_quantum);
285
286         for (int i = 0; i < n; i++)
287         {
288             customers[i].remaining_time = customers[i].transaction_time;
289         }
290
291         round_robin(customers, n, time_quantum);
292         break;
293     }
294
295     case 5:
296         free(customers); // Proper memory dealloc
297         printf("Exiting program.\n");
298         exit(0);
299
300     default:
301         printf("Invalid choice.\n");
302     }
303 }
304
305 return 0;
306 }
```

```

150     printf("\n--- ROUND ROBIN SCHEDULING ---\n");
151
152     while (completed < n)
153     {
154         if (front == rear)
155         {
156             time++;
157             for (int i = 0; i < n; i++)
158             {
159                 if (!visited[i] && customers[i].arrival_time <= time)
160                 {
161                     queue[rear++] = i;
162                     visited[i] = 1;
163                 }
164             }
165             continue;
166         }
167
168         int idx = queue[front++];
169         int exec_time =
170             customers[idx].remaining_time > time_quantum
171             ? time_quantum
172             : customers[idx].remaining_time;
173
174         customers[idx].remaining_time -= exec_time;
175         time += exec_time;
176
177         for (int i = 0; i < n; i++)
178         {
179             if (!visited[i] && customers[i].arrival_time <= time)
180             {
181                 queue[rear++] = i;
182                 visited[i] = 1;
183             }
184         }
185
186         if (customers[idx].remaining_time > 0)
187         {
188             queue[rear++] = idx;
189         }
190         else
191         {
192             completed++;
193             customers[idx].turnaround_time =
194                 time - customers[idx].arrival_time;
195             customers[idx].waiting_time =
196                 customers[idx].turnaround_time -
197                 customers[idx].transaction_time;
198
199             total_waiting_time += customers[idx].waiting_time;
200             total_turnaround_time += customers[idx].turnaround_time;
201         }
202     }
203
204     printf("ID\tWaiting\tTurnaround\n");
205     for (int i = 0; i < n; i++)
206     {
207         printf("%d\t%d\t%d\n",
208               customers[i].customer_id,
209               customers[i].waiting_time,
210               customers[i].turnaround_time);
211     }
212
213     printf("\nAverage Waiting Time: %.2f\n",
214           total_waiting_time / n);
215     printf("Average Turnaround Time: %.2f\n",
216           total_turnaround_time / n);
217 }
218 }
```

```

219 int main()
220 {
221     int choice;
222     int n;
223
224     // Dynamic memory alloc
225     Customer *customers = (Customer *)malloc(MAX_CUSTOMERS * sizeof(Customer));
226
227     if (customers == NULL)
228     {
229         printf("Memory allocation failed.\n");
230         return 1;
231     }
232
233     while (1)
234     {
235         printf("\n===== BANK ATM SCHEDULING SYSTEM =====\n");
236         printf("1. Enter Customer Details\n");
237         printf("2. FCFS Scheduling\n");
238         printf("3. Priority Scheduling (Preemptive)\n");
239         printf("4. Round Robin Scheduling\n");
240         printf("5. Exit\n");
241         printf("Enter your choice: ");
242         scanf("%d", &choice);
243
244         switch (choice)
245         {
246             case 1:
247                 printf("Enter number of customers (max %d): ", MAX_CUSTOMERS);
248                 scanf("%d", &n);
249
250                 for (int i = 0; i < n; i++)
251                 {
252                     printf("\nCustomer %d\n", i + 1);
253                     customers[i].customer_id = i + 1;
254
255                     printf("Arrival Time: ");
256                     scanf("%d", &customers[i].arrival_time);
257
258                     printf("Transaction Duration: ");
259                     scanf("%d", &customers[i].transaction_time);
260
261                     printf("Priority (0 = VIP, 1 = Regular): ");
262                     scanf("%d", &customers[i].priority);
263
264                     customers[i].remaining_time = customers[i].transaction_time;
265                 }
266                 break;
267
268             case 2:
269                 fcfs(customers, n);
270                 break;
271
272             case 3:
273                 for (int i = 0; i < n; i++)
274                 {
275                     customers[i].remaining_time = customers[i].transaction_time;
276                 }
277                 priority_preemptive(customers, n);
278                 break;
279
280             case 4:
281                 {
282                     int time_quantum;
283                     printf("Enter Time Quantum: ");
284                     scanf("%d", &time_quantum);
285
286                     for (int i = 0; i < n; i++)
287                     {
288                         customers[i].remaining_time = customers[i].transaction_time;
289                     }

```

```
bhargavdhole@Bhargavs-MacBook-Air ATM_Scheduling_OS % gcc atm_scheduling.c -o atm
bhargavdhole@Bhargavs-MacBook-Air ATM_Scheduling_OS % ./atm

===== BANK ATM SCHEDULING SYSTEM =====
1. Enter Customer Details
2. FCFS Scheduling
3. Priority Scheduling (Preemptive)
4. Round Robin Scheduling
5. Exit
Enter your choice: 1
Enter number of customers (max 50): 3

Customer 1
Arrival Time: 1
Transaction Duration: 1
Priority (0 = VIP, 1 = Regular): 1

Customer 2
Arrival Time: 2
Transaction Duration: 2
Priority (0 = VIP, 1 = Regular): 1

Customer 3
Arrival Time: 33
Transaction Duration: 3
Priority (0 = VIP, 1 = Regular): 0

===== BANK ATM SCHEDULING SYSTEM =====
1. Enter Customer Details
2. FCFS Scheduling
3. Priority Scheduling (Preemptive)
4. Round Robin Scheduling
5. Exit
Enter your choice: 2

--- FCFS SCHEDULING ---
ID      Arrival Burst  Waiting Turnaround
1       1        1      0          1
2       2        2      0          2
3      33        3      0          3

Average Waiting Time: 0.00
Average Turnaround Time: 2.00
```

Thank You.