

Mohammed Nizamuddin

2503B05144

Lab Task: Round Robin CPU Scheduling (Time Quantum = 3 ms)

A system receives the following processes NOT in arrival order:

Processes	Arrival Time (AT)	Burst Time (BT)
P1	4	7
P2	0	5
P3	6	3
P4	2	9
P5	1	4

Student Tasks

1. Do NOT reorder by arrival time.
Simulate exactly as given.
2. Apply Round Robin Scheduling with Time Quantum = 3 ms.
3. Draw the complete Gantt Chart.
4. Compute for each process:
 - Completion Time (CT)
 - Turnaround Time ($TAT = CT - AT$)
 - Waiting Time ($WT = TAT - BT$)
5. Compute:
 - Average TAT
 - Average WT
6. Write a C program that performs RR scheduling.
7. Compare your manual result with the program output.

CODE:

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
struct GSeg { int pid; int start; int end; };
```

```
int main() {
```

```
    int n = 5;
```

```
    vector<int> AT = {4, 0, 6, 2, 1};
```

```
    vector<int> BT = {7, 5, 3, 9, 4};
```

```
    int quantum = 3;
```

```
    vector<int> rem = BT;
```

```
    vector<int> CT(n, 0);
```

```
    vector<int> added(n, 0);
```

```
    deque<int> q;
```

```
    int time = 0;
```

```
    int completed = 0;
```

```
    for (int i = 0; i < n; ++i) {
```

```
        if (AT[i] <= time && !added[i]) {
```

```
            q.push_back(i);
```

```
            added[i] = 1;
```

```
        }
```

```
    }
```

```
    if (q.empty()) {
```

```
        int nextAt = INT_MAX;
```

```
        for (int i = 0; i < n; ++i) if (!added[i]) nextAt = min(nextAt, AT[i]);
```

```
        time = nextAt;
```

```
        for (int i = 0; i < n; ++i) {
```

```
            if (AT[i] <= time && !added[i]) {
```

```
                q.push_back(i);
```

```
                added[i] = 1;
```

```
            }
```

```
        }
```

```
    }
```

```
    vector<GSeg> gantt;
```

```
    while (completed < n) {
```

```
        if (q.empty()) {
```

```
            int nextAt = INT_MAX;
```

```
            for (int i = 0; i < n; ++i) if (!added[i]) nextAt = min(nextAt, AT[i]);
```

```
            time = nextAt;
```

```

    for (int i = 0; i < n; ++i) {
        if (AT[i] <= time && !added[i]) {
            q.push_back(i);
            added[i] = 1;
        }
    }
    continue;
}

int idx = q.front(); q.pop_front();
int exec = min(rem[idx], quantum);
int start = time;
int end = time + exec;

rem[idx] -= exec;
time = end;

gantt.push_back({idx+1, start, end});

for (int i = 0; i < n; ++i) {
    if (AT[i] <= time && !added[i]) {
        q.push_back(i);
        added[i] = 1;
    }
}

if (rem[idx] == 0) {
    CT[idx] = time;
    completed++;
} else {
    q.push_back(idx);
}
}

cout << "Gantt chart segments:\n";
for (auto &s : gantt) cout << "| P" << s.pid << " ";
cout << "\n";
for (auto &s : gantt) cout << s.start << "   ";
cout << gantt.back().end << "\n\n";

cout << left << setw(6) << "Proc" << setw(6) << "AT" << setw(6) << "BT"
    << setw(6) << "CT" << setw(6) << "TAT" << setw(6) << "WT" << "\n";

double sumTAT = 0.0, sumWT = 0.0;
for (int i = 0; i < n; ++i) {
    int tat = CT[i] - AT[i];
    int wt = tat - BT[i];
    sumTAT += tat;

```

```

sumWT += wt;
cout << "P" << (i+1) << setw(5) << ""
    << setw(6) << AT[i] << setw(6) << BT[i]
    << setw(6) << CT[i] << setw(6) << tat << setw(6) << wt << "\n";
}

cout << fixed << setprecision(2);
cout << "\nAverage TAT = " << (sumTAT / n) << " ms\n";
cout << "Average WT = " << (sumWT / n) << " ms\n";

return 0;
}

```

Explore Problems Contest Discuss Interview Store

Run Code Untitled Save C++

```

1 #include <bits/stdc++.h>
2 using namespace std;
3
4 struct GSeg { int pid; int start; int end; };
5
6 int main() {
7     int n = 5;
8     vector<int> AT = {4, 0, 6, 2, 1};
9     vector<int> BT = {7, 5, 3, 9, 4};
10    int quantum = 3;
11
12    vector<int> rem = BT;
13    vector<int> CT(n, 0);
14    vector<int> added(n, 0);
15
16    deque<int> q;
17    int time = 0;
18    int completed = 0;
19
20    for (int i = 0; i < n; ++i) {
21        if (AT[i] <= time && !added[i]) {
22            q.push_back(i);
23            added[i] = 1;
24        }
25    }

```

Output: Finished Clear Console

Finished in 0 ms

Gantt chart segments:

```

| P2 | P4 | P5 | P2 | P1 | P3 | P4 | P5 | P1 | P4 | P1 |
0   3   6   9   11  14  17  20  21  24  27  28

```

Proc	AT	BT	CT	TAT	WT
P1	4	7	28	24	17
P2	0	5	11	11	6
P3	6	3	17	11	8
P4	2	9	27	25	16
P5	1	4	21	20	16

Average TAT = 18.20 ms
Average WT = 12.60 ms

Share Live Add Snippet

Explore Problems Contest Discuss Interview Store

Run Code Untitled Save C++

```

27 if (q.empty()) {
28     int nextAt = INT_MAX;
29     for (int i = 0; i < n; ++i) if (!added[i]) nextAt = min(nextAt, AT[i]);
30     time = nextAt;
31     for (int i = 0; i < n; ++i) {
32         if (AT[i] <= time && !added[i]) {
33             q.push_back(i);
34             added[i] = 1;
35         }
36     }
37 }
38
39 vector<GSeg> gantt;
40
41 while (completed < n) {
42     if (q.empty()) {
43         int nextAt = INT_MAX;
44         for (int i = 0; i < n; ++i) if (!added[i]) nextAt = min(nextAt, AT[i]);
45         time = nextAt;
46         for (int i = 0; i < n; ++i) {
47             if (AT[i] <= time && !added[i]) {
48                 q.push_back(i);
49                 added[i] = 1;
50             }
51         }

```

Output: Finished Clear Console

Finished in 0 ms

Gantt chart segments:

```

| P2 | P4 | P5 | P2 | P1 | P3 | P4 | P5 | P1 | P4 | P1 |
0   3   6   9   11  14  17  20  21  24  27  28

```

Proc	AT	BT	CT	TAT	WT
P1	4	7	28	24	17
P2	0	5	11	11	6
P3	6	3	17	11	8
P4	2	9	27	25	16
P5	1	4	21	20	16

Average TAT = 18.20 ms
Average WT = 12.60 ms

Share Live Add Snippet

[←](#)
[→](#)
[🔍](#)
[🌟](#)
[📄](#)
[👤](#)
[⋮](#)

[Explore](#)
[Problems](#)
[Contest](#)
[Discuss](#)
[Interview](#)
[Store](#)

[🔔](#)
[🔥](#)
[0](#)
[👤](#)
[Premium](#)

[Run Code](#)
[Untitled](#)
[Save](#)
[C++](#)
[⚙️](#)

```

50         }
51     }
52     continue;
53 }
54
55 int idx = q.front(); q.pop_front();
56 int exec = min(rem[idx], quantum);
57 int start = time;
58 int end = time + exec;
59
60 rem[idx] -= exec;
61 time = end;
62
63 gantt.push_back({idx+1, start, end});
64
65 for (int i = 0; i < n; ++i) {
66     if (AT[i] <= time && !added[i]) {
67         q.push_back(i);
68         added[i] = 1;
69     }
70 }
71
72 if (rem[idx] == 0) {
73     CT[idx] = time;
74     completed++;

```

Output: **Finished**

Clear Console

```

Finished in 0 ms
Gantt chart segments:
| P2 | P4 | P5 | P2 | P1 | P3 | P4 | P5 | P1 | P4 | P1 |
0   3   6   9   11  14  17  20  21  24  27  28

Proc AT  BT  CT  TAT  WT
P1   4   7  28  24  17
P2   0   5  11  11   6
P3   6   3  17  11   8
P4   2   9  27  25  16
P5   1   4  21  20  16

Average TAT = 18.20 ms
Average WT  = 12.60 ms

```

[←](#)
[→](#)
[🔍](#)
[🌟](#)
[📄](#)
[👤](#)
[⋮](#)

[Explore](#)
[Problems](#)
[Contest](#)
[Discuss](#)
[Interview](#)
[Store](#)

[🔔](#)
[🔥](#)
[0](#)
[👤](#)
[Premium](#)

[Run Code](#)
[Untitled](#)
[Save](#)
[C++](#)
[⚙️](#)

```

73     CT[idx] = time;
74     completed++;
75 } else {
76     q.push_back(idx);
77 }
78
79
80 cout << "Gantt chart segments:\n";
81 for (auto &s : gantt) cout << "| P" << s.pid << " ";
82 cout << "\n";
83 for (auto &s : gantt) cout << s.start << " ";
84 cout << gantt.back().end << "\n\n";
85
86 cout << left << setw(6) << "Proc" << setw(6) << "AT" << setw(6) << "BT"
87     << setw(6) << "CT" << setw(6) << "TAT" << setw(6) << "WT" << "\n";
88
89 double sumTAT = 0.0, sumWT = 0.0;
90 for (int i = 0; i < n; ++i) {
91     int tat = CT[i] - AT[i];
92     int wt  = tat - BT[i];
93     sumTAT += tat;
94     sumWT  += wt;
95     cout << "P" << (i+1) << setw(5) << ""
96         << setw(6) << AT[i] << setw(6) << BT[i]
97         << setw(6) << CT[i] << setw(6) << tat << setw(6) << wt << "\n";

```

Output: **Finished**

Clear Console

```

Finished in 0 ms
Gantt chart segments:
| P2 | P4 | P5 | P2 | P1 | P3 | P4 | P5 | P1 | P4 | P1 |
0   3   6   9   11  14  17  20  21  24  27  28

Proc AT  BT  CT  TAT  WT
P1   4   7  28  24  17
P2   0   5  11  11   6
P3   6   3  17  11   8
P4   2   9  27  25  16
P5   1   4  21  20  16

Average TAT = 18.20 ms
Average WT  = 12.60 ms

```

[←](#)
[→](#)
[🔍](#)
[🌟](#)
[📄](#)
[👤](#)
[⋮](#)

[Explore](#)
[Problems](#)
[Contest](#)
[Discuss](#)
[Interview](#)
[Store](#)

[🔔](#)
[🔥](#)
[0](#)
[👤](#)
[Premium](#)

[Run Code](#)
[Untitled](#)
[Save](#)
[C++](#)
[⚙️](#)

```

82 cout << "\n";
83 for (auto &s : gantt) cout << s.start << " ";
84 cout << gantt.back().end << "\n\n";
85
86 cout << left << setw(6) << "Proc" << setw(6) << "AT" << setw(6) << "BT"
87     << setw(6) << "CT" << setw(6) << "TAT" << setw(6) << "WT" << "\n";
88
89 double sumTAT = 0.0, sumWT = 0.0;
90 for (int i = 0; i < n; ++i) {
91     int tat = CT[i] - AT[i];
92     int wt  = tat - BT[i];
93     sumTAT += tat;
94     sumWT  += wt;
95     cout << "P" << (i+1) << setw(5) << ""
96         << setw(6) << AT[i] << setw(6) << BT[i]
97         << setw(6) << CT[i] << setw(6) << tat << setw(6) << wt << "\n";
98 }
99
100 cout << fixed << setprecision(2);
101 cout << "\nAverage TAT = " << (sumTAT / n) << " ms\n";
102 cout << "Average WT  = " << (sumWT / n) << " ms\n";
103
104 return 0;
105 }
106

```

Output: **Finished**

Clear Console

```

Finished in 0 ms
Gantt chart segments:
| P2 | P4 | P5 | P2 | P1 | P3 | P4 | P5 | P1 | P4 | P1 |
0   3   6   9   11  14  17  20  21  24  27  28

Proc AT  BT  CT  TAT  WT
P1   4   7  28  24  17
P2   0   5  11  11   6
P3   6   3  17  11   8
P4   2   9  27  25  16
P5   1   4  21  20  16

Average TAT = 18.20 ms
Average WT  = 12.60 ms

```