-Tanisha Gotadke
1BM21CS229

```
#include <stdio.h>
#include <stdbool.h>
#include <limits.h>

struct Process {
    int pid;
    int burst_time;
    int arrival_time;
    int priority;
    int remaining_time;
    int turnaround_time;
};

void srtfNonPreemptive(struct Process processes[], int n) {
    int total_time = 0;
    int total_turnaround_time = 0;

    for (int i = 0; i < n; i++) {
        total_time += processes[i].burst_time;
    }

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

        for (int j = i + 1; j < n; j++) {
            if (processes[j].burst_time <
processes[shortest_process].burst_time) {
                shortest_process = j;
            }
        }

        struct Process temp = processes[i];
        processes[i] = processes[shortest_process];
```

```
        processes[shortest_process] = temp;

        processes[i].turnaround_time = total_time - processes[i].arrival_time;
        total_turnaround_time += processes[i].turnaround_time;
    }

    printf("SRTF (Non-Preemptive)\n");
    printf("Process\tBurst Time\tArrival Time\tTurnaround Time\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\t\t%d\n", processes[i].pid,
processes[i].burst_time, processes[i].arrival_time,
processes[i].turnaround_time);
    }
    printf("Average Turnaround Time: %.2f\n", (float)total_turnaround_time /
n);
}

void priorityNonPreemptive(struct Process processes[], int n) {
    int total_time = 0;
    int total_turnaround_time = 0;

    for (int i = 0; i < n; i++) {
        total_time += processes[i].burst_time;
    }

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

        for (int j = i + 1; j < n; j++) {
            if (processes[j].priority < processes[highest_priority].priority) {
                highest_priority = j;
            }
        }

        struct Process temp = processes[i];
```

```
        processes[i] = processes[highest_priority];
        processes[highest_priority] = temp;

        processes[i].turnaround_time = total_time - processes[i].arrival_time;
        total_turnaround_time += processes[i].turnaround_time;
    }

    printf("\nPriority (Non-Preemptive)\n");
    printf("Process\tBurst Time\tArrival Time\tPriority\tTurnaround Time\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\t\t%d\t\t%d\n", processes[i].pid,
processes[i].burst_time, processes[i].arrival_time, processes[i].priority,
processes[i].turnaround_time);
    }
    printf("Average Turnaround Time: %.2f\n", (float)total_turnaround_time /
n);
}

void roundRobin(struct Process processes[], int n, int quantum) {
    int total_time = 0;
    int total_turnaround_time = 0;

    int remaining_burst_time[n];
    for (int i = 0; i < n; i++) {
        remaining_burst_time[i] = processes[i].burst_time;
    }

    while (true) {
        bool all_processes_complete = true;

        for (int i = 0; i < n; i++) {
            if (remaining_burst_time[i] > 0) {
                all_processes_complete = false;

                if (remaining_burst_time[i] > quantum) {
```

```
                total_time += quantum;
                remaining_burst_time[i] -= quantum;
            } else {
                total_time += remaining_burst_time[i];
                processes[i].turnaround_time = total_time -
processes[i].arrival_time;
                total_turnaround_time += processes[i].turnaround_time;
                remaining_burst_time[i] = 0;
            }
        }
    }

    if (all_processes_complete)
        break;
  }

  printf("\nRound Robin (Quantum: %d)\n", quantum);
  printf("Process\tBurst Time\tArrival Time\tTurnaround Time\n");
  for (int i = 0; i < n; i++) {
    printf("%d\t%d\t\t%d\t\t%d\n", processes[i].pid,
processes[i].burst_time, processes[i].arrival_time,
processes[i].turnaround_time);
  }
  printf("Average Turnaround Time: %.2f\n", (float)total_turnaround_time /
n);
}

int main() {
  int n;

  printf("Enter the number of processes: ");
  scanf("%d", &n);

  struct Process processes[n];
```

```
    for (int i = 0; i < n; i++) {
        printf("\nProcess %d\n", i + 1);
        processes[i].pid = i + 1;

        printf("Enter burst time: ");
        scanf("%d", &processes[i].burst_time);

        printf("Enter arrival time: ");
        scanf("%d", &processes[i].arrival_time);

        printf("Enter priority: ");
        scanf("%d", &processes[i].priority);
    }

    srtfNonPreemptive(processes, n);
    priorityNonPreemptive(processes, n);

    int quantum;
    printf("\nEnter the time quantum for Round Robin: ");
    scanf("%d", &quantum);
    roundRobin(processes, n, quantum);

    return 0;
}
```

42. Consider the following processes, arrival times, and CPU processing requirements with R-R scheduling algorithm.

Process	CPU Time(in ms)	Arrival Time
A	8	0
B	1	1
C	2	3
D	1	4
E	5	2

```
Enter the number of processes: 5
Process 1
Enter arrival time: 0
Enter burst time: 8
Enter priority: 0
Process 2
Enter arrival time: 1
Enter burst time: 1
Enter priority: 0
Process 3
Enter arrival time: 3
Enter burst time: 2
Enter priority: 0
Process 4
Enter arrival time: 4
Enter burst time: 1
Enter priority: 0
Process 5
Enter arrival time: 2
Enter burst time: 5
Enter priority: 0

Select a scheduling algorithm:
1. SJF Non-preemptive
2. SJF Preemptive
3. Priority Non-preemptive
4. Priority Preemptive
5. Round Robin
Enter your choice: 5

Enter the quantum size for Round Robin: 4

Round Robin Scheduling (Quantum: 4):
Process Turnaround Time Waiting Time
1       16              8
2       4               3
3       4               2
4       4               3
5       15              10
Average Turnaround Time: 8.60
Average Waiting Time: 5.20


...Program finished with exit code 0
Press ENTER to exit console.
```

Consider the following 4 processes with the length of CPU burst time given in milliseconds together with their respective arrival time.

Process	Arrival time	Burst time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

3. If preemptive SJF scheduling is used then average waiting time will be,
(A) 10 ms (B) 12 ms
(C) 6.5 ms (D) 5.5 ms

```
Enter the number of processes: 4

Process 1
Enter burst time: 8
Enter arrival time: 0
Enter priority: 0

Process 2
Enter burst time: 4
Enter arrival time:
1
Enter priority: 0

Process 3
Enter burst time: 9
Enter arrival time: 2
Enter priority: 0

Process 4
Enter burst time: 5
Enter arrival time: 3
Enter priority: 0
SRTF (Non-Preemptive)
Process Burst Time      Arrival Time    Turnaround Time
2       4               1               25
4       5               3               23
1       8               0               26
3       9               2               24
Average Turnaround Time: 24.50

Priority (Non-Preemptive)
Process Burst Time      Arrival Time    Priority        Turnaround Time
2       4               1               0               25
4       5               3               0               23
1       8               0               0               26
3       9               2               0               24
Average Turnaround Time: 24.50
```

Consider the following processes with the corresponding length of CPU burst time given in ms. All processes arrive at time 0

Process	Burst time	Priority
P_1	10	3
P_2	1	1
P_3	2	5
P_4	1	4
P_5	5	2

44. Calculate turn around time of process P_1 if non-preemptive priority scheduling is used. Smaller number implies higher priority.
(A) 10 (B) 6 (C) 16 (D) 17

```
Enter the number of processes: 5

Process 1
Enter burst time: 10
Enter arrival time: 4
Enter priority: 3

Process 2
Enter burst time: 1
Enter arrival time: 4
Enter priority: 1

Process 3
Enter burst time: 2
Enter arrival time: 4
Enter priority: 5

Process 4
Enter burst time: 1
Enter arrival time: 4
Enter priority: 4

Process 5
Enter burst time: 5
Enter arrival time: 4
Enter priority:
2
SRTF (Non-Preemptive)
Process Burst Time      Arrival Time    Turnaround Time
2       1               4               15
4       1               4               15
3       2               4               15
5       5               4               15
1       10              4               15
Average Turnaround Time: 15.00

Priority (Non-Preemptive)
Process Burst Time      Arrival Time    Priority        Turnaround Time
2       1               4               1               15
5       5               4               2               15
1       10              4               3               15
4       1               4               4               15
3       2               4               5               15
Average Turnaround Time: 15.00
```