Assignment - 3

OPERATING SYSTEM

TOPIC: Process Scheduling, - PART1

---

1. Write a C programme to simulate the following **non-preemptive** CPU scheduling algorithms to find the turnaround time and waiting time for the above problem.
    a. FCFS
    b. SJF
    c. Priority

★ FCFS CPU SCHEDULING ALGORITHM

- For the FCFS scheduling algorithm, read the number of processes/jobs in the system, and their CPU burst times.
- The scheduling is performed based on the arrival time of the processes, irrespective of their other parameters.
- Each process will be executed according to its arrival time.
- Calculate the waiting time and turnaround time of each of the processes accordingly.

```c
#include <stdio.h>

void findWaitingTimeFCFS(int processes[], int n, int bt[], int wt[]) {
    wt[0] = 0;
    for (int i = 1; i < n; i++) {
        wt[i] = bt[i - 1] + wt[i - 1];
    }
}

void findTurnAroundTimeFCFS(int processes[], int n, int bt[], int wt[], int tat[]) {
    for (int i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
    }
}

void findAvgTimeFCFS(int processes[], int n, int bt[]) {
    int wt[n], tat[n], total_wt = 0, total_tat = 0;

    findWaitingTimeFCFS(processes, n, bt, wt);
    findTurnAroundTimeFCFS(processes, n, bt, wt, tat);

    printf("Processes  Burst time  Waiting time  Turn around time\n");
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
        printf(" %d\t\t%d\t\t%d\t\t%d\n", processes[i], bt[i], wt[i], tat[i]);
    }

    printf("Average waiting time = %.2f\n", (float)total_wt / (float)n);
    printf("Average turn around time = %.2f\n", (float)total_tat / (float)n);
}

int main() {
    int processes[] = {1, 2, 3};
```

```
    int n = sizeof(processes) / sizeof(processes[0]);
    int burst_time[] = {10, 5, 8};

    printf("FCFS Scheduling:\n");
    findAvgTimeFCFS(processes, n, burst_time);

    return 0;
}
```

```
PS C:\Users\Rudradeep\Desktop\New folder (2)> gcc -o fcfs fcfs.c -mconsole
PS C:\Users\Rudradeep\Desktop\New folder (2)> ./fcfs
FCFS Scheduling:
Processes  Burst time  Waiting time  Turn around time
 1             10             0              10
 2             5              10             15
 3             8              15             23
Average waiting time = 8.33
Average turn around time = 16.00
```

★ SJF CPU SCHEDULING ALGORITHM

- For the SJF scheduling algorithm, read the number of processes/jobs in the system, and their CPU burst times.
- Arrange all the jobs in order with respect to their burst times.
- Two jobs may be in queue with the same execution time, and then the FCFS approach will be performed.
- Each process will be executed according to the length of its burst time.
- Then calculate each process's waiting time and turnaround time accordingly.

```
#include <stdio.h>

void findWaitingTimeSJF(int processes[], int n, int bt[], int wt[]) {
    int rt[n];
    for (int i = 0; i < n; i++) {
        rt[i] = bt[i];
    }

    int complete = 0, t = 0, minm = 10000, shortest = 0, finish_time;
    int check = 0;

    while (complete != n) {
        for (int j = 0; j < n; j++) {
            if ((rt[j] < minm) && (rt[j] > 0)) {
                minm = rt[j];
                shortest = j;
                check = 1;
            }
        }

        if (check == 0) {
            t++;
            continue;
        }

        rt[shortest]--;
```

```
        minm = rt[shortest];
        if (minm == 0) {
            minm = 10000;
        }

        if (rt[shortest] == 0) {
            complete++;
            check = 0;
            finish_time = t + 1;
            wt[shortest] = finish_time - bt[shortest];

            if (wt[shortest] < 0) {
                wt[shortest] = 0;
            }
        }
        t++;
    }
}

void findTurnAroundTimeSJF(int processes[], int n, int bt[], int wt[], int tat[]) {
    for (int i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
    }
}

void findAvgTimeSJF(int processes[], int n, int bt[]) {
    int wt[n], tat[n], total_wt = 0, total_tat = 0;

    findWaitingTimeSJF(processes, n, bt, wt);
    findTurnAroundTimeSJF(processes, n, bt, wt, tat);

    printf("Processes  Burst time  Waiting time  Turn around time\n");
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
        printf(" %d\t\t%d\t\t%d\t\t%d\n", processes[i], bt[i], wt[i], tat[i]);
    }

    printf("Average waiting time = %.2f\n", (float)total_wt / (float)n);
    printf("Average turn around time = %.2f\n", (float)total_tat / (float)n);
}

int main() {
    int processes[] = {1, 2, 3};
    int n = sizeof(processes) / sizeof(processes[0]);
    int burst_time[] = {10, 5, 8};

    printf("SJF Scheduling:\n");
    findAvgTimeSJF(processes, n, burst_time);
```

```
    return 0;
}
```

```
PS C:\Users\Rudradeep\Desktop\New folder (2)> gcc -o sjf sjf.c -mconsole
PS C:\Users\Rudradeep\Desktop\New folder (2)> ./sjf
SJF Scheduling:
Processes  Burst time  Waiting time  Turn around time
 1             10             13             23
 2             5              0              5
 3             8              5              13
Average waiting time = 6.00
Average turn around time = 13.67
PS C:\Users\Rudradeep\Desktop\New folder (2)>
```

★ PRIORITY CPU SCHEDULING ALGORITHM

- For the priority scheduling algorithm, read the number of processes/jobs in the system, their CPU burst times, and the priorities.
- Arrange all the jobs in order with respect to their priorities.
- There may be two jobs in queue with the same priority, and then FCFS approach will be performed.
- Each process will be executed according to its priority.
- Calculate the waiting time and turnaround time of each of the processes accordingly.

```c
#include <stdio.h>

void findWaitingTimePriority(int processes[], int n, int bt[], int pr[], int wt[]) {
    int rt[n];
    for (int i = 0; i < n; i++) {
        rt[i] = bt[i];
    }

    int complete = 0, t = 0, minm = 10000, highest_priority = -1, finish_time;
    int check = 0;

    while (complete != n) {
        for (int j = 0; j < n; j++) {
            if ((pr[j] > highest_priority) && (rt[j] > 0)) {
                minm = rt[j];
                highest_priority = pr[j];
                check = 1;
            }
        }

        if (check == 0) {
            t++;
            continue;
        }

        rt[highest_priority]--;
```

```c
        if (rt[highest_priority] == 0) {
            complete++;
            check = 0;
            finish_time = t + 1;
            wt[highest_priority] = finish_time - bt[highest_priority];

            if (wt[highest_priority] < 0) {
                wt[highest_priority] = 0;
            }
        }
        t++;
    }
}

void findTurnAroundTimePriority(int processes[], int n, int bt[], int wt[], int tat[]) {
    for (int i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
    }
}

void findAvgTimePriority(int processes[], int n, int bt[], int pr[]) {
    int wt[n], tat[n], total_wt = 0, total_tat = 0;

    findWaitingTimePriority(processes, n, bt, pr, wt);
    findTurnAroundTimePriority(processes, n, bt, wt, tat);

    printf("Processes  Burst time  Waiting time  Turn around time\n");
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
        printf(" %d\t\t%d\t\t%d\t\t%d\n", processes[i], bt[i], wt[i], tat[i]);
    }

    printf("Average waiting time = %.2f\n", (float)total_wt / (float)n);
    printf("Average turn around time = %.2f\n", (float)total_tat / (float)n);
}

int main() {
    int processes[] = {1, 2, 3};
    int n = sizeof(processes) / sizeof(processes[0]);
    int burst_time[] = {10, 5, 8};
    int priority[] = {2, 1, 3};

    printf("Priority Scheduling:\n");
    findAvgTimePriority(processes, n, burst_time, priority);

    return 0;
}
```

```
PS C:\Users\Rudradeep\Desktop\New folder (2)> gcc -o priority priority.c -mconsole
PS C:\Users\Rudradeep\Desktop\New folder (2)> ./priority
Priority Scheduling:
Processes  Burst time  Priority  Waiting time  Turn around time
 1             10             2              5              15
 2             5              1              0              5
 3             8              3              15             23
Average waiting time = 6.67
Average turn around time = 14.33
PS C:\Users\Rudradeep\Desktop\New folder (2)>
```