Assignment-3

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

- FCFS CPU SCHEDULING ALGORITHM
    - For FCFS scheduling algorithm, read the number of processes/jobs in the system, their CPU burst times.
    - The scheduling is performed on the basis of 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.
- SJF CPU SCHEDULING ALGORITHM
    - For SJF scheduling algorithm, read the number of processes/jobs in the system, their CPU burst times.
    - Arrange all the jobs in order with respect to their burst times.
    - There may be two jobs in queue with the same execution time, and then FCFS approach is to be performed.
    - Each process will be executed according to the length of its burst time.
    - Then calculate the waiting time and turnaround time of each of the processes accordingly.
- PRIORITY CPU SCHEDULING ALGORITHM
    - For 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 is to be performed.
    - Each process will be executed according to its priority.
    - Calculate the waiting time and turnaround time of each of the processes accordingly.

Answer:

```
#include <stdio.h>
#include <stdlib.h>

struct Process {
    int id;
    int burstTime;
    int arrivalTime;
    int priority;
};

void swap(struct Process *xp, struct Process *yp) {
    struct Process temp = *xp;
    *xp = *yp;
    *yp = temp;
}

// Function to perform FCFS scheduling
void fcfs(struct Process *processes, int n) {
    int wt[n], tat[n];

    wt[0] = 0;
```

```
    tat[0] = processes[0].burstTime;

    for (int i = 1; i < n; i++) {
        wt[i] = wt[i - 1] + processes[i - 1].burstTime;
        tat[i] = wt[i] + processes[i].burstTime;
    }

    printf("\nFCFS Scheduling:\n");
    printf("Process\tBurst Time\tWaiting Time\tTurnaround Time\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\t\t%d\n", processes[i].id, processes[i].burstTime, wt[i], tat[i]);
    }
}

// Function to perform SJF scheduling
void sjf(struct Process *processes, int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (processes[j].burstTime > processes[j + 1].burstTime) {
                swap(&processes[j], &processes[j + 1]);
```

```
            }
        }
    }

    fcfs(processes, n);
}

// Function to perform Priority scheduling
void priority(struct Process *processes, int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (processes[j].priority > processes[j + 1].priority) {
                swap(&processes[j], &processes[j + 1]);
            }
        }
    }

    fcfs(processes, 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++) {
        processes[i].id = i + 1;
        printf("Enter arrival time, burst time, and priority for
Process %d: ", i + 1);
        scanf("%d %d %d", &processes[i].arrivalTime,
&processes[i].burstTime, &processes[i].priority);
    }

    sjf(processes, n);
    priority(processes, n);

    return 0;
}
```

Output:

```
FCFS Scheduling:
Process 1    Burst Time    Waiting Time    Turnaround Time
1            6             0               6
2            4             6               10
3            7             10              17
4            5             17              22

SJF Scheduling:
Process 2    Burst Time    Waiting Time    Turnaround Time
2            4             0               4
1            6             4               10
3            7             10              17
4            5             17              22

Priority Scheduling:
Process 2    Burst Time    Waiting Time    Turnaround Time
2            4             0               4
3            7             4               11
1            6             11              17
4            5             17              22
```