Week 2

#include <stdio.h>

#include <stdlib.h>

#define MAX\_PROCESSES 10

typedef struct {

int pid; // Process ID

int arrival; // Arrival time

int burst; // Burst time

} Process;

// Function to implement FCFS scheduling

void fcfsScheduling(Process queue[], int n) {

int waiting\_time = 0, turnaround\_time = 0, completion\_time = 0;

int total\_waiting\_time = 0, total\_turnaround\_time = 0;

printf("\nPID\tArrival\tBurst\tCompletion\tTurnaround\tWaiting\n");

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

if (completion\_time < queue[i].arrival) {

completion\_time = queue[i].arrival;

}

completion\_time += queue[i].burst;

int turnaround = completion\_time - queue[i].arrival;

int waiting = turnaround - queue[i].burst;

// Accumulate waiting time and turnaround time for average calculation

total\_turnaround\_time += turnaround;

total\_waiting\_time += waiting;

// Print details of each process

printf("%d\t%d\t%d\t%d\t\t%d\t\t%d\n", queue[i].pid, queue[i].arrival, queue[i].burst,

completion\_time, turnaround, waiting);

}

// Calculate and display average waiting time and average turnaround time

if (n > 0) {

float avg\_waiting\_time = (float)total\_waiting\_time / n;

float avg\_turnaround\_time = (float)total\_turnaround\_time / n;

printf("\nAverage Waiting Time: %.2f", avg\_waiting\_time);

printf("\nAverage Turnaround Time: %.2f\n", avg\_turnaround\_time);

}

}

// Function to perform multi-level queue scheduling

void multiLevelQueueScheduling(Process systemQueue[], int sysCount, Process userQueue[], int userCount) {

printf("\nExecuting System Processes (Higher Priority):\n");

fcfsScheduling(systemQueue, sysCount);

printf("\nExecuting User Processes (Lower Priority):\n");

fcfsScheduling(userQueue, userCount);

}

int main() {

int n, sysCount = 0, userCount = 0;

Process systemQueue[MAX\_PROCESSES], userQueue[MAX\_PROCESSES];

printf("Enter the number of processes: ");

scanf("%d", &n);

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

Process p;

int type;

printf("\nEnter details for process %d\n", i + 1);

printf("Process ID: ");

scanf("%d", &p.pid);

printf("Arrival Time: ");

scanf("%d", &p.arrival);

printf("Burst Time: ");

scanf("%d", &p.burst);

printf("Enter process type (1 - System, 2 - User): ");

scanf("%d", &type);

if (type == 1) {

systemQueue[sysCount++] = p;

} else {

userQueue[userCount++] = p;

}

}

multiLevelQueueScheduling(systemQueue, sysCount, userQueue, userCount);

return 0;

}

