LAB 3

3. Write a C program to simulate Real-Time CPU Scheduling algorithms:

a) Rate- Monotonic
b) Earliest-deadline First

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

#define MAX 10

typedef struct{
    int id;
    int period;
    int execution_time;
    int remaining_time;
    int next_start_time;
}Task;

void sort_by_priority(Task tasks[],int n){
    for(int i=0;i<n-1;i++){
        for(int j=0;j<n-i-1;j++){
            if(tasks[j].period>tasks[j+1].period){
                Task temp=tasks[j];
                tasks[j]=tasks[j+1];
                tasks[j+1]=temp;
            }
        }
    }
}

void rms_scheduling(Task tasks[],int n,int total_time){
    sort_by_priority(tasks,n);

    for(int t=0;t<total_time;t++){
        for(int i=0;i<n;i++){
            if(t%tasks[i].period==0){
                tasks[i].remaining_time=tasks[i].execution_time;
            }
        }

        int executed=0;
        for(int i=0;i<n;i++){
            if(tasks[i].remaining_time>0){
```

```
                printf("Time %d: Executing Task %d\n",t,tasks[i].id);
                tasks[i].remaining_time--;
                executed=1;
                break;
            }
        }

        if(!executed){
            printf("Time %d: Idle\n",t);
        }
    }
}

int main(){
    Task tasks[MAX];
    int num_tasks,total_time;

    printf("Enter the number of tasks: ");
    scanf("%d",&num_tasks);

    if(num_tasks>MAX||num_tasks<=0){
        printf("Error: Invalid number of tasks.\n");
        return 1;
    }

    for(int i=0;i<num_tasks;i++){
        printf("Enter the period and execution time for Task %d: ",i+1);
        tasks[i].id=i+1;
        scanf("%d %d",&tasks[i].period,&tasks[i].execution_time);
        tasks[i].remaining_time=0;
    }

    printf("Enter the total simulation time: ");
    scanf("%d",&total_time);

    rms_scheduling(tasks,num_tasks,total_time);
    return 0;
}
```

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

#define MAX 10

typedef struct{
    int id;
    int period;
```

```c
    int execution_time;
    int remaining_time;
    int next_deadline;
}Task;

void sort_by_deadline(Task tasks[],int n){
    for(int i=0;i<n-1;i++){
        for(int j=0;j<n-i-1;j++){
            if(tasks[j].next_deadline>tasks[j+1].next_deadline){
                Task temp=tasks[j];
                tasks[j]=tasks[j+1];
                tasks[j+1]=temp;
            }
        }
    }
}

void edf_scheduling(Task tasks[],int n,int total_time){
    for(int t=0;t<total_time;t++){
        for(int i=0;i<n;i++){
            if(t%tasks[i].period==0){
                tasks[i].remaining_time=tasks[i].execution_time;
                tasks[i].next_deadline=t+tasks[i].period;
            }
        }
        sort_by_deadline(tasks,n);
        int executed=0;
        for(int i=0;i<n;i++){
            if(tasks[i].remaining_time>0){
                printf("Time %d: Executing Task %d\n",t,tasks[i].id);
                tasks[i].remaining_time--;
                executed=1;
                break;
            }
        }
        if(!executed){
            printf("Time %d: Idle\n",t);
        }
    }
}

int main(){
    Task tasks[MAX];
    int num_tasks,total_time;
    printf("Enter the number of tasks: ");
    scanf("%d",&num_tasks);
    if(num_tasks>MAX||num_tasks<=0){
```

```
        printf("Error: Invalid number of tasks. Please enter a number between
1 and %d.\n",MAX);
        return 1;
    }
    for(int i=0;i<num_tasks;i++){
        printf("Enter the period and execution time for Task %d: ",i+1);
        tasks[i].id=i+1;
        scanf("%d %d",&tasks[i].period,&tasks[i].execution_time);
        tasks[i].remaining_time=0;
        tasks[i].next_deadline=tasks[i].period;
    }
    printf("Enter the total simulation time: ");
    scanf("%d",&total_time);
    edf_scheduling(tasks,num_tasks,total_time);
    return 0;
}
```

OUTPUT:

//RMS

```
Enter the number of tasks: 3
Enter the period and execution time for Task 1: 20 3
Enter the period and execution time for Task 2: 5 2
Enter the period and execution time for Task 3: 10 2
Enter the total simulation time: 20
Time 0: Executing Task 2
Time 1: Executing Task 2
Time 2: Executing Task 3
Time 3: Executing Task 3
Time 4: Executing Task 1
Time 5: Executing Task 2
Time 6: Executing Task 2
Time 7: Executing Task 1
Time 8: Executing Task 1
Time 9: Idle
Time 10: Executing Task 2
Time 11: Executing Task 2
Time 12: Executing Task 3
Time 13: Executing Task 3
Time 14: Idle
Time 15: Executing Task 2
Time 16: Executing Task 2
Time 17: Idle
Time 18: Idle
Time 19: Idle
```

//EDF

```
Enter the number of tasks: 3
Enter the period and execution time for Task 1: 20 3
Enter the period and execution time for Task 2: 5 2
Enter the period and execution time for Task 3: 10 2
Enter the total simulation time: 20
Time 0: Executing Task 2
Time 1: Executing Task 2
Time 2: Executing Task 3
Time 3: Executing Task 3
Time 4: Executing Task 1
Time 5: Executing Task 2
Time 6: Executing Task 2
Time 7: Executing Task 1
Time 8: Executing Task 1
Time 9: Idle
Time 10: Executing Task 2
Time 11: Executing Task 2
Time 12: Executing Task 3
Time 13: Executing Task 3
Time 14: Idle
Time 15: Executing Task 2
Time 16: Executing Task 2
Time 17: Idle
Time 18: Idle
Time 19: Idle
```