Write a C-program to simulate Real-Time CPU scheduling algorithm Earliest deadline-first.

→

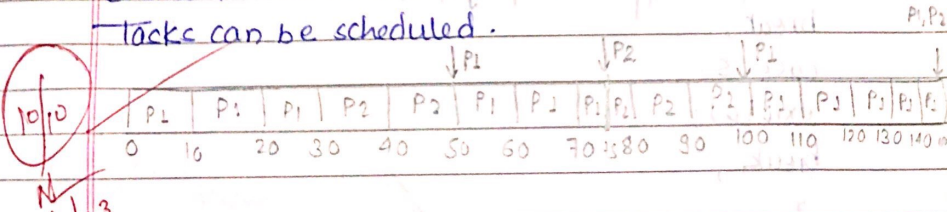
```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int deadline;
    int execution;
    int execution_copy;
} Tasks;
int min(Task *tasks, int n);
void update_execution_copy(Task *tasks, int n);
void execute_task(Task *tasks, int task_id, int
  time);
int main()
{
int n, timer=0;
float cpu_utilization;
Printf("Enter number of tasks:");
Scanf("%d", &n);
Task *tasks=malloc(n*sizeof(Task));
for(int i=0; i<n; i++) {
Printf("Enter task %d parameters:\n", i+1);
Printf("Execution time:");
scanf("%d", &tasks[i].execution);
printf("Deadline time:");
scanf("%d", &tasks[i].deadline);
tacks[i].execution_copy=0;
}
CPU utilization=0;
for(int i=0; i<n; i++)
{
CPU_utilization+=(float)task[i].execution/
    (float)tasks[i].deadline;
```

```
}
Printf("CPU utilization: %f\n", CPU_utilization);
if (CPU_utilization < 1)
Printf("Tasks can be scheduled.\n");
else
Printf("schedule isn't feasible.\n");
while (1) {
int(active_task_id == -1) {
Printf("%d Idle\n", timer);
}
else {
execute_task(tasks, active_task_id, timer);
if(tasks[active_task_id].execution_copy==0){
update_execution_copy(tasks, active_task_id);
}
}
timer++;
int all_completed=1;
for(int i=0; i<n; i++) {
if(tasks[i].execution_copy>0) {
all_completed=0;
break;
}
}
if(all_completed){
break;
}
}
free(tasks);
return 0;
}
int min(Task * tasks, int n) {
int min_deadline = __INT_MAX__;
int task_id = -1;
```

```
for (int i=0; i<n; i++){
if (tasks[i].execution_copy>0 && tasks[i].deadline<
min_deadline){
min_deadline = tasks[i].deadline;
    task_id = i;
}
}
return task_id;
}
void update_execution_copy (Task *tasks, int n){
tasks[n].execution_copy = tasks[n].execution;
}
Void execute_task (Task *tasks, int task_id, int time){
tasks[task_id].execution_copy--;
printf("%d Task %d\n", time, task_id+1);
}
```

Output:

```
Enter number of tasks: 2
Enter Task 1 parameters:
Execution time: 25
Deadline time: 50
Enter Task 2 parameters:
Execution time: 30
deadline time: 75
CPU utilization : 0.900000
Tasks can be scheduled.
```



10/10

19/7/23

```
Enter number of tasks: 2
Enter Task 1 parameters:
Execution time: 25
Deadline time: 50
Enter Task 2 parameters:
Execution time: 30
Deadline time: 75
CPU Utilization: 0.900000
Tasks can be scheduled.
0 Idle
```