write a c program to implement Rate monotonic scheduling:

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_TASKS 10
typedef struct {
    int Ti;
    int Ci;
    int deadline;
    int RT;
    int id;
} Task;
void Input (Task tasks[], int *n_tasks) {
    printf("Enter no. of tasks");
    scanf("%d", &n_tasks);
    for (int i=0; i<*n_tasks; i++) {
        task[i].id = i++;
        printf("Enter Ti of task %d", i+1);
        scanf("%d", &tasks[i].[i]);
        printf("Enter execution time of task %d
                i+1);
        scanf("%d", &tasks[i].Ci);
        tasks[i].deadline = tasks[i].Ti;
        tasks[i].RT = tasks[i].Ci;
    }
}
int compare_by_period (const void *a const void
{
    return ((Task*)a)->Ti - ((Task*))->Ti
}
void RMS (Task tasks[], int n_tasks, int
                time_frame)
{
```

```
qsort (tasks, n_tasks, sizeof (Tasks), compare_
    by_Period);
printf ("\n Rate_monotonic scheduling");
for (int time=0; time < time_frame; time++)
{
    int s_tasks = -1;
    for (int i=0; i<n_tasks; i++) {
        if (time % tasks[i].Ti==0) {
            tasks[i].RT = tasks[i].Ci;
        }
    if (tasks[i].RT>0 && (s_task == -1) ||
        tasks[i].Ti < tasks[s_task].Ti)) {
            s_task = i;
        }
    }
    if (s_task != -1) {
        printf ("Time %d: task %d \n", time,
            tasks[s_task].id);
        tasks[s_task].RT--;
    }
    else {
        printf ("Time %d : Idle\n", time);
    }
}
}

int main() {
    Task tasks[max_tasks];
    int n_tasks;
    int time_frame;
    input (tasks, &n_tasks);
    printf ("Enter time frame for simulation:");
    scanf ("%d", &time_frame);
    rms (tasks, n_tasks, time_frame);
    return 0;
}
```

PAGE NO :
DATE :

Output:

Enter no. of tasks : 3
Enter Ti of task 1 : 20
Enter Execution time of task 1 : 3
Enter Ti of task 2 : 5
Enter Execution time of task 2 : 2
Enter Ti of task 3 : 10
Enter time ~~frame~~ execution of task 3 : 2
Enter time frame for simulation : 20
Rate monotonic scheduling :
Time 0 : task 2
time 1 : task 2
time 2 ; task 3
time 3 : task 3
time 4 : task 1
time 5 : task 2
time 6 : task 2
time 7 : task 1
time 8 : task 1
time 9 ; Idle
time 10 ; task 2
time 11 : task 2
time 12 : task 3
time 13 ; task 3
time 14 : Idle
time 15 : Task 2
time 16 : Idle
time 17 : Idle
time 18 : Idle
time 19 : Idle