12/5/23

Write a program for Multilevel queue Scheduling algorithm. There must be 3 queues generated.

```
#include<stdio.h>
int spat[10], upat[10], n, n1, P1[10], P2[10];
int spp1[10], upp1[10], time=0, op=0
    y. z. pe
int sptat[10], uptat[10];
int spwt[10], upwt[10];
float spatat=0, spawt=0;
float upatat=0; upawt=0;
    void process(int x, int isystem)
    {
      if (isystem)
      {
        opt = spp1[x];
        sptat[x] = op - spat[x];
        spp1[x]=0;
        spwt[x] = sptat[x] - P1[x];
        spatat[e] = sptat[x];
        spwt + = spwt[x];
  }
else
{
    opt = upp1[x];
    uptat[x] = op - upat[x];
    upp1[x] = 0
    upwt[x] = uptat[x] - p.2[x];
```

```
    uptat = uptat [x];
    uptaut t = upat[x];
    }
  }

int main ()
{
  Printf ("Enter the no of system process : ");
  Scanf ("%d", &n);
 Printf ("Enter the no of user process");
  Scanf ("%d", &n2);
  Printf ("Enter AT of system process");
  for (i=0; i<n; i++)
    Scanf ("%d", &spat[i]);
  Pf ("Enter the AT for user process");
      for (i=0; i<n2; i++)
      Sf ("%d", &upat[i]);
  Pf ("Enter the Process time for user Process");
      for (i=0 ; i<n2; i++)
       Sf ("%d", &uppt[i]);
      for (i=0; i<n ; i++)

else
 { opt++;
  }
}
```

```
Printf("%d",oP);
Printf("\n");
 Printf("System process:\n");
   for(i=0;i<n;i++)
 Pf("SP %d %d %d \n",i++,sptat[i],spwt[i]);
 Pf("ATwt(System process): %2f\n",sptat/n);
 Pf("AWT system process): %2f\n",spwt/n);
  Pf("user process);
    for(i=0;i<n2;i++)
   Pf("up %d %d %d",i+1,uptat[i],upwt[i]);
   Pf("ATAT(user process): %2f\n",upwat/n2);
   Pf("AWT(user Process): %2f\n",upwt/n2);
    return 0;
 }
```

output

Enter the no of System process : 3
Enter the no of user Process : 3
Enter the AT of SP;
0 1 1
Enter the PT of SP;
5 3 4

Enter the AT of UP:

13 4

Enter the PT for UP:

5 3 2

System process:

SP 1	5	0
SP 2	7	4
SP 3	11	7

ATAT (System process): 7.67

AWT (System Process): 3.67

user process

UP 1	16	11
UP 2	17	14
UP 3	18	16

AIAT (user Process): 17.00

AWT (user Process): 17.67

EDF

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <string.h>
#include <limits.h>
#include <time.h>.

typedef struct
{
  char name[20];
  int deadline;
  int Priority;
  int start time;
  int end-time;
} task;

void sumtask (task to task)
{
  Printf ("Running %s", task->name);
  task -> Stagt - time = time (NULL);
  Sleep (1);
  task -> end-time = time (NULL);
}

void def - Schedules [Task task[]; int num-task
{
  int corred-time = 0;
    int i;
  while (true)
  {
```

```
int earliest_deadline = INT_MAX;

void print_gantt_chart(Task tasks[], int
                       num_tasks) {
Printf("\n Gantt chart : \n");
Printf("\n");
for (int i=0; i<num_tasks; i++)
  {
    Printf("| %-10s", tasks[i].name);
  }
  Printf("|\n|");
  for (int i=0; i<num_tasks; i++)
  {
    int task_duration = tasks[i].end_time - tasks[i].
    start_time);
    Printf("%-10d", task_duration);
  }

int main()
int num_task, i;
Printf("Enter the no of tasks: ");
Scanf("%d", &num_tasks);
Task tasks = (Task) malloc(num_tasks *
                    sizeof(Task));

for (i=0; i<num_tasks; i++){
  Printf("Enter task name: ");
  Scanf("%s", tasks[i].name);
```

```
Printf ("Enter task deadline: ");
Scanf ("%d", &tasks[i].deadline);
Printf ("Enter task Priority ");
Scanf ("%d", &tasks[i].Priority);
}
Edf-Schedule (tasks, num-tasks);
Print gantt-Chart (tasks, num-tasks);
Print tat_table (tasks, num tasks);
free (tasks);
return 0;
}
```

Output

Enter the no of tasks 2:

AT - 0

ET : 3

Deadline : 80.

Gantt Chart.

Time	task
0	task 1
20	completed
21	task 2
55	completed.