עבודה להגשה מס' 3

תאריך ההגשה: 28.04.2022 שעה 23:55

הנחיות:

- קראו היטב את השאלות.
- ניתן להגיש את העבודה בזוגות. לא ניתן להגיש את העבודה בקבוצה מעל שני אנשים.
- יש להגיש את העבודה בקובץ zip ובתוכו קצבי קוד C, קבצי header וקובץ readme.txt
- שם הקובץ שיוגש למערכת ההגשה יהיה מורכב מת"ז של המגיש/ים. לדוגמה:

עבור הגשה ביחיד - 111111111.zip

עבור הגשה בזוג - 111111111_22222222.zip

- במקרה של הגשה בזוגות, רק אחד מבני הזוג יגיש את העבודה במודל.
- גם בתחילת הקובץ בעבודה יש לרשום את שמות ומספרי ת.ז. של המגישים.
- איחור במועד ההגשה יגרור הורדה של ציון, 5 נק' לכל יום איחור או חלק ממנו.
  בכל מקרה לא יהיה ניתן להגיש מעבר ל-3 ימי איחור ממועד ההגשה המקורי.
  במקרים חריגים בלבד יש לפנות למרצה כדי לקבל אישור על הגשה באיחור.
- שאלות לגבי העבודה יש לשאול בפורום באתר הקורס ("מודל") או בשעות קבלה. **אין לשלוח שאלות שקרושות לעבודה במייל.**

You will write a program to implement the following CPU scheduling algorithms.

1. **First Come First Serve**
2. **Shortest Job First**
3. **Highest Response Ratio Next**
4. **Round Robin**
5. **Priority with Round Robin - multi level scheduling**
   There are several queues, each queue managed by RR. Each queue has absolute priority over the lower priority queue. No process can run until the high priority queues are empty.

The program reads a list of tasks from a file and then schedules them based on the chosen scheduling algorithm.

The program gets the file name from command line as an argument. For example:

```
annafr2@ubuntu:~/Desktop$ ./a.out test3.txt
```

Each line in the task file describes a single task. Each task has a name, a priority (for algorithms that need priority), arrival time, and a CPU burst separated by commas.

Priority, burst time, and arrival time are represented by an integer number. The higher the number, the higher the priority.

Example task file:

<task id>,<priority>,<task arrival time>,<burst time>

1, 4, 0,10

2, 3, 3, 8

3, 3, 2, 9

Function to implement:

1- Build (.....)

The function Build converts each line in the file to a struct called *task* that has the same fields (task id, priority …).

2- Table (....)

The functions returns an array of tasks, you could assume that the MAX length of tasks is equal to 10. (MAX is macro)

3- Display(....)

The function prints a table that shows (like in a practical session) the tasks in the file.

4- Schedule (....)

The function print the scheduling of an array of tasks, you need to use enum (DONT USE NUMBERS FOR ALGORTHIMS) for each algorithm name.

For example – using enum:

```
enum Algorithm{
    First_Come_First_Served=1 ,
    Shortest_Job_First=2 ,
    HRRN=3 ,
    Round_Robin=4 ,
    Priority_With_Round_Robin=5
};
```

The output should be:

<P<taskid>,CPU_TIME><P<taskid>,CPU_TIME>.....

Example: (FCFS, array_tasks)

Array_tasks contains 2 tasks:

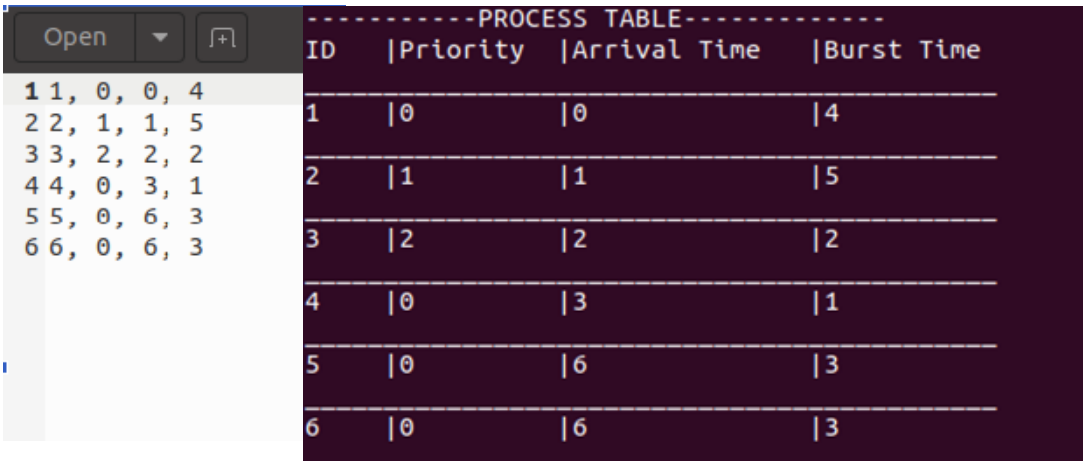
0, 4, 0,10

1,3,1,5

The output:

<P0,10><P1,5>

output example 2:



```
Scheduling Tasks - First Come First Served Algorithm:
<P1,4> <P2,5> <P3,2> <P4,1> <P5,3> <P6,3>

Scheduling Tasks - Shortest Job First Algorithm:
<P1,4> <P4,1> <P3,2> <P5,3> <P6,3> <P2,5>

Scheduling Tasks - HRRN Algorithm:
<P1,4> <P3,2> <P4,1> <P2,5> <P5,3> <P6,3>

Scheduling Tasks - Round Robin Algorithm:
<P1,2> <P2,2> <P3,2> <P1,2> <P4,1> <P2,2> <P5,2> <P6,2> <P2,1> <P5,1> <P6,1>

Scheduling Tasks - Priority With Round Robin Algorithm:
<P1,2> <P3,2> <P2,2> <P2,2> <P2,1> <P1,2> <P4,1> <P5,2> <P6,2> <P5,1> <P6,1>
```

סבודה נעימה