CS 3413

Assignment 3

Due Date: January 27th, 2020 at 12:30 pm

ASSIGNMENT IS TO BE COMPLETED INDIVIDUALLY BY ALL STUDENTS!

Your solution is to be written in C and submitted via D2L.

Let's improve our scheduling yet again! Computers today are being used in increasingly important applications where they need to act within a certain time frame in order to avoid an unpleasant outcome. A simple scheduling algorithm to achieve this is known as Earliest Deadline First (EDF). In EDF, each job gives the time that it must be completed by in order to meet a deadline. When considering what job to complete, you pick the job that has the earliest deadline. To break ties, you pick the shortest job first. If you still have a tie then you pick the job that arrived first. Does this sound familiar? It should … this is how most students decide what order to complete their assignments! For example, consider the following:

If we have 2 CPUs:

User	Process	Arrival	Duration	Deadline
Jim	A	2	19	24
Mary	B	2	21	24
Sue	C	5	1	8

At time 2, A and B will start to run on CPU1 and CPU2. When job C arrives at time 5 it will get a CPU right away as it had an earlier deadline along with A (since it has the same deadline as B, but wins the tie breaker because it is a shorter duration). After C is completed, then B gets a CPU again.

You are to add one more line to the program's output. After the summary of when people's jobs are completed you are to print the number of jobs that missed their deadline. That is jobs whose last execution was after the deadline that was specified. For example:

```
2 missed deadlines
```

In addition, let's expand our model on scheduling to now actually use threads for each CPU! Using pthreads, modify your program to create one pthread per cpu that you specify. Each thread should act as a cpu and remove a job from the run queue and then sleep()[1] for 1 second to represent 1 unit of execution time for the task. Once it is completed sleeping

[1] Look up "man sleep" to get a description of the sleep function.

then it will repeat and select the next task to execute for 1 second – which may be the same task! Be careful to protect your shared data by using locks to control access!

The output format remains as two tables. First table is the running time and the job(s) currently executing (tab separated). The second table is a summary with the user name (in the order in which jobs arrive) and the time when their **last** job is completed.