24.03.2017

BLG 312E – Computer Operating Systems

Homework 2

Submission Deadline: 13.04.2017, 23:55
(Late Submission Deadline: 14.04.2017, 23:55)

- You are expected to work individually on all exams and homeworks. All forms of collaboration are discouraged and will be treated as cheating. This includes actions such as, but not limited to, submitting the work of others as one's own (even if in part and even with modifications) and copy/pasting from other resources (including Internet resources) even when attributed. Serious offenses will be reported to the administration for disciplinary measures. All parties involved in the act will be treated equally.
- You have to achieve at least 20 out of 100 points on a homework for its submission to be accepted. Homeworks with lower grades will NOT be considered as submitted. Submitting parts of the codes provided in class will NOT be sufficient to achieve a grade of 20.
- Submission of the 3rd homework is compulsory for being allowed to take the final exam.
- Late submissions will be allowed for only 24 hours after the submission deadline. Regardless of the time of the late submission and its reason, **ALL** late submissions will be graded as 50% of the original grade the homework deserves. (Please note that in the case of late submissions, for a homework to be considered as submitted, its original (unreduced) grade should be at least 40).

What to submit: You should submit your source file(s) via the Ninova system. (No additional report file is required; however, it is expected that you include comments in your source file)

Program: Write and test a C program that implements the described behavior below:
In this homework, there will be one process with three threads.

Description: In a company the human resources (HR) department deals with hiring new employees. In this *department*, there is one *receptionist* and three *interviewers*. When an applicant arrives at the *department*, the *receptionist* registers him/her. After this step, the *receptionist* sends the applicant into the waiting room. Whenever one of the *interviewers* becomes available, he/she calls the first applicant in line and interviews him/her. While the *interviewers* interview the applicants, the *receptionist* continues registering incoming applicants. (Note: Assume that the waiting room has infinite capacity.)

You are required to model this HR *department* as a process with the *interviewers* and the *receptionist* as threads of this process. The *receptionist* needs `nr` seconds to register each applicant (this time is fixed and is read as a command line argument). Interviewing an applicant

takes *ni* seconds for an *interviewer* and *ni* depends on the job position the applicant is applying for. To simulate applicants with different types of position applications, an *interviewer* thread should read the time it will take to interview the current applicant from an input file. (**Hint!** To simulate the waiting times during the registration and interview stages, you can use the "sleep" command.)

Please note: For this homework, you are required to model the HR *department* as ONE process with FOUR threads (one *receptionist* thread and three *interviewer* threads). Solving the problem with multiple processes and no threads, will NOT get any points.

Please preserve the order and meaning of the program arguments.

Please test your program with different input files and make sure to achieve expected results.

Please check that your program correctly removes all allocated resources (e.g. shared memory locations, semaphores, and any others you have used).

Input parameters and input file formats: Input files for testing your program must be in the format given below where each line contains the *ni* interviewing times for each applicant. Input file must be in ".txt" file format. Assume that there are 4 applicants with following interviewing times. For example, interviewing time for the first applicant is 50 seconds, for the second one 30 seconds, so on.

Input.txt

```
50
30
70
60
..
```

Test: Your program will be tested in the form:

./program input.txt *nr*

Output format: Your program must print the events on the screen in their order of occurrence in the format given below. Assume that *nr = 1* seconds in this example.

```
Applicant 1 applied to the receptionist
Applicant 2 applied to the receptionist
Applicant 3 applied to the receptionist
Applicant 4 applied to the receptionist
```

```
Applicant 1’s registeration is done
Interviewer 1 started interview with Applicant 1
Applicant 2’s registeration is done
Applicant 3’s registeration is done
Interviewer 3 started interview with Applicant 2
Interviewer 2 started interview with Applicant 3
Applicant 4’s registeration is done
Interviewer 3 finished interview with Applicant 2
Interviewer 3 started interview with Applicant 4
Interviewer 1 finished interview with Applicant 1
Interviewer 2 finished interview with Applicant 3
Interviewer 3 finished interview with Applicant 4
All applicants have interviewed successfully.
```