Real Time Systems, January 2024

Independently run C programs using shared variables

Opened: Thursday, 25 January 2024, 12:00 AM
Due: Friday, 9 February 2024, 11:59 PM

✔ Done

Under the present assignment, two programs, say, **teacher.c** and **student.c** have to be written which will be compiled into two executable files, say, **teacher.out** and **student.out**, respectively.

At any point time, only one process will run the executable file teacher.out. However, there can be multiple (say, *n* number of) processes running student.out.

That is, there could be a **teacher** process (executing **teacher.out**) and *n* number of **student** processes (running **student.out**).

The **teacher** process is initiated by the command "**./teacher.out <n>**", where *n* is a +ve integer (eg, "**./teacher.out 107**") signifying the number of students in the class, and creates a shared array of structures of type **"struct att"** (defined below) of size *n* (say, **struct att rolls[107]** or **struct att *rolls**) which is shared by all *n student* processes.

struct att {
    int roll; /* The roll number of the student giving the attendance ***/**

    time_t tm; /* To keep the time when attendance was given by the student. Read the manual for **time()**, that is, "**man 2 time**" Also see the manual entries mentioned in the "**see also**" section of this manual page! ***/**
}

The teacher process initializes the structures of the **rolls** array by making the **roll** field to be -1. Each structure element of this array is used by a student to mark his/her attendance. A **student** process is initiated by the command "**./student.out <n>"**, where *n* is a +ve integer signifying the roll number of the student running this program (eg, **"./student.out 15**" signifies that it is being executed by the student having roll number 15**)**. A student process finds the **roll** field of its corresponding structure element in the **rolls[]** to be **-1**, and puts **his/her roll number** at that place. Additionally it puts the current time in the tm field of the structure.

When the **teacher process** is terminated by ^C (ctrl+c) it prints the roll number of all the students who have marked there attendance in the shared array rolls **in chronological order of the attendance**, along with the time (in a readable format like "January 31, 2023, 13:50") of attendance. It also destroys the shared memory segment and quits.

Please remember that there will be two different programs, namely, **teacher.c** and **student.c** which are to be executed to create the teacher process and the student processes. These programs are to be executed independently.

Your program(s) should make arrangement for releasing shared memory that has been crated during execution.

As enhancement of your program (**not to be submitted today**) you may try to support the following in your program.

1. One student process can access only its own element from the shared array
2. The teacher process may keep track of the details of the student processes that gave attendances (for example, for every attendance what was the corresponding student process id, the owner of that student process, anything you may conceive).
3. Modify your program by changing the call to the function **ftok()** to ...**ftok(".", ...)** and justify the outcome.

Submission status

Attempt number	This is attempt 1.
Submission status	Submitted for grading

Grading status	Not graded
Time remaining	Assignment was submitted 3 mins 28 secs early
Last modified	Friday, 9 February 2024, 11:55 PM
File submissions	student.c 9 February 2024, 11:55 PM teacher.c 9 February 2024, 11:55 PM
Submission comments	▸ Comments (0)