Programming Project 6

OS2025-SWU-软工中外 34 班

Spring 2025

Experimental topic

Programming Projects in Chapter 7

Page P-38 to P-38 of Operating System Concepts (10th) or this pdf

Experimental resources

/resources/code_for_project/code_for_project6

Experimental report naming format:

StudentID-name-Project-X (e.g., 2220183211-张三-Project-1)

Experimental objectives

1. (Course Objective 2) Master the basic concepts, design ideas, and status of the process. Understand the basic principles and ideas of process synchronization and mutually exclusion, and identify the key links, steps and constraints of engineering problems in this field. Use the basic means of realizing process synchronization and mutual exclusion to analyze and solve the teaching assistant problem (also known as the barber problem) in the field of process synchronization and mutual exclusion.

> **Attention**
>
> Please upload your report with naming format to the ftp server within **two weeks** (Deadline: 2025-05-13).

1 The Sleeping Teaching Assistant

A university computer science department has a teaching assistant (TA) who helps undergraduate students with their programming assignments during regular office hours. The TA's office is rather small and has room for only one desk with a chair and computer. There are three chairs in the hallway outside the office where students can sit and wait if the TA is currently helping another student. When there are no students who need help during office hours, the TA sits at the desk and takes a nap. If a student arrives during office hours and finds the TA sleeping, the student must awaken the TA to ask for help. If

a student arrives and finds the TA currently helping another student, the student sits on one of the chairs in the hallway and waits. If no chairs are available, the student will come back at a later time.

Using POSIX threads, mutex locks, and semaphores, implement a solution that coordinates the activities of the TA and the students. Details for this assignment are provided below.

1.1 The Students and the TA

Using Pthreads (Section 4.4.1), begin by creating n students. Each will run as a separate thread. The TA will run as a separate thread as well. Student threads will alternate between programming for a period of time and seeking help from the TA. If the TA is available, they will obtain help. Otherwise, they will either sit in a chair in the hallway or, if no chairs are available, will resume programming and will seek help at a later time. If a student arrives and notices that the TA is sleeping, the student must notify the TA using a semaphore. When the TA finishes helping a student, the TA must check to see if there are students waiting for help in the hallway. If so, the TA must help each of these students in turn. If no students are present, the TA may return to napping.

Perhaps the best option for simulating students programming—as well as the TA providing help to a student—is to have the appropriate threads sleep for a random period of time.

Coverage of POSIX mutex locks and semaphores is provided in Section 7.3. Consult that section for details.