CvP

Programming Assignment 3

Java Concurrent Programming

For this assignment a program, written in Java, must be compiled, run, studied and analyzed, and two derived programs have to be implemented.

The purpose of this assignment is to get some experience implementing concurrent programs in Java. The assignment is divided in two parts:

1. Analysis of a Java program which uses threads to implement a solution to a problem involving several beer drinkers and a toilet;
2. Implementation of two similar Java programs which use threads to implement the readers/writers problem.

1. Analysis of a Java Program with Threads

Copy the files of `BeerDrinkers.zip` to a directory and study the makefile to find out how you can compile the program and execute it. Analyze the Java source code of this program and find out how it implements a solution for the problem explained in the source file. There is no need to submit a report for this part of the assignment.

2. Implementation of Java Programs with Threads

Inspired by the program above, write two Java programs for the following problem:

The Readers / Writers Problem

Two kinds of processes – readers and writers – share a database. Writers execute transactions that update the database; reader transactions access the database without modifying it. The database is assumed initially to be in a consistent state (i.e. one in which relations between data are meaningful). Each transaction, if executed in isolation, transforms the 2 database from one consistent state to another. To preclude interference between transactions, a writer process must have exclusive access to the database. Assuming no writer is accessing the database, any number of readers may concurrently exe cute transactions.

First Program

Implement the Readers/Writers problem using monitors.

Second Program

Implement the Readers/Writers problem using monitors and giving priority to the writers; i.e., if a writer wants to enter the database but has to wait, then no reader may enter. Of course, the writer still has to wait for the current readers in the database to finish their transactions.

Monitors and Condition Variables in Java

Every object in Java with synchronized methods acts as a monitor. Unfortunately, Java does not allow the use of more than one condition variable per object. You can use the following synchronization methods:

- wait(): blocks the invoking thread;
- notify(): reactivates one thread blocked in a wait() ;
- notifyAll(): reactivates all threads blocked in a wait() .

Submission Instructions

Your programs should be submitted together with a makefile and a written report in which you explain your programs to Sander van Rijn svr003@gmail.com.

The report can only be submitted as plain text or PDF; other formats than `.txt` and `.pdf` will not be accepted. Remember to include your full name and student number in the body of the email.