MITx 6.004.3x
Computation Structures 3: Computer Organization

Help selfpoised

Course Progress Dates Discussion

Course / 19. Concurrency and Synchronization / Lecture Videos (36:48)

Previous Next

LE19.3

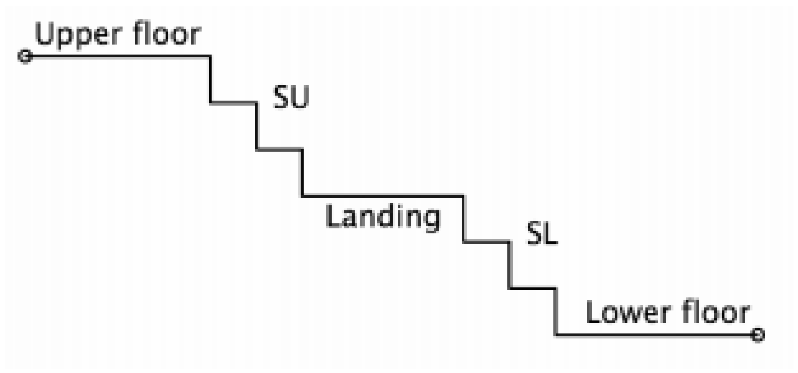
Bookmark this page

Calculator

LE19.3.1: Stairs!

0.0/1.0 point (ungraded)

The MIT Safety Office is worried about congestion on stairs and has decided to implement a semaphore-based trafficcontrol system. Most connections between floors have two flights of stairs with an intermediate landing (see figure). The constraints the Safety Office wishes to enforce are



- Only 1 person at a time on each flight of stairs
- A maximum of 3 persons on a landing
- As few traffic constraints as possible
- No deadlock (a particular concern if there's bidirectional travel)

Assume stair traffic is unidirectional: once on a flight of stairs, people continue up or down until they've reached their destination floor (no backing up!), although they may pause at the landing.

There are three semaphores: they control the upper flight of stairs (SU), the landing (L), and the lower flight of stairs (SL). **Please provide appropriate initial values for these semaphores and add the necessary wait() and signal() calls to the Down() and Up() procedures below.** Note that the Down() and Up() routines will be executed by many students simultaneously and the semaphores are the only way their code has of interacting with other instances of the Down() and Up() routines. Your code must avoid deadlock and enforce the stair and landing occupancy constraints. **Hint:** You may find it easier to first implement a solution where only 1 person at time is in-between floors (but be careful of deadlock here too!).

For each drop down, select the missing line of code. If a particular code region only requires one command, then select that command for the first drop down and select None for the second drop down. If no commands are needed in a region then select None for both answers.

// Semaphores shared by all students, provide initial values

semaphore SU = ;

semaphore SL = ;

semaphore L = ;

// code for going downstairs	// code for going upstairs
Down() {	Up() {
Select an option	Select an option
Select an option	Select an option
Enter SU;	Enter SL;
Select an option	Select an option
Select an option	Select an option

Calculator

Exit SU / enter landing;	Exit SL / enter landing;
Select an option	Select an option
Select an option	Select an option
Exit landing / enter SL;	Exit landing / enter SU;
Select an option	Select an option
Select an option	Select an option
Exit SL;	Exit SU;
Select an option	Select an option
Select an option	Select an option
}	}

Submit

Discussion

Hide Discussion

Topic: 19. Concurrency and Synchronization / LE19.3

Add a Post

Show all posts | by recent activity

There are no posts in this topic yet.

✖

‹ Previous

Next Up: Worked Examples ›
8 min + 1 activity

Affiliates
edX for Business
Open edX
Careers
News

Legal

Terms of Service & Honor Code
Privacy Policy
Accessibility Policy
Trademark Policy
Sitemap

Connect

Blog
Contact Us
Help Center
Media Kit
Donate

GET IT ON Google play
Download on the App Store

Calculator