

15-410, Fall 2009, Homework Assignment 1.
Due Monday, October 5, 20:59:59 p.m.

Please observe the non-standard submission time... As we intend to make solutions available on the web site immediately thereafter for exam-study purposes, please turn your solutions in on time.

Homework must be submitted in either PostScript or PDF format (not: Microsoft Word, Word Perfect, Apple Works, LaTeX, XyWriter, WordStar, etc.). Submit your answers by placing them in the appropriate hand-in directory, e.g., `/afs/cs.cmu.edu/academic/class/15410-s09-users/$USER/hw1/$USER.ps` or `/afs/cs.cmu.edu/academic/class/15410-s09-users/$USER/hw1/$USER.pdf`. A plain text file (`.text` or `.txt`) is also acceptable, though it must conform to Unix expectations, meaning lines of no more than 120 characters separated by newline characters (note that this is *not* the Windows convention). Please avoid creative filenames such as `hw1/my_15-410_homework.PdF`.

1 Financial crisis (4 pts.)

Consider simulated commerce in a situation such as SimCity. Imagine that some prominent citizens have been issued “special privilege cards” which entitle them to expedited service in stores. In particular, these citizens have “interrupt privileges”: they are not required to wait in line while non-privileged citizens complete their business (e.g., `Purchase()` or `Deliver()`). Instead, a privileged citizen can display her card, immediately perform her business, and leave the store. Privileged citizens can interrupt non-privileged citizens, but not each other.

Using the tabular trace format and the “e-commerce” sample code found in the lecture slides, show how the presence of a privileged citizen “interrupt” can cause deflation in the general economy. Please be sure that your trace is clear and conclusive (otherwise your answer will not receive full credit).

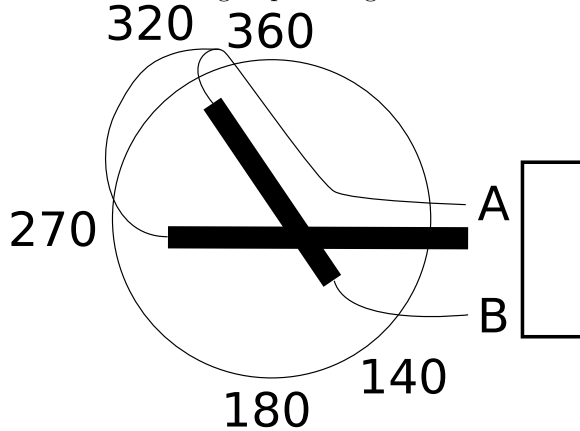
You may use more or fewer lines or columns than are provided in this sample table. You may change the column headings if you wish.

Execution Trace

time	Thread A	Thread B	Thread C
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

2 Runways (10 pts.)

Consider the following airport diagram.



The rectangle at the right is the airport terminal; A and B are taxiways, which airplanes use to travel between runways and the terminal; the two heavy lines which cross are two runways. The circle and numbers are not part of the actual airport, but are an important key to airport nomenclature.

According to standard airport runway nomenclature,¹ the horizontal line is known as runway 09/27 because it runs between the compass points at 90 degrees and 270 degrees; the other runway is known as 14/32 because it runs between 140 and 320 degrees. Depending on wind and traffic conditions, an airplane may be directed to take off from or land on a runway in either direction. So if a pilot is directed to take off from west to east (left to right on the horizontal-line runway on the diagram), the directive would be “cleared for takeoff on Runway 09,” which for the purposes of this problem we abbreviate as `takeoff(09)`.

At this airport, an airplane which executes `land(09)` and taxis to the end of the runway is already at the terminal, but an airplane which executes `land(32)` must then taxi along taxiway A (“`taxi_in(A)`”). Similarly, an airplane executing `takeoff(09)` must first have executed `taxi_out(A,09)` (note that taxiway A is connected to both the “32” end of runway 14/32 and the “27” end of runway 09/27).

It is not easy for large passenger aircraft to turn around in place, and many of them cannot “back up” (reverse) at all without the help of a pusher vehicle. Thus we will assume that airplanes cannot turn around or back up.

In this problem we will explore the ability of this airport to deadlock. Note that some situations which involve long delays are not deadlocks. For example, if the controller directs one flight to `taxi_out(A,14)` and another flight to `taxi_out(B,32)`, the airport is not deadlocked because eventually the winds will make it possible for the controller to direct either `takeoff(14)` or `takeoff(32)`. If a storm suddenly arrives at the airport, there may be a substantial delay before one flight or the other can take off, in which case there will be front-page news headlines and calls for Congress to pass a law. However, this does *not* constitute a deadlock.

2.1 5 pts

Explain a genuine deadlock situation which can arise. Show *both* a process/resource graph and a trace of directives issued by the hapless controller.

2.2 5 pts

Can you state a concise anti-deadlock rule which should be added to the training handbook for controllers working at this airport? Or can you otherwise characterize which patterns of orders the controller must not issue?

¹<http://www.bbc.co.uk/dna/h2g2/A810145>