Concurrent Programming

COMP 409, Fall 2010

Assignment 3

Due date: Monday, November 22, 2010
6pm

Assignment Questions

1. Suppose you are given the following code and data declarations. Assuming basic *happens-before* consistency, contrast the output values (variable contents) that could be justified in the following two cases. **10**

```
int a=b=x=y=z=0;
```

Thread 1	Thread 2
x = 2;	a=4;
x = 3;	a=x;
y = 1;	b=y;
y = b;	b=6;
x = 5;	
z = a;	

(a)

```
int a=b=x=y=z=0;
```

Thread 1	Thread 2
x = 2;	a=4;
x = 3;	a=x;
lock m;	lock m;
y = 1;	b=y;
y = b;	unlock m;
unlock m;	b=6;
x = 5;	
z = a;	

(b)

Programming Questions

You may use Java, or PThreads (C/C++) for the following implementations. In all cases your code must be in a professional style: **well-commented,** properly structured, and appropriate symbol names. Marks will be very generously deducted if not! All programs should include demonstrative, but not excessive, output. Your programs should not have race conditions and should maximize the ability of threads to execute concurrently.

2. A simple computer game allows game characters to move around a 2D space discretized into a grid, with each character or obstacle fully occupying a grid cell, and each character moving atomically to adjacent cells.

   Given a destination, characters search for the shortest path to their destination and attempt to follow that plan. Once they get to their destination, or if they cannot reach their destination at some point, (perhaps because another character has moved into the way) they pause a turn, and then choose a new destination and calculate a new movement plan.

Model each character as a thread, initially distributed around the perimeter of an $m \times m$ grid, and given random destinations. Each character moves discretely, at 20ms intervals, although movement rates otherwise just depend on thread scheduling. Obstacle cells are never chosen for destinations, and paths are calculated with Dijkstra's or any other convenient shortest-path algorithm, ignoring the current position of other characters.

Provide 3 different obstacle maps for your simulation. The first is empty, no obstacles. The second has a random r cells set to obstacles. The third should contain a spiral of obstacles, with r random gaps, as shown below. Note that obstacles are never within two squares of the perimeter.

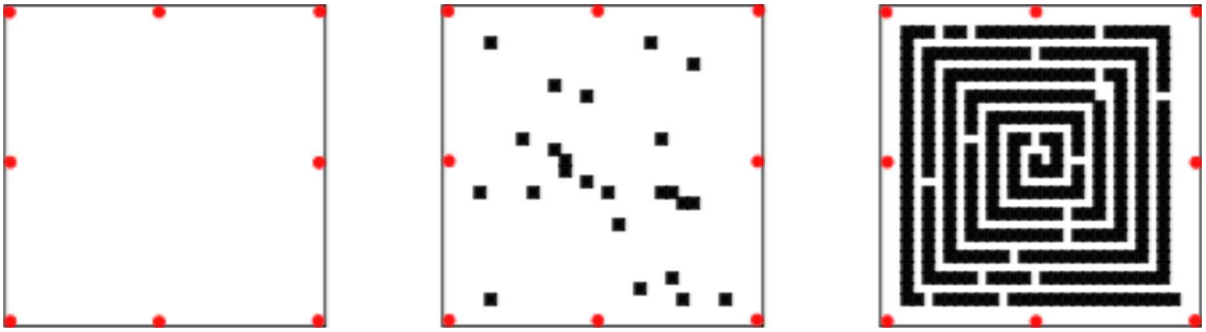

Figure 1: The three maps. Red dots show ($n = 8$ here) perimeter-distributed characters.

Your application should take $n\ m\ k$ as parameters, in that order and construct a simulation of n characters on an $m \times m$ grid, for map $k \in \{1, 2, 3\}$. A visual depiction is not required, although you may find it helpful for debugging.

It should always be possible for characters at least 2 squares away from each other to move concurrently, and movement must be atomic. Explain why your simulation does not deadlock.

Try your simulation for $m = 20$ and $m = 100$, and values of n and r that produce sparsely populated and densely populated situations. Let each simulation run for 2 minutes. Once the simulation has ended report the total number of moves made by each character. **20**

3. Modify your solution to the above so the simulation ends as soon as any character successfully reaches 10 destinations in a row. At that point the simulation should end as promptly as possible, and emit the move number data. **10**

What to hand in

For assignment submission you will use `moodle`:

`http://moodle.cs.mcgill.ca/moodle`

If you did not use this for assignment 1 be sure that you login and familiarize yourself with the system before the deadline. Clock accuracy varies, and late assignments will not be accepted without a medical note: you have ample time, **do not wait until the last minute** to do the assignment! Assignments must be submitted on the due date **before 6pm**.

Where possible hand in only **source code** files containing code you write. Do not submit compiled binaries or .class files. For the written answer questions submit either an ASCII text document or a .pdf file *with all fonts embedded*. Do not submit .doc or .docx files.

Note that for written answers you must show all intermediate work to receive full marks.

This assignment is worth 10% of your final grade. **40**