COMP124 Individual Coursework

Assignment 2

**NOTE:** This is the second COMP124 practical exercise that is to be formally assessed.

It will contribute up to 10% to the overall mark of the module.

You will probably find it beneficial to study and implement the thread-based producer-consumer code given in lectures before attempting the following.

**Description**

Craftsperson A produces ceramic pots at the rate of one every 5 minutes. Craftsperson B produces similar pots at the slightly slower rate of one every 6 minutes. All pots are placed on a shelf of limited size (5 pots), from where they are taken by a packer and packed into cartons at the rate of one every 4 minutes. You should assume that the potters are unable to produce any pots when the shelf is full, and that the packer is unable to operate when the shelf is empty.

Write a thread-based Java program to simulate the above scenario. You will need threads for each of the two potters, plus a thread for the packer. Your program should continue until each potter has produced 10 pots and the packer has packed all 20 pots. Its output should be a running commentary on the actions of each of the workers (see below).

**Hint**

To simulate time passing, use the Thread.sleep() call, which takes as its argument the number of milliseconds that the thread should sleep. You should speed up time by replacing each minute by 100 milliseconds. Exceptions should be caught when sleep() is called; e.g. your run() method for Potter A should contain the following code at the point a pot is being made:

try  
{ Thread.sleep(500);  
} catch (InterruptedException e) {}

Similarly, the other potter and the packer should sleep for 600 ms and 400 ms, respectively.

**Program output**

Your program should produce something along the following lines (although feel free to tailor it):

Potter 1 (Harry) has started

Potter 2 (Beatrix) has started

The Packer (Macca) has started

Macca is ready to pack

Shelf is empty. Waiting to remove . . .

Harry has made a pot

Inserting pot. There are now 1 pots on the shelf

Harry has put his pot on the shelf

Removing pot. There are now 0 pots on the shelf

Beatrix has made a pot

Inserting pot. There are now 1 pots on the shelf

Beatrix has put her pot on the shelf

Macca has packed a pot

Macca is ready to pack

Removing pot. There are now 0 pots on the shelf

Harry has made a pot

Inserting pot. There are now 1 pots on the shelf

Harry has put his pot on the shelf

Beatrix has made a pot

Inserting pot. There are now 2 pots on the shelf

Beatrix has put her pot on the shelf

Macca has packed a pot

Macca is ready to pack

Removing pot. There are now 1 pots on the shelf

Harry has made a pot

Inserting pot. There are now 2 pots on the shelf

Harry has put his pot on the shelf

Macca has packed a pot

Macca is ready to pack

Removing pot. There are now 1 pots on the shelf

Beatrix has made a pot

Inserting pot. There are now 2 pots on the shelf

Beatrix has put her pot on the shelf

Harry has made a pot

Inserting pot. There are now 3 pots on the shelf

Harry has put his pot on the shelf

etc.

**What to submit**

For this assignment you are not asked to submit a report. You should submit a **single** .java file containing all your Java code. At the head of your program, include your name, student ID and University email address.

**How to submit**

The work must be submitted electronically by using the Department’s e-submission system.

This must be done by

**16.00 on** **Friday March 27**

at the latest.

**Marking**

Your submission will be marked according to the following criteria:

Programming (correctness, efficiency, quality of output, etc.): 80%

Style (comments, identifiers, layout, etc.): 20%

Students submitting on time and including the correct email address in their work will receive detailed feedback within three working weeks of the deadline given above.

HAPPY PARALLEL POTTING!