COMP124 Assessed 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.

In the previous lab session you were asked to code up the producer-consumer problem described in lectures. You should now be ready to write your own thread-based Java program.

PART A
The Computators of Vrymnos have been given a sacred duty to perform. On a wall in the Temple of Numerological Heaven is a row of small cubicles. The cubicles are numbered 1 to 18. Within each cubicle is a stone tablet upon which is inscribed a number in the range 1 to 99.

Six Apprentices have been instructed to do the following. Each must consult the Oracle, which will select one of the cubicles at random. The Apprentice must go to that cubicle and look inside. If there is a tablet there, he must remove it and add the number it holds to his own running total. If the cubicle is empty, he must keep moving to the next cubicle to his right, looping back to the first cubicle if necessary, until he finds one which is full. He then uses that tablet as before.

When an Apprentice has successfully retrieved and added up the numbers contained on three tablets, he must announce his sum.

Your task:
The file Compute1.java contains a program to implement the situation described above. You will see that it declares an array of six threads to represent the Apprentices. Please run the program and study it to make sure you understand how it works.

PART B
Volumina, the High Priestess of the Temple, has decided that her Apprentices have been given responsibilities beyond their lowly status. She declares that henceforth each Apprentice must not be allowed to proclaim his sum. Instead, he must inscribe it on a Parchment of Partial Enlightenment, which he then attempts to place into the Box of Digital Delights. The Box is capable of holding a maximum of only two parchments at a time.

Volumina is the only person permitted to remove tablets from the Box. When she has all six, she adds them up and announces the Grand Total.

Your task:
Alter the given code to implement the new scenario. You will need to create an additional thread to represent Volumina. You will also need to declare an object to represent the Box. Take care over simultaneous access to this object, and watch out for full/empty conditions.

Your program should work with any values on the tablets, but in the version you submit, make sure the tablets are initialised in the way I have done in my code (i.e. the tablet in cubicle 1 is inscribed with 1, etc.). Using this data, the Grand Total announced by Volumina should be 171.

The output from your program should take the form of a running commentary on the activity taking place.

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

**** THE DEADLINE FOR THIS ASSIGNMENT IS 16.00 ON THURSDAY APRIL 4 ****

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

MARKING SCHEME

Compilation and execution (15%)

Does the program compile successfully?
Does it execute without crashing or hanging?
(Note that marks for this will only be awarded if the code is a reasonable attempt at the problem, and not just my code or any old Java program!)

Programming approach (50%)

Does the program do what is required of it?
Have suitable techniques been used in the implementation of the program?
Has thought been given to the choice of programming constructs used?
Is the code simple and efficient?

Output (15%)

Is the correct result produced?
Does the output give a good running commentary on the concurrent activity?

Style (20%)

Is the program suitably annotated with comments?
Have meaningful variable names and identifiers been used?
Has a helpful program layout been used?