CS 420
HOMEWORK ASSIGNMENT H1

1. Simple Multi-Threaded programming - 1.

Write a Java application that creates three new threads from a class that is a subclass of class `java.lang.Thread`. One should print a finite sequence of positive integers prefaced with the character 'A' (such as A4), another prints a finite sequence of positive integers prefaced with the character 'B' (such as B369) and the third thread should print a finite sequence of positive integers prefaced with the character 'C' (such as C12). All sequences should be relatively long -- 4,000 numbers or more.

So, for instance, the first thread would print:
A1
A2
A3
A4
…
…
..
and keeps looping until it prints at least 4,000 of these lines – so the last few lines would look like:
..
..
A3998
A3999
A4000

The second thread puts the character "B" in front of each number and the third thread puts the character "C" in front of each number – and they each also print a sequence of 4,000 lines.

No thread should explicitly yield to another thread while it is executing. Be sure to use the `System.out.println()` function to assure that each line of output is printed immediately after execution.

Run the program four or five times to see how the output is interleaved between the threads. Explain the sequencing of the output.

Also create a MS-DOS batch file named `prob1.bat` *that will execute the application*. *Make sure the last line in the batch file is the* `pause` *command*. Store all files necessary for the execution of your application in a subfolder named `Prob1` of the folder that you submit electronically for the assignment, as instructed below.

2. Simple Multi-Threaded programming - 2.

Modify your solution for exercise 1 to use tasks represented by objects that implement the `java.lang.Runnable` interface, rather than being a subclass of `Thread`.

Run the program four or five times to see how the output is interleaved between the threads. Explain the sequencing of the output. Are the programs you have created for the first two problems functionally equivalent?

Also create a MS-DOS batch file named `prob2.bat` *that will execute the application*. *Make sure the last line in the batch file is the* `pause` *command*. Store all files necessary for the execution of your application in a subfolder named `Prob2` of the folder that you submit electronically for the assignment, as instructed below.

3. Simple Multi-Threaded programming - 3.

Modify your solution for exercise 2 so that each thread explicitly yields to another thread after printing ten numbers. Use the static `yield`() method from class `Thread`.

Run the program to see how the output is interleaved between the threads. Explain the sequencing of the output. Does the use of the `yield()` have any effect on the program execution?

Also create a MS-DOS batch file named `prob3.bat` *that will execute the application*. *Make sure the last line in the batch file is the* `pause` *command*. Store all files necessary for the execution of your application in a subfolder named `Prob3` of the folder that you submit electronically for the assignment, as instructed below.

The following questions required your independent reading from the book or online resources. Type your solutions in Microsoft WORD.

4. Explain the difference between "multiprogramming" and "multiprocessing (multiple CPUs)". What were the reasons for the development of each?

5. Research the concept of "least privilege" as it applies to security and explain how it is related to the concepts of user mode, monitor mode and privileged instructions.

6. Explain in your own words how DMA and "cycle stealing" increase computer performance.

7. What are two significant advantages of a distributed operating system controlling a set of loosely coupled computers? What is the significant difficulty in designing and implementing a distributed operating system?

8. How did the development of interactive computing (time-shared systems), that was discussed in the evolution of OS, contribute to software programmer productivity?

General Instructions:

- First three problems count for 50% of the score.
- Homework submissions must be prepared using computer document preparation applications such a word processor or similar editor.
- Homework submissions will be clearly marked with the student's name, date and assignment identification at the top of the first page.
- All homework is to be completed by each student individually and represent that student's original, unassisted work. Any material copied in any way from other sources must be clearly identified and attributed.
- The non-programming problem solutions **are printed on paper and submitted at the start of class on the due date.**

Programming Instructions:

- Put a block of comments at the beginning of every physical file containing program source code that includes your name, the course name and number, information identifying what functions the program is designed to perform, and instructions how to execute the program. (required)
- For the programming problems, place ONLY the Java source files into a zip file in the Canvas course site using the assignment submission capability. You can also add comments when you submit a file for the assignment.
- Each assignment must be submitted on Canvas by the start of class on the day the assignment is due.
- Submit only one zip file with your entire assignment.
- If you have already submitted a homework assignment and then decide you must resubmit the assignment before it is due, you can submit another zip file to replace previous submissions. You can also use comments with the submission to further explain your submission to the grader. You may resubmit as many times as you find necessary before the assignment due date.

- **DO NOT FORGET TO ALSO SUBMIT THE PRINTED NON-PROGRAMMING PROBLEM SOLUTIONS AND JAVA SOURCE CODE AT THE START OF CLASS ON THE DUE DATE.**