Homework 4

1. (20 pts) For the following program, explain the interesting elements related to threads. Focus on explaining the output of the program.

```
1  public class TaskThreadDemo {
2   public static void main (String args []) {
3    String [] sa = {"a", "X", "+", "."};
4    for (String s: sa) {
5     Runnable ps = new PrintChar (s, 200);
6     Thread ts = new Thread (ps, s);
7     ts.start ();
8    } // end for each character
9   } // end main
10 } // end class TaskThreadDemo
11
12 class PrintChar implements Runnable {
13  String ch;
14  int times;
15
16  public PrintChar (String c, int n) {
17   ch = c;
18   times = n;
19  } // end constructor
20
21  public void run () {
22   for (int i = 0; i < times; i++) {
23    System.out.print (ch);
24   } // end for loop
25  } // end method run
26 } // end class PrintChar
```

2. (20 pts) What is changed if the method called on line 7, start(), is replaced with run()? Explain (of course). Focus on explaining the output of the program.

3. (20 pts) What is changed if the method Thread.yield() is added between lines 23 and 24? Explain. Focus on explaining the output of the program.

4. (20 pts) Using the jconsole or jvisualvm utilities provided in the JDK, list and explain some of the threads that are created in your code for Project 3. Note that you can name the threads created in the program, as is done on line 6 in Problem 1 above, which can make this discussion a lot easier to follow.

5. (20 pts) Explain how the java.util.concurrent.Semaphore class might be used in Project 4 to coordinate the requirements of the various jobs. Then address the question of whether or not this actually makes sense in the context of the requirements of program. In other words, can you suggest approaches to handling shared resource pools that would be simpler than using semaphores?

Grading Rubric:

Attribute	Meets	Does not meet
Problem 1	**20 points** Explains the interesting elements related to threads. Focuses on explaining the output of the program.	**0 points** Does not explain the interesting elements related to threads. Does not focus on explaining the output of the program.
Problem 2	**20 points** Explains what is changed if the method called on line 7, start(), is replaced with run().Focuses on explaining the output of the program.	**0 points** Does not explain what is changed if the method called on line 7, start(), is replaced with run(). Does not focus on explaining the output of the program.
Problem 3	**20 points** Explains what is changed if the method Thread.yield() is added between lines 23 and 24. Focuses on explaining the output of the program.	**0 points** Does not explain what is changed if the method Thread.yield() is added between lines 23 and 24. Does not focus on explaining the output of the program.
Problem 4	**20 points** Lists and explains the threads that are created in your code for Project 3.	**0 points** Does not list or explain the threads that are created in your code for Project 3.
Problem 5	**20 points** Explains how the java.util.concurrent.Semaphore might class be used in Project 4 to coordinate the requirements of the various jobs. Addresses the question of whether or not this actually makes sense in the context of the requirements of program. Suggests approaches to handling shared resource pools that would be simpler than using semaphores.	**0 points** Does not explain how the java.util.concurrent.Semaphore might class be used in in Project 4 to coordinate the requirements of the various jobs. Does not address the question of whether or not this actually makes sense in the context of the requirements of program. Does not suggest approaches to handling shared resource pools that would be simpler than using semaphores.