You may bring one page of notes, front and back, and a calculator.

Questions will be in short-answer format with partial credit for partial answers.

You will be asked to both read and write Java code, but only write code using primitive types (including arrays of primitive types) and System.out.println().

Topics:

- All midterm topics
- Array lists and linked lists, hash maps: complexity of add, remove, and access operations
- Stacks, queues (including priority), and sets: Basic properties and typical operations
- Linked lists: Internal structure, implementation of add/remove from beginning/end
- Generic classes: purpose, defining a generic class, instantiating a generic object
- Runnable interface: run() method, executing with an ExecutorService
- Thread synchronization: LinkedBlockingQueue, synchronized keyword
- Networks: Purpose of ServerSocket and Socket classes, blocking calls, TCP versus UDP

Sample questions:

1. [5] What is the Big-O complexity of a linked list access operation (i.e., get an item at an arbitrary index)? Why?


3. [5] Briefly describe how a set differs from a list:

4. [10] Consider the following Java code:

```java
public synchronized void print(int start, int end) {
    for (int i = start; i < end; ++i)
        System.out.println(i);
}

public void printNumbersThreaded() {
    ExecutorService executor = Executors.newCachedThreadPool();
    executor.execute(new Runnable() { public void run() { print(0, 10); }});
    executor.execute(new Runnable() { public void run() { print(10, 20); }});
}
```

a. (10) What (if anything) can we guarantee about the order in which the numbers will be printed?

5. [5] Briefly describe the purpose of the ServerSocket accept() method in terms of what it returns and when it returns: