Name **SOLUTION** ID _____ Final Score 10
Extra Credit _____

Section (circle one): MW 3:30-6:30

CSCI 201L Written Exam #2
Summer 2016
10% of course grade

1. Anonymous Inner Classes – In lecture we walked through the following:

1. Having two classes in different file.
2. Having two classes in the same file.
3. Having one class with the second class inside of it (an inner class).
4. Having one class with the second class inside of one of the methods.
5. Having one class with the second class inside the parameter call to a method with no name (anonymous inner class).

a. Give two reasons why a programmer would choose to create an anonymous inner class (#5 above) instead of an inner class inside of a method (#4 above). **(0.5% + 0.5%)**

Reason #1

If no other code in the method needs to use the inner class, we want to hide the class so other code is not able to call it.

Reason #2

Since the class will be declared in line, it will be more readable since it is closer to the method that is using it.

b. Give two reasons why a programmer would choose to create a class in a different file (#1 above) instead of a class in the same file as another class (#2 above). **(0.5% + 0.5%)**

Reason #1

If the class needs to be public so it is acceptable outside of the current package, it would have to be declared in its own file.

Reason #2

For code organization, having classes in their own files makes it easier to find in a project.

2. **JDBC** – JDBC uses reflection to help facilitate Java's motto of "write once, run anywhere.
   a. Explain what reflection is. **(0.5%)**

   Reflection allows a class to be dynamically loaded at runtime from a string.

   b. Even though Java tried to create its JDBC framework so that no code has to be changed when you change from one database to another, the implementation unfortunately does not allow this. Explain why. **(0.5%)**

   The SQL implementations provided by different database vendors are different. That means that every SQL statement may need to be changed even though no Java code may need to be changed.

3. **Networking Theory** – You have started a small web hosting company and you need to get the network set up with many Internet-facing computers. Since you have taken CSCI 201, you know that the best way to do this is to have a static IP address on each of those servers. Assume that you need 27 IP addresses. When you call your ISP, there is a UCLA alumnus working there. You tell him what you want, and he says, "One of your IP addresses is 215.76.194.57," but he doesn't tell you anything else. Because you went to USC, you can hopefully answer the following questions though.

IP – `1101 0111  0100 1100  1100 0010  0011 1001`

   a. What is the network address? Provide this in the dotted IP notation, not in binary. **(0.5%)**

   215.76.194.0

   b. So that you are given the fewest number of IP addresses for your desired purpose, what is the subnet mask? Provide this in the dotted IP notation AND slash notation, not in binary. **(0.5% + 0.5%)**

   255.255.255.224 (0.5%)
   215.76.194.57/27

   c. What is the network/subnet address combination? **(0.5%)**

   215.76.194.32

4. Databases and SQL – Answer the following questions concerning the database below.

Here is the `Book` table.

bookID	title	author	isbn	numCopies
1	Tonight on the Titanic	Mary Pope Osborne	978-0-606-16894-6	3
2	Afternoon on the Amazon	Mary Pope Osborne	978-0-679-86372-9	1
3	Balto of the Blue Dawn	Mary Pope Osborne	978-0-553-51085-0	2
4	Happy Birthday, Bad Kitty	Nick Bruel	978-0-545-29863-6	2
5	Bad Kitty Does Not Like Candy	Nick Bruel	978-1-62672-230-9	1
6	Bad Kitty Drawn to Trouble	Nick Bruel	978-1-62672-117-3	2

Here is the `User` table.

userID	username
1	jimmy
2	joannie
3	johnny
4	jenny

Here is the `CheckedOut` table.

checkedOutID	bookID	userID	numCheckedOut
1	3	4	1
2	3	3	1
3	1	1	2
4	2	2	1

a. Write the SQL query that returns the following table. **(1.0%)**

title	username	numCheckedOut
Afternoon on the Amazon	joannie	1
Balto of the Blue Dawn	johnny	1
Balto of the Blue Dawn	jenny	1
Tonight on the Titanic	jimmy	2

```
SELECT b.title, u.username, co.numCheckedOut
  FROM Book b, User u, CheckedOut co
  WHERE b.bookID=co.bookID
    AND u.userID=co.userID
  ORDER BY b.title asc;
```

5. Locks and Conditions – Answer the following questions about locks and conditions.

a. The type of lock we used in Java was called a `ReentrantLock`. Explain what a `ReentrantLock` is and how it is different from a lock that is no reentrant. **(0.5%)**

With a ReentrantLock, after a thread has acquired a lock, it will be able to enter other sections of code that require the same lock without having to acquire it again. Otherwise, this would cause deadlock.

b. Describe why a thread must obtain a lock before calling a method on a condition associated with the lock. **(0.5%)**

When the await() method on a condition is called, the lock associated with that condition is released. Before being able to signal threads that are waiting on a condition, the lock must be obtained because it doesn't make sense to signal on a condition associated with a lock without having the lock. Why would one thread be able to signal another thread waiting on a condition when it doesn't have anything to do with that condition or lock?

Variations on the above are fine.

6. **Multithreading –** Give three rules that will always be true about the output of the following program. **(0.5% + 0.5% + 0.5%)**

```
import java.util.ArrayList;
import java.util.concurrent.Semaphore;

public class Problem6 extends Thread {
   public static ArrayList<Integer> al = new ArrayList<Integer>();
   public static Semaphore sem = new Semaphore(2);
   private int num;
   public Problem6(int num) {
         this.num = num;
   }
   public void run() {
         try {
               sem.acquire();
               System.out.println(num + " starting ");
               for (int i=0; i < al.size(); i++) {
                     System.out.println(al.get(i));
               }
         } catch(InterruptedException ie) {

         } finally {
               System.out.println(num + " ending ");
               sem.release();
         }
   }
   public static void main(String[] args) {
         for (int i=0; i < 5; i++) {
               al.add(i);
         }
         for (int i=0; i < 100; i++) {
               Problem6 p6 = new Problem6(i);
               P6.start();
         }
   }
}
```

1. **"1 starting" will always get printed before "1 ending".**
2. **The values of al will always be printed in order, though there may be other numbers in the middle.**
3. **Only two threads will have printed "starting" before there must be an "ending".**

Other rules may be acceptable if true.

7. **Monitors and Locks –** When Java uses the **synchronized** keyword, it is utilizing monitor functionality. There is also lock functionality in Java through the `Lock` interface.

   a. Is the functionality behind a monitor the same as that through an explicit `Lock` variable? **(1.0%)**

A monitor consists of a lock, but we do not have access to conditions on the underlying lock.

   b. Why would a programmer choose to use a Lock instead of a monitor? **(0.5%).**

A programmer would choose to use a Lock when Conditions are needed.

If they say that a programmer could also choose to use a Lock for personal preference, give 0.25%.

Extra Credit Question

Extra credit is applied after the curve so does not affect other students.

8. **Extra Credit –** We covered six major topics in this class. Rank these six topics in order of what you thought was most useful to least useful. Provide one sentence explaining why you thought the most useful topic was the most useful and one sentence explaining why you thought the least useful topic was the least useful. The topics were: Java porting from C++, GUIs, Software Engineering, Networking, Databases, and Concurrent Programming. **(0.25% + 0.25%)**

#1 (Most Useful) –
Explanation –

#2 –

#3 –

#4 –

#5 –

#6 (Least Useful) –
Explanation –