Name **SOLUTION**	ID	Final Score /15 Extra Credit /0.5

Lecture Section (circle one): TTh 8:00-9:20 TTh 9:30-10:50 TTh 11:00-12:20

CSCI 201L Written Exam #1
Fall 2017
15% of course grade

The exam is one hour and 50 minutes and is closed book, closed note with one 8.5"x11" double-sided paper of ***hand-written*** *notes allowed.*

1. **Exception Handling** – Java, unlike C++, has both checked and unchecked exceptions. Explain why unchecked exceptions in Java do not need to be handled by a corresponding `try` block. **(1.0%)**

   Unchecked exceptions can be avoided through good programming. Handling unchecked exceptions through exception handling instead of good programming practice is much slower if the exception is thrown.

2. **Server-Side vs Client-Side** – Form validation can be done in JavaScript, but it is good coding practice to validate in a back-end language instead (or in addition to). Give one reason why. **(1.0%)**

   Users are able to disable JavaScript in the browser, meaning that the form validation in JavaScript could be circumvented. Back-end validation will always happen though and can't be avoided by users.

3. **Garbage Collection** – Why are we not able to invoke the garbage collector but merely make a request that the garbage collector run sooner? **(1.0%)**

   The garbage collector is a low-priority thread, so we are not able to have that thread preempt the currently-executing thread. We are able to increase the priority of the garbage collector, which will hopefully enable it to run sooner.

4. Multi-Threading – Answer the questions below based on the following program.

```
1  import java.util.concurrent.*; // to save space
2  public class Question4 extends Thread {
3     private int num;
4     public Question4(int num) {
5           this.num = num;
6           if (num == 1) {
7                 this.setPriority(Thread.MAX_PRIORITY);
8           }
9     }
10    public void run() {
11          for (char letter='a'; letter < 'f'; letter++) {
12                System.out.print("" + letter + num + " ");
13                if (letter == 'd') {
14                      Thread.yield();
15                }
16                if (letter == 'e') {
17                      System.out.println();
18                }
19          }
20    }
21    public static void main(String [] args) {
22          ExecutorService executors = Executors.newCachedThreadPool();
23          for (int i=0; i < 3; i++) {
24                executors.execute(new Question4(i));
25          }
26    }
27 }
```

a. Give two possible outputs. **(0.5% + 0.5%)**

<u>Output #1</u>

```
a0 a1 b1 c1 d1 a2 b2 c2 d2 e2 e1
b0 c0 d0
e0
```

<u>Output #2</u>

```
a0 b0 c0 d0 a1 b1 c1 d1 e1
a2 b2 c2 d2 e0
e2
```

Other outputs may also be correct, as long as the letters associated with a number are in order and there is a new line in the output somewhere after the "e" value prints (though possibly not immediately after).

b. What impact on the output would removing line 7 have? **(0.5%)**

Removing line 7 would potentially not have any impact on the output. It possibly would allow threads other than #1 to execute sooner, but since priorities are based on probabilities, it may not have any impact.

5. **Polymorphism** – Does the following code compile? If so, what is the output? If not, correct the error(s) so that it does compile. (NOTE: You are not allowed to comment or remove any lines.) **(0.5% + 1.0%)**

```java
public class Question4 extends A implements B {
    public void bar() {
        System.out.println("bar 2");
    }
    public void foo() {
        System.out.println("foo 2");
    }
    public static void main(String [] args) {
        A a = new Question4();
        a.bar();
        a.foo();
        a = new A();
        a.bar();
        a.foo();
        B b = new Question4();
        b.bar();
        b.foo();
        C c = new A();
        c.foo();
    }
}
class A implements B {
    public void bar() {
        System.out.println("bar 1");
    }
    public void foo() {
        System.out.println("foo 1");
    }
}
interface B extends C {
    public void bar();
}
interface C {
    public void foo();
}
```

Yes, the code compiles. The output is:

```
bar 2
foo 2
bar 1
foo 1
bar 2
foo 2
foo 1
```

6. **Serialization –** We talked about using serialization with file I/O. Give two advantages that serializing an object to a file has over writing code to save all of the member variables of a class to a file. **(0.5% + 0.5%)**

<u>Reason #1</u>
It saves the programmer time of not needing to parse the data when reading it back in from the file.

<u>Reason #2</u>
If the class inherits from another class, all of the member variables in the parent classes will also be serialized.

7. **Software Engineering –** Agile methodologies were developed long after plan-based methodologies. Give two principles that make agile-based methodologies more accepted by programmers. **(0.5% + 0.5%)**

<u>Reason #1</u>
Individuals and interactions are valued.

<u>Reason #2</u>
Working software is valued over documentation.

<u>Reason #3</u>
Collaborating with customers is valued over contracts.

<u>Reason #4</u>
Responding to change is valued over following a plan.

8. **HTML –** HTML forms can be submitted as a GET or a POST. Explain the difference and show an example of the URL after submitting the form for both based on the following form. **(0.5% + 0.5% + 0.5% + 0.5%)**

```
<form action="submit.jsp" method="<GET or POST>">
  <input type="text" name="user" value="dtrump" />
  <input type="password" name="pw" value="america" />
  <input type="submit" name="submit" value="Validate" />
</form>
```

GET Explanation
When a form is submitted with the GET method, the data from the form is displayed in the URL, allowing client-side code to access it.

GET URL
http://<hostname>/submit.jsp?user=dtrump&pw=america&submit=Validate

---

POST Explanation
When a form is submitted with the POST method, the data from the form is sent to the server but not returned to the client.

POST URL
http://<hostname>/submit.jsp

9. **JavaScript and AJAX –** The third parameter to the `open` function on the `XMLHttpRequest` object is a `boolean` value representing whether the request will be made synchronously (false) or asynchronously (true). Explain what synchronous and asynchronous calls are. **(0.5% + 0.5%)**

Synchronous Call
With a synchronous (blocking) call, no other code will execute until the synchronous call returns.

Asynchronous Call
With an asynchronous (non-blocking) call, the client JavaScript code will continue to execute, but there will be a callback function that gets called after the asynchronous call returns.

10. AJAX – Since we are able to submit a form to another page to dynamically load content, why did AJAX come into existence? What additional feature does AJAX provide that we didn't have before? **(0.5%)**

AJAX allows just a portion of a page to be updated instead of forcing a reload on the entire page.

11. CSS – Using the `font-family: verdana` CSS attribute, write the CSS code for the following: **(0.5% + 0.5% + 0.5%)**

<u>The font of the entire page will be Verdana</u>
body { font-family: verdana; }

<u>The font of any `h1` tag with an id of "`verdana`" will be Verdana.</u>
h1#verdana { font-family: verdana; }

<u>The font of any `h6` tag nested inside of a paragraph tag will be Verdana.</u>
p h6 { font-family: verdana; }

12. Servlets and JSPs – Since every JSP is compiled to a servlet, why do servlets still exist? Give two reasons to use servlets instead of (or in addition to) JSPs. **(0.5% + 0.5%)**

<u>Reason #1</u>
Servlets allow HTML code to be embedded in Java, so if there is a lot of Java logic, servlets will be more convenient than JSPs.

<u>Reason #2</u>
Servlets can redirect to a JSP and pass data to them.

<u>Reason #3</u>
Servlets are just Java classes, so they may be more familiar to programmers than JSPs.

13. Networking – The constructor for the `Socket` class has the potential of throwing an `IOException`. Give two situations when instantiating a `Socket` will throw an `IOException`. **(0.25% + 0.25%)**

Reason #1
When the server specified by the "hostname" cannot be reached.

Reason #2
When there is no application listening to the specified "port" on the server.

14. Networking – Since we ran out of IPv4 addresses many years ago, solutions had to be developed to still allow computers to communicate on the Internet. Give two solutions. **(0.25% + 0.25%)**

Solution #1
IPv6 increased the number of addresses.

Solution #2
Network Address Translation (NAT) allowed computers within a network to have private IP addresses, which can be repeated in different networks.

Solution #3
Dynamic Host Control Protocol (DHCP) allowed IP addresses to be reused by a different computer once a computer is no longer using it on the network.

Other answers may be acceptable.

Extra Credit Question

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

15. How much experience did you have with each of the following topics **before** starting CSCI 201? Place one "X" in each row. **(0.5%)**

Topics	No Experience	Some Experience	Substantial Experience
Java	*34.6%*	*45.8%*	*19.6%*
HTML	*29.9%*	*54.2%*	*15.9%*
CSS	*41.1%*	*45.8%*	*13.1%*
JavaScript	*57.4%*	*30.6%*	*12.0%*
AJAX	*84.1%*	*10.3%*	*5.6%*
JSP	*97.2%*	*2.8%*	*0%*
Servlet	*99.1%*	*0.9%*	*0%*
Multi-Threading	*75.7%*	*21.5%*	*2.8%*
Network Programming	*82.2%*	*15.0%*	*2.8%*

For any topic where you didn't have substantial experience, did you feel as if you were at a disadvantage over other students in the class? Explain.

Disadvantage – *40.7%*
Advantage – *59.3%*