

CSSE 220—Object-Oriented Software Development

Exam 2, Oct. 19, 2016

This exam consists of two parts. Part 1 is to be solved on these pages. There is an additional blank page at the end of Part 1 if you need more room. Part 2 is to be solved using your computer, and will be taken on Friday. You will need network access to download template code and upload your solution for part 2.

Resources for Part 1: You may use a single sheet of $8\frac{1}{2} \times 11$ inch paper with notes on both sides. Your computer *must be closed* the entire time you are completing Part 1.

Resources for Part 2: This portion is open book, notes, and computer but with limited network access. You may use the network only to access your own files, the course Moodle site and web pages, Piazza (though do not post/respond to questions), the textbook's site, Oracle's Java website, and Logan Library's Safari Tech Books Online. Any communication with anyone other than the instructor or a TA during the exam may result in a failing grade for the course.

When you have finished Part 1, please turn it in and then wait quietly for the programming exam review portion of class to begin.

	Problem	Poss. Pts.	Earned
	1	5	_____
	2	10	_____
	3	9	_____
	4	4	_____
	5	16	_____
	6	12	_____
Paper Part Subtotal		56	_____
C1. Recursion problems		21	_____
C2. Polymorphism problem		6	_____
C3. GUI problem		17	_____
Computer Part Subtotal		44	_____
Total		100	_____

Part 1—Paper Part

1. (5 points)

The following sentences each describe the design of a different object oriented system. Label each of them with one of the following phrases that best match (*you'll use each phrase exactly once*): High Coupling, Low Coupling, High Cohesion, Low Cohesion, and None (for the description that isn't really describing coupling or cohesion).

_____ You notice a class named ImageOrDate which, depending on how you initialize it, can handle image manipulation or computing the number of days between dates.

_____ Rather than using a method to avoid duplication, a programmer cuts and pastes the same code in 4 parts of one class.

_____ The DataCalculator class in your system does not depend on any other classes.

_____ Every other class in the system has a field of type GUIHandler, which handles all input and output with the user.

_____ You take a large class and split it into two smaller classes, each of which does one conceptual thing.

2. (10 points) This problem is a design exercise. First, read the problem description below. Then answer the questions.

Imagine an auction system that auctions a small number of items each day. Each item auctioned has an id number and a brief description. Throughout the day, users can stop by and view the items available today and decide to bid a specific amount for each item they want to bid on; they can't see what others have bid. Users are identified by a name and unique bidder number. Then, at the end of the day the owner enters the "end auction" command into the system. The user with the highest bid for that item wins the auction. Then, the next day, the users can come by, see the items they won, and pay for their bids.

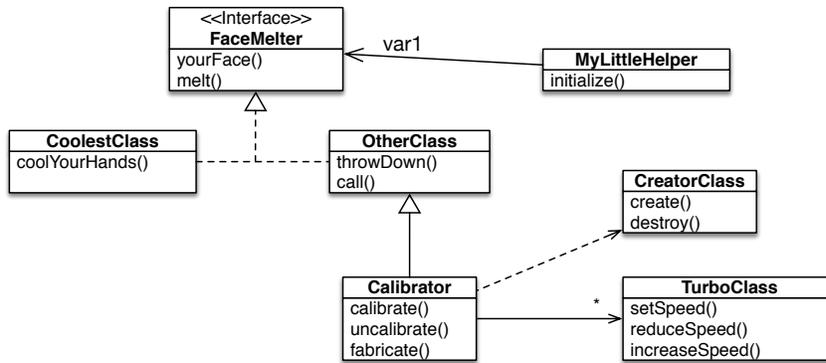
Don't worry about a GUI for this problem – assume the employees access the system using a very simple web-based system built entirely in the Main class.

- a. (5 points) Draw a UML class diagram showing how you would design this system. You do not need to include every method or field - just the important ones. It should be clear from your diagram how all the data mentioned above is stored, and it should follow the OO principles we've discussed in class.

Be sure to include the class that contains main, you can assume this class also includes methods: `handleBid`, `handleEndAuction`, `getItemsWonByUser`.

- b. (5 points) Describe in a few sentences what happens when the `handleEndAuction` command is executed. Make it clear what methods call what other methods. Reference what you wrote in your UML class diagram as much as possible.

3. (9 points) Use this UML class diagram to answer the subsequent questions.



a. (5 points) Circle true or false for each of the following statements.

- T F OtherClass objects must have a yourFace() method AND a melt() method
- T F MyLittleHelper objects must have a yourFace() method AND a melt() method
- T F CoolestClass is a subclass of Object
- T F The diagram implies that multiple Calibrator objects can point at the same TurboClass object
- T F If you have a Calibrator object c, this code will compile: c.melt();

b. (2 points) Which of these could explain the line between Calibrator and CreatorClass?

- (a) Calibrator has a field of type CreatorClass
- (b) CreatorClass has a field of type Calibrator
- (c) The method calibrate() in Calibrator uses a local variable of type CreatorClass
- (d) The method create() in CreatorClass uses a local variable of type Calibrator
- (e) None of the above could explain the relationship

c. (2 points) If you added a method foo() to OtherClass, what other classes must change? (If no other classes need to change, write "none")

4. (4 points) Consider the following code.

```
public static void innerFunction(int val) throws IllegalStateException {
    System.out.println("before");
    if(val == 0)
        throw new IllegalStateException("Zero is bad");
    System.out.println("after");
}

public static void outerFunction() throws NullPointerException {
    try {
        innerFunction(0);
        System.out.println("after2");
    } catch (IllegalStateException e) {
        System.out.println("catch1");
    }
}

public static void main(String[] args) {

    try {
        outerFunction();
        System.out.println("after3");
    } catch (NullPointerException e) {
        System.out.println("catch2");
    }
}
```

What does this code print?

5. (16 points) Consider the following related declarations:

```
public abstract class A {
    public abstract void one();
    public void two() {
        System.out.println("A2");
    }
}
public class B extends A{

    public void one() {
        System.out.println("B1");
    }
    public void two() {
        System.out.println("B2");
    }
    public void four() {
        two();
        System.out.println("B4");
    }
}
```

```
public class C extends B{
    private ArrayList<D> forwarders;
    public C() {
        forwarders = new ArrayList<D>();
        forwarders.add(new D());
        forwarders.add(new D());
    }
    public void three() {
        System.out.println("C3");
    }
    public void two() {
        System.out.println("C2");
    }
}
public class D {
    public void two() {
        System.out.println("D2");
    }
}
```

- a. (4 points) Draw a UML diagram to represent the given interface and classes. Include all methods, but when writing subclass methods, only show a method on the subclass if the subclass method overrides the parent class's method, or if the method is specific only to the subclass:

- b. (12 points) Continuing the same problem, suppose we declare and initialize these variables:

```
A ac = new C();
B bb = new B();
B bc = new C();
```

For each line of code below, if the line results in an error, **circle** the appropriate error; otherwise, provide the output in the provided blank. If the code works but does not print anything, write “nothing”. Consider each line of code separately. That is, if a line would give an error, then assume that line doesn’t affect any others. If the result would print on multiple lines, remove the newline from your result and show it on a single line.

Code	Either circle the error or provide the output	
bc.three();	<i>runtime error</i>	<i>compile error</i> _____
bc.two();	<i>runtime error</i>	<i>compile error</i> _____
ac.two();	<i>runtime error</i>	<i>compile error</i> _____
((A) ac).two();	<i>runtime error</i>	<i>compile error</i> _____
((C) ac).three();	<i>runtime error</i>	<i>compile error</i> _____
((A) bc).three();	<i>runtime error</i>	<i>compile error</i> _____
((D) bc).two();	<i>runtime error</i>	<i>compile error</i> _____
C cc = bc;	<i>runtime error</i>	<i>compile error</i> _____
A ab = bb;	<i>runtime error</i>	<i>compile error</i> _____
A aa = new A();	<i>runtime error</i>	<i>compile error</i> _____
ac.one();	<i>runtime error</i>	<i>compile error</i> _____
bc.four();	<i>runtime error</i>	<i>compile error</i> _____

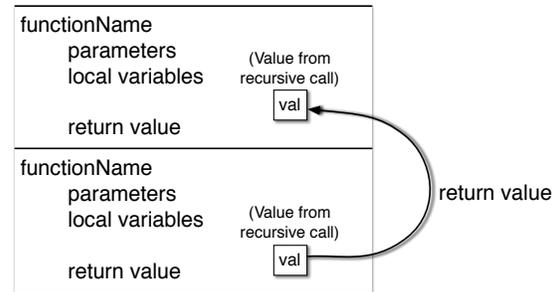
6. (12 points) For this problem, use the frame technique we practiced in the course to trace the execution of the recursive function call. Start your trace with the first call to mystery on line 14. A frame template is provided for your reference.

Once you are finished, answer the question at the bottom of the page.

```

1 public class Foo {
2     public static String mystery(String s) {
3         if (s.length() <= 2) {
4             return "x";
5         }
6         char ch = s.charAt(0);
7         if (ch == 'a') {
8             return mystery(s.substring(2)) + ch + ch;
9         } else {
10            return mystery(s.substring(2)) + ch;
11        }
12    }
13
14    public static void main(String[] args) {
15        System.out.println(mystery("earth"));
16    }
17 }

```



For the code above, *what would the final output be?* _____

Use this page for additional workspace if you need it.