

Name: KEY Section: _____ CM: _____

CSSE 220—Object-Oriented Software Development

Exam 2, October 18th, 2017

This exam consists of two parts. Part 1 is to be solved on these pages. 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. You can also use your "UML Cheatsheet" and your "Design Principles" handouts if you brought them. Your computer *must be closed* the entire time you are completing Part 1.

	Problem	Poss. Pts.	Earned
	1	5	_____
	2	10	_____
	3	10	_____
	4	4	_____
	5	12	_____
	6	15	_____
	Paper Part Subtotal	56	_____
	C1. Recursion problems	21	_____
	C2. Polymorphism problem	6	_____
	C3. GUI problem	17	_____
	Computer Part Subtotal	44	_____
	Total	100	_____

This page is left intentionally blank.

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).

high cohesion You create a new class called EmployeeIDNumber because there are actually a variety of operations that have to do with ID numbers that would complicate the Employee class.

low coupling The Logger class has a NetworkFileSystem field, but because there are not getters/setters for this field, no other class in the system depends on NetworkFileSystem.

none Looking at a class, you notice the same method - with minor modifications - has been duplicated several times.

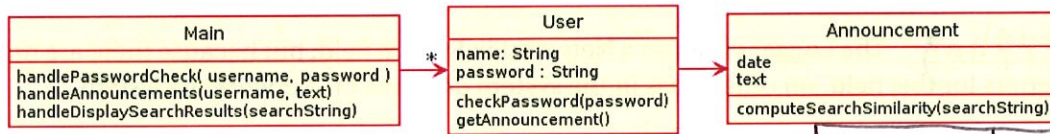
low cohesion A class has two kinds of methods - methods that have to do with reading files and methods that have to do with matrix transforms.

high coupling In order to compute an order's tax rate, the DBUpdater class needs to call 8 different getter methods on the Order class.

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

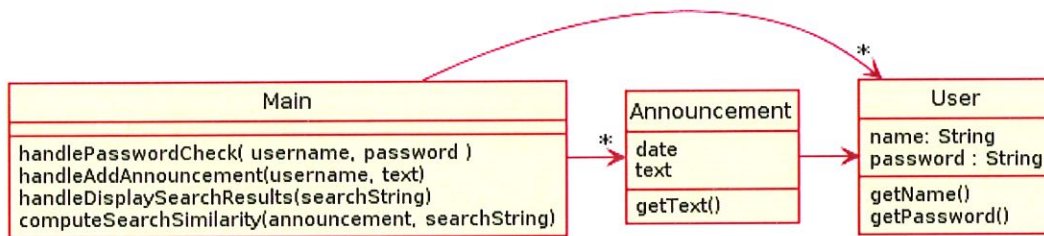
On a particular website, users log on with a username and password. Once logged on, they can make announcements that have their username, the current date and some text associated with them. The website also has a feature where you can search for announcements. Given a search string and announcement, an algorithm can compute a similarity rank in the range of 0-100. The results are then sorted in similarity rank order.

a. (3 points) Explain the problems with this solution using your design principles.



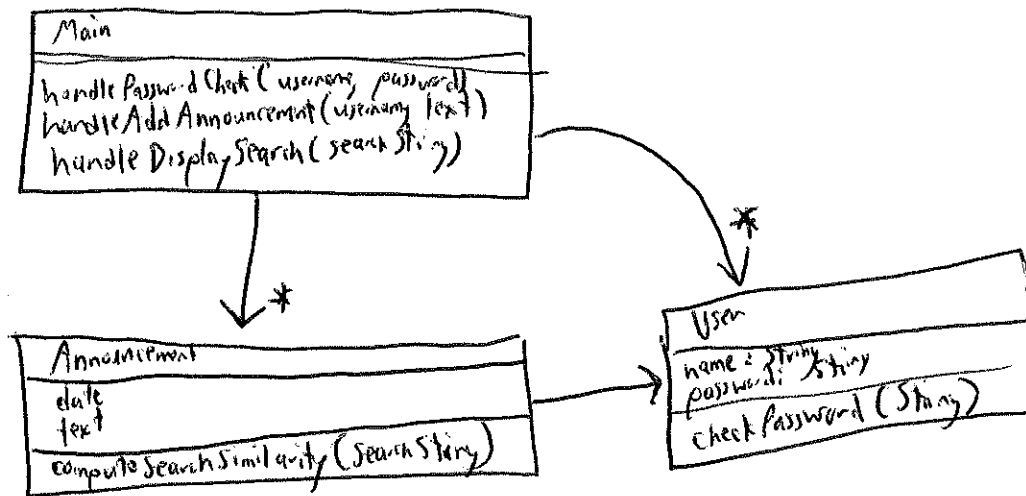
2a. main must search through users → to find announcements (minor if at all)
 2b. each user can only have one announcement
 minor lib message to compute similarity

b. (3 points) Explain the problems with this solution using your design principles.



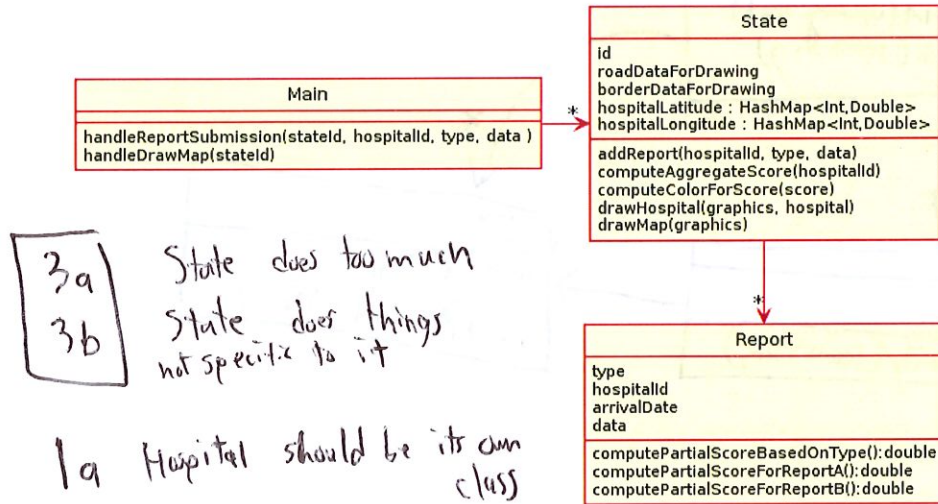
3a. announcement should compute its own similarity
 3b. user should handle password check
 minor

c. (4 points) Make a UML diagram of your proposed solution to the problem.

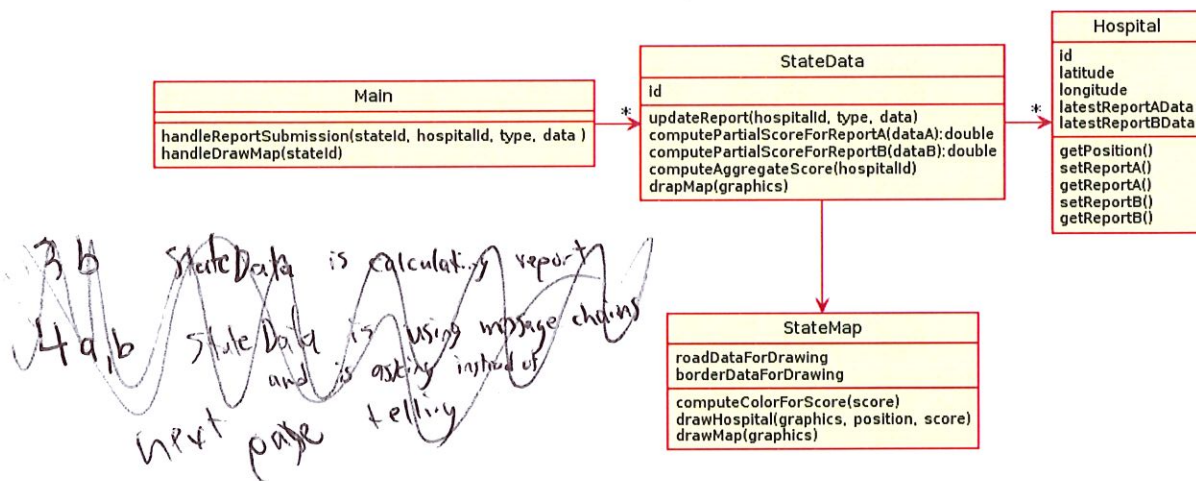


3. (10 points) A government agency tracks hospital data to make a monthly report. The highlight of the report is a set of colorful maps that show every hospital in a particular state with colors indicating the hospital's overall status. To make this report, hospitals submit two different kinds of reports (reportA, reportB) that come in at different times. Each of these reports is analyzed differently and produces a partial score. The final map color is produced by combining these three scores into an aggregate score, using the most up-to-date version of each report available at that time. Color on the map is positioned according to the hospital's specific latitude and longitude which is stored in the system and does not change. The drawn map also includes the states borders and major roads.

a. (3 points) Explain the problems with this solution using your design principles.



b. (3 points) Explain the problems with this solution using your design principles.

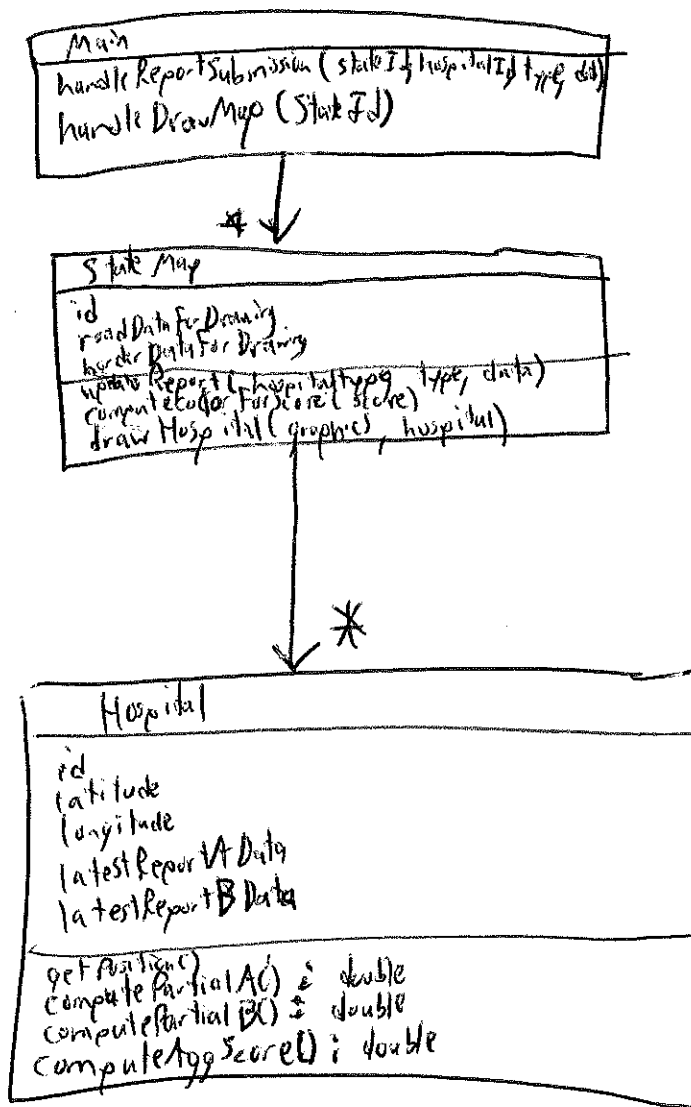


(write your answer on the next page)

(space to answer 3b)

36 state data is calculating for hospital's data
4 a, b state data is using message chains, asking instead of telling

c. (4 points) Make a UML diagram of your proposed solution to the problem.



4. (4 points) Consider the following code.

```
public static int fun(int x) {
    int result = 10;
    System.out.println("fun x = " + x);
    try {
        result = otherFun(x);
        result = otherFun(x+10);
    } catch (IllegalArgumentException e) {
        System.out.println("Fun caught");
        result = 0;
    }
    return result;
}

public static int otherFun(int x) throws IllegalArgumentException {
    if (x < 10) {
        throw new IllegalArgumentException();
    }
    System.out.println("other returns " + (x * 2));
    return x*2;
}

public static void main(String[] args) {
    try {
        System.out.println(fun(5) + " is fun(5)");
        System.out.println();
        System.out.println(fun(12) + " is fun(12)");
        System.out.println("done.");
    } catch (IllegalArgumentException e) {
        System.out.println("Main caught");
    }
}
```

What would this code print?

fun x = 5
Fun caught
0 is fun(5)
fun x = 12
other returns 24
other returns 44
44 is fun(12)
done

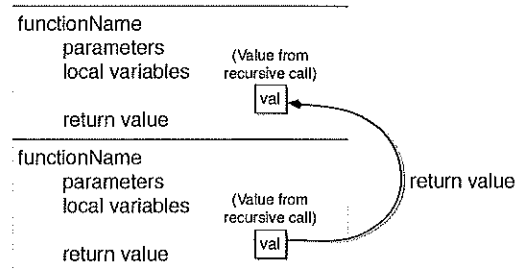
5. (12 points) For this problem, use the frame technique we practiced in the course to trace the execution of the recursive function call. Start your trace with the first call to mystery on line 14. A frame template is provided for your reference.

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

```

1  public static int mystery(int x) {
2      if(x <= 0) {
3          return 0;
4      }
5      boolean isEven = (x % 2 == 0);
6      if(isEven) {
7          return mystery(x/2) * 2;
8      } else {
9          return mystery(x-5) + 5;
10     }
11 }
12
13 public static void main(String[] args) {
14     System.out.println(mystery(11));
15 }

```



mystery
~~x~~ = 11
x <= 0? no
isEven = false
isEven? no
return 10 + 5

mystery
x = 6
x <= 0? no
isEven = true
isEven? yes
return 5 * 2

mystery
x = 3
x <= 0? no
isEven = false
isEven? no
return 0 + 5

mystery
x = -2
x <= 0? yes
return 0

For the code above, what would the final output be? 15

6. (15 points) Consider the following related declarations:

```
public class Spanish {
    int x;
    public void method1() {
        System.out.println("uno");
    }

    public void method2() {
        System.out.println("dos");
    }
}

public class English extends Spanish {
    Qqqqq y;
    public void method2() {
        System.out.println("two");
    }

    public void method3() {
        System.out.println("three");
    }
}
```

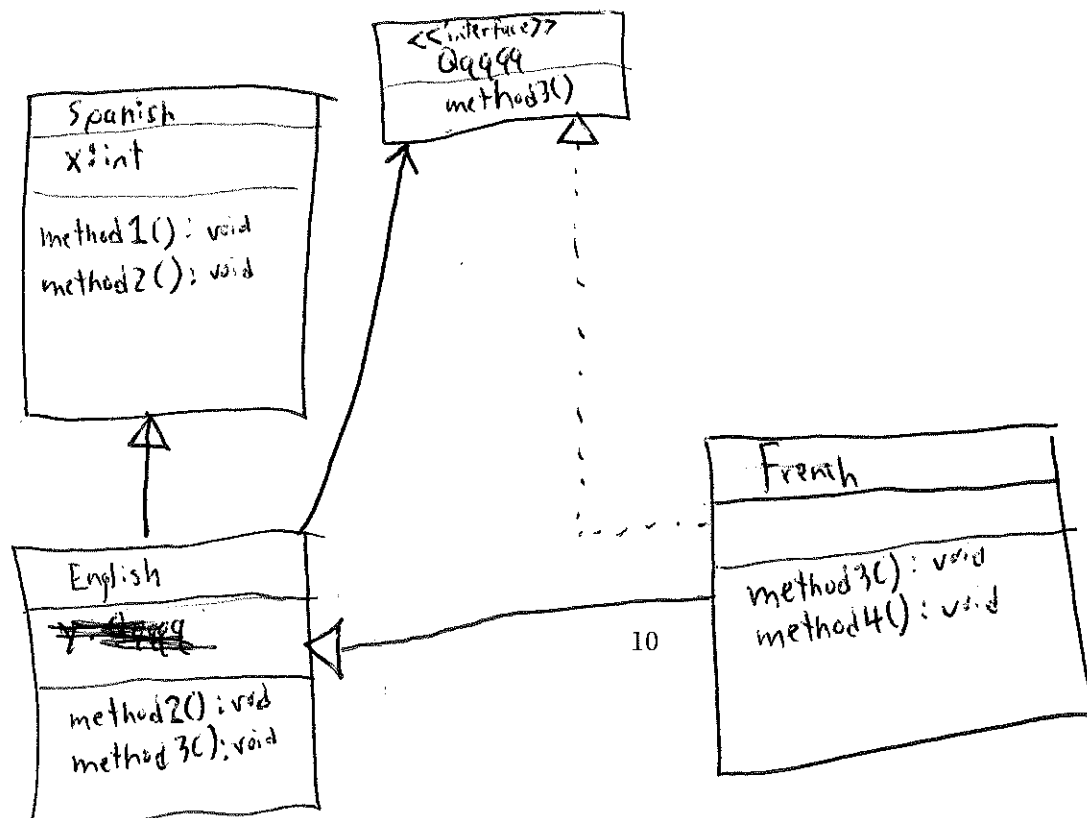
```
public class French extends English
implements Qqqqq {

    public void method3() {
        System.out.println("trois");
    }

    public void method4() {
        System.out.println("quatre");
    }
}

public interface Qqqqq {
    void method3();
}
```

- 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. In places where lines representing fields are appropriate, use lines and do NOT re-list the same field in the field list.



- b. (11 points) Continuing the same problem, suppose we declare and initialize these variables:

```
English myE = new English();
Spanish myS = new English();
French myF = new French();
Spanish myS2 = new French();
Qqqqq myQ = new French();
```

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		
myE.method1();	runtime error	compile error	<u>uno</u>
myS.method2();	runtime error	compile error	<u>two</u>
myS.method3();	runtime error	<u>compile error</u>	
myQ.method3();	runtime error	compile error	<u>trois</u>
myQ.method4();	runtime error	<u>compile error</u>	
myF.method1();	runtime error	compile error	<u>uno</u>
((English) myS).method3();	runtime error	compile error	<u>three</u>
((English) myS2).method3();	runtime error	compile error	<u>trois</u>
((English) myF).method3();	runtime error	compile error	<u>trois</u>
((French) myS).method3();	<u>runtime error</u>	compile error	
((Qqqqq) myE).method3();	<u>runtime error</u>	compile error	<u>nothing</u>

