Name: ____________________________________ Section: ____________

CSSE 220—Object-Oriented Software Development

Exam 2 – Part 2, January 26th, 2018

Allowed Resources on Part 2. Open book, open notes, and computer. Limited network access. You may use the network only to access your own files, the course Moodle and Piazza sites (but obviously don't post on Piazza) and web pages, the textbook's site, Oracle's Java website, and Logan Library's online books. **You may only use a search engine (like Google) to search within Oracle's Java website - all others uses or website other than those mentioned above are not allowed.**

Instructions. *You must disable Microsoft Lync, IM, email, and other such communication programs before beginning part 2 of the exam. Any communication with anyone other than the instructor or a TA during the exam may result in a failing grade for the course.*

You must actually get these problems working on your computer. Almost all of the credit for the problems will be for code that actually works. There are several different small methods to write, so you can get a lot of partial credit by getting some of them to work. If you get every part working, comments are not required. If you do not get a method to work, comments may help me to understand enough so we can give you (possibly a small amount of) partial credit.

Begin part 2 by checking out the project named *Exam2-201820* from your course SVN repository. (Ask for help immediately if you are unable to do this.)

When you have finished a problem, and more frequently if you wish, **submit your code by committing it to your SVN repository**. We will check commit logs, so you must be careful not to commit anything after the end of the exam. For grading, we will ensure that the included JUnit tests have not been changed.

Be sure to turn in these instructions, with your name written above, to your exam proctor. You should not exit the examination room with these instructions.

Honesty pledge.

I understand that I may not communicate in any way with anyone other than the instructor and their assistants or use any non-approved resources during the exam.

I understand that after the exam, I will not communicate anything about the exam to any student that has not already taken the exam.

I understand that if I violate either of the above, that the penalty is at least a -100% on this whole exam, and that I may be expelled from Rose-Hulman.

If you understand these and agree to abide by them, then check here: ____

Otherwise, check here and talk to your professor privately soon after the exam: ____

Your signature: ______________________________

Problem Descriptions

Part C1: Recursion Problems (18 points)

The class Recursion in the recursion package contains 4 recursion problems (test cases are also included). *You only need to solve 3 of the 4 problems.* Leave the problem you chose to skip blank and leave a comment saying that you skipped it. These problems must be solved with recursion - **a working solution with loops is worth no credit**. If you have time and want to do a fourth one for fun, that's fine, but we suggest saving it until you finish the rest of the exam.

Part C2: Polymorphism Problem (12 points)

The zookeeper is still feeding her pets. But now, pets also have special abilities that they show off. You are given a working solution to a Pet program. Unfortunately this code has duplication everywhere: PetMain, Zookeeper *and* in the various pet classes (Cat, Dog, Fish). Your job is to use inheritance and/or interfaces to remove as much code duplication as you can. Make any other small changes you need to the code to make it work with your modified classes. For full credit, structure your code so that you only implement methods that will be used.

Your changes should not affect the functionality of the solution (it should keep giving the same output as it does now),as follows:

```
Mary is feeding Tiger. Cat Tiger is eating food.
Mary is feeding Misty. Cat Misty is eating food.
Mary is feeding Tiny. Dog Tiny is eating food.
Mary is feeding Bubbles. Fish Bubbles is eating food.
Mary is feeding Comet. Fish Comet is eating food.
Mary is feeding Flash. Fish Flash is eating food.
There are 6 pets eating
Tiger says: Yawn. Zzz...
Smokey says: Yawn. Zzz...
Misty says: Yawn. Zzz...
Spot says: Bow wow!
Tiny says: Bow wow!
Athena says: Arf!
Bubbles says, time to move!
Wiggle 1
Wiggle 2
Wiggle 3
Comet says, time to move!
Wiggle 1
Flash says, time to move!
Wiggle 1
Wiggle 2
Wiggle 3
Wiggle 4
Wiggle 5
```

Part C3: Return of the Roomba Cat (20 points)

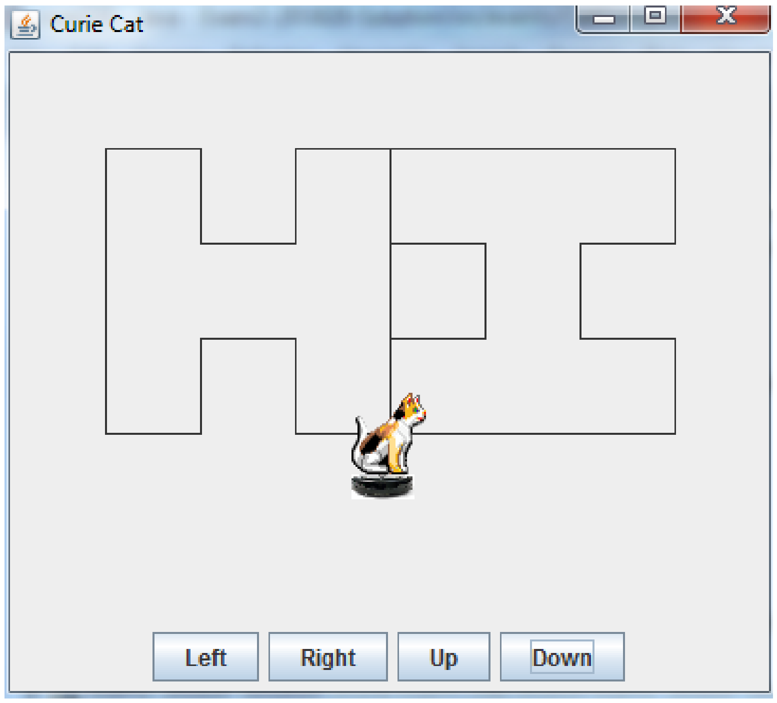

Figure 1: Final UI

In the package **events** is code for a simple program that displays a cat riding a Roomba, which is a small robotic vacuum cleaner. Much of the drawing code has already been written for you, but you will have to add the buttons and make them work.

You may add any new classes or make any changes to existing code you feel necessary, but you must use **good design.** In particular, **do not use static variables or methods.** Students who write static code will earn **no** credit.

Part 1: (4 points) Add the four movement JButtons to the bottom of the given UI. Lay them out per the picture.

Part 2: (12 points) Connect the four JButtons to the Roomba so that the Roomba actually moves in the right direction when the JButtons are pressed.

Part 3: (4 points) Enhance the drawing code to trace the path that the cat makes as she rides her Roomba around the screen. Whenever the cat moves, add a new line segment to the screen connecting the Roomba's last position to its current position.

After several moves, you should see a winding line tracing her path from the start of the program. The lines should appear underneath the cat (so draw the line before the cat). By default, the lines will appear to come from the center of the cat, which is OK with us. (Actually, having the lines come out of any part of the cat or roomba is OK.)

For full credit, use **good design.**