

CSSE 220—Object-Oriented Software Development

Exam 2 – Part 2, Oct. 24, 2015

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.

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 I can give you (a small amount of) partial credit.

Begin part 2 by checking out the project named *Exam2-201510* 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.

Part 2 is included in this document. **Do not use non-approved websites like search engines (Google) or any website other than those above.** 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.



Initial game state (left). During play (center). After clicking a mine (right).

Part 2—Computer Part

Problem Descriptions

Part C1: Recursion Problems (15 points)

The class `RecursionProblems` contains 3 recursion problems (test cases are also included). These problems must be solved with recursion - a working solution with loops is only worth 1/5 of the credit.

Part C2: Minesweeper GUI (15 points)

Minesweeper is a simple computer game played on a grid (7x7 in this problem) - randomly scattered across the grid are a random number of “mines.” The goal of the game is to identify the squares with the mines without accidentally setting a mine off.

Initially, the game board has all “?” labelled squares plus a message “Enjoy playing minesweeper.” Players click on squares to reveal them. If a player clicks on a square without a mine, the text of the square changes to the number of mines in the 8 squares adjacent to the clicked square.

If a player clicks a square with a mine, the square displays an “X” and the player has lost. The message then changes to “You lose!” (see picture). The player can continue clicking on other squares after losing if he/she wishes.

Your program need not change the message or perform any special actions if a player wins.

To help you implement this, we’ve provided a class called `MinesweeperGame` that creates a 7x7 Minesweeper board and populates it with mines. It has functions that let you detect mines or compute the number of mines adjacent to a particular square. You should NOT modify the `MinesweeperGame` class in any way. You MUST use this class in your solution - you should not create your own Minesweeper board from scratch.

Your job is to create the GUI. Your GUI should approximately match the one on the pictures given, but it does not have to be exact (your buttons can be slightly different sizes or be positioned slightly differently).

8 points of your grade depends on your ability to get the starting board positioned and looking correct. 7 points of your grade depends on the game actually playing correctly.