COP3503 Computer Science 2
Dr. Andrew Steinberg
Spring 2025

Programming Assignment 2
Possible Treasure Coordinates
Due: 2/16/2025 at 11:59pm

Objective: Students will apply concepts of advanced Backtracking.

Assignment Description: You have discovered a very old crumbly piece of parchment that contains a bunch of digits in between a pair of parentheses. You realize that these digits symbolize a precise location to a valuable piece of treasure. Luckily, you can put together the digits, however you are unsure if a decimal place exists from the wrinkles on the parchment. The best thing you can do is come up with **all possibilities** to at least get some idea of where the treasure is possibly located. In order to solve this problem, you will need to apply backtracking to generate all possible answers.

The coordinates are presented with a sequence of digits (0-9) all enclosed between a pair of parentheses. For example (123) is a sample input. This will result in the possible coordinates:

- (1, 2.3)
- (1, 23)
- (1.2, 3)
- (12, 3)

From this example, we have a possible set of scenarios of where the decimal place could be placed.

Implementation Details for this Assignment:

1. You are going to create a Class called `TreasureCoordinates`.
2. There is no need to implement an actual constructor as we are going to utilize the default one that is already provided.
3. You will implement a non-static method called `determineCoordinates` that takes one parameter which references a string of the digits between a set of parentheses. The method will return a reference that points to an `ArrayList` that stores strings. Those strings will represent the possible coordinates. This method will compute and store all possible 2D coordinates that can be generated from the input parameter. Each computed coordinate will be a string in the form `"(x,y)"` where `x` represents the x-coordinate and `y` represents the y-coordinate. Each index in the `ArrayList` is a possible coordinate. Your method must use **recursive backtracking** to generate all possible solutions to store in the `ArrayList`. Based on the example from above, the method would return [(1, 2.3), (1, 23), (1.2, 3), (12, 3)]. If iterative backtracking is used, then no credit will be given for the determineCoordinates category.
4. The `determineCoordinates` method must have a running time of $O(n^3)$.
5. You are welcome (and encouraged) to create additional helper methods as you see fit. Please make sure that the methods are implemented in your solution file and NOT the driver file (as the driver file will be modified for grading purposes).

6. You may assume that the digits are in the expected position. That means you do not need to compute possible permutations (reordering) of the digits.

About the driver file: A driver file (`TreasureCoordinatesDriver.java`) has been provided for you to show you how the `determineCoordinates` method is called along with 5 test cases to see if you get the same scenario result. **Do not change any contents of the driver file as the graders will be provided with a new one that changes some of the test cases.**

What to submit: Submit a file called `TreasureCoordinates.java` to webcourses. You are not required to submit the driver file as that will be provided for the graders to test your code. Please make sure the driver file provided works for your code. Any name changes may cause your program not to work when graded, which will result in a lower score on the assignment and would not be changed.

Important Note for running Eustis: Many of you are probably using IDEs like Netbeans and Eclipse to build your Java Solutions. Please note that some of these IDEs will automatically place your Java file in some sort of package. Please make sure your Java file is not defined in some package as this can result package private errors. Any such error that occurs during the grading will not be fixed and points will be deducted as such in accordance with the respective categories in the rubric. Also, DO NOT create a main method in your solution file!! This will result in your code not running properly with the runner file which will result in points being deducted from the respective categories.

Code Style Requirements: It is expected that students will follow the provided code style guidelines. These guidelines are available in webcourses on the assignment page. Students who violate any of the guidelines will receive point deductions!