**Introduction**

* This assignment continues the program begun in the previous unit. This time you will add the ability to explore multiple paths, especially ones that lead to a dead end. By the end of Unit 4, you will have a program that can read a 2-dimensional cave layout from a file, search the layout to find a path to a mirror pool, then print the path to take.

This assignment builds a set of classes to implement a search through a cave.

**Directions**

Write a class CaveExplorer, that has the following methods:

1. Complete your program from the previous unit if you have not already.
2. Extend your method solve to handle branching paths. In the example below, if the first move is West, then the explorer finds the mirror pool. But if the first move is South, the explorer hits a dead end. One trick to doing this is to use a stack. Whenever the explorer comes to a location with two or three possible moves, the moves should be put on a stack. If the explorer ever reaches a dead end without finding a mirror pool, then it’s time to pop the stack to make a move on a path not yet taken.

5 6  
RRRRRR  
R..S.R  
R.R.RR  
R.MRRR  
RRRRRR

1. Extend your method getPath so that it shows exactly a path to the mirror pool, without showing any dead ends. This will require a little extra bookkeeping and coordination with the solve method.
2. main – test your class by writing a main method that creates at least 4 CaveExplorer objects, prints the starting layout, the final layout, and the path taken, if it exists, for each one. Be sure to create test cases that require backtracking.
3. Answer the following questions in your Word document:
   * Describe the state of your project, what works and what doesn’t.
   * Describe how you tested your program, including tests that made you rethink your code. Include the layouts you used.
   * In a sentence or two, what did you learn?
   * In a sentence or two, what did you like about this project?
   * In a sentence or two, what did you find confusing or would like to see done differently regarding this project?
   * In a sentence or two, if you had another hour or two, what would you like to add to the project or how would you do things differently?

**Notes**

After your program is working for each step of the process, you should commit your changes to the repository. Be sure to include your data file(s) as well. This allows you to go back to a previous working version in case you decide to throw out new code and try a different approach.

**Submit**

Be sure to put your Word document in your repository and do one final commit. Generate a .zip file from the repository and submit that file to Canvas. Double check that what you submit to Canvas has all your .java files.

**Due Dates**

* by 11:59 p.m., Sunday, CT