Problem Solving Session

- The remainder of today's class will comprise the ***problem solving session*** (***PSS***).
- Your instructor will divide you into ***teams of 3 or 4 students***.
- Each team will ***work together*** to solve the following problems over the course of ***20-30 minutes***.
  - You may work on paper, a white board, or digitally as determined by your instructor.
  - You will submit your solution by pushing it to GitHub before the end of class.
- Your instructor will go over the solution before the end of class.
- If there is any time remaining, you will begin work on your homework assignment.

Class participation is a significant part of your grade (20%). This includes in class activities and the problem solving session.

Your Course Assistants will grade your participation by verifying that you pushed your solutions before the end of the class period each day.

Problem Solving Team Members

Record the name of each of your problem solving team members here.

Do not forget to ***add every team member's name***! Your instructor (or course assistant) may or may not use this to determine whether or not you participated in the problem solving session.

AMAN PATEL
Emmalee Carpenter
Dessa Shapiro

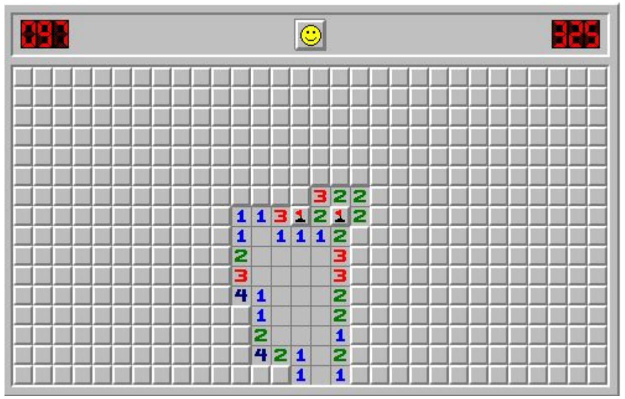
Problem Solving 1

Minesweeper is a classic computer game in which you try to uncover every tile on a board without hitting a mine. In general you have two options available for each turn, uncover a tile or mark the tile with a flag. You should uncover a tile when you are pretty sure there isn't a mine under it and mark it with a flag if you it is highly likely there is a mine under it.

Take a couple minutes and try out the game at https://minesweeperonline.com/.

For this assignment you'll be writing a solver, using backtracking, for a game of minesweeper.

To simplify the complexity, we won't be writing a full game. Therefore the solver doesn't need to know how many mines are nearby and spaces will only be uncovered by selecting them. I.E. if the solver selects a tile with no mines near it, the surrounding space will not be uncovered.



Problem Solving 2

Many games are played on a 2-D board, and Minesweeper is no different. However, it is often easier to work with a one dimensional array in code.

Given the illustration below about converting a 2-D array into a 1-D array, answer the questions on the right.

(0,0)	(0,1)	(0,2)	(0,3)
(1,0)	(1,1)	(1,2)	(1,3)
(2,0)	(2,1)	(2,2)	(2,3)
(3,0)	(3,1)	(3,2)	(3,3)

What would be some of the benefits of using a 1-D array over a 2-D array?

Easier to access each value through one loop.
Easy to copy
Easy to iterate through

Using `row/column` ordering, write a formula to determine the `index` in the 1-D array for any given `row/column` pair and the number of `rows` and `columns`.
Given row n and col m
Index = ((n*m) + m)

Given the total number of `rows` and `columns`, write the formula to determine the specific `row/column` for any given `index` in the 1-D array.
Index n
row = n / rows
col =n%columns

(0,0)	(0,1)	(0,2)	(0,3)	(1,0)	(1,1)	(1,2)	(1,3)	(2,0)	(2,1)	(2,2)	(2,3)	(3,0)	(3,1)	(3,2)	(3,3)

Problem Solving 3

Assume that you have been provided with a map (`solution_map`) of all the mine locations. This is how you will determine if a tile has a mine under it when you flip it over.

Answer the questions on the left about what other information your configuration class will need to solve the game.

What state (fields) will your configuration need in addition to the `solution_map`?
Rows, cols, board array, index, spaces visited

How many successors will each configuration have? How will you make them?
2, one for selecting as a tile and one for flagging.
Create a copy of the current configuration and flip the next index in one and flag the same next index in the other. Must always flip as the first successor otherwise could flag all the tiles (must flip before flag)

How will you determine if a configuration is valid or not? Will you have a pruning strategy?
Check to see if flipped tile doesn't have mine under or its a flaga

How will you determine if a configuration is the goal?
If every tile has been flipped or flagged. Only works if discovered mine and then flagged it after seeing it.

Problem Solving 4

It is critical that each successor start with a deep copy of the previous successor. In the space to the right, detail the state for your `Minesweeper` configuration.

Also, complete the copy constructor for the configuration. Ensure all elements are being copied using a deep copy and not a shallow one.

```
public class Minesweeper implements Configuration {
    // State
      Private final String board [];
      Private final int index;
      Private final int rows;
      Private final int cols;
      Private final int count;
      Private final char map [];




    // Copy Constructor
    public Minesweeper (Minesweeper orig) {
      Board = arrays.copyOf(orig.board, orig.board.length);
      Columns = orig.columns;
      Rows = orig.rows;
      This.index = orig.index;
      Count = orig.count;

      this.map = orig.map;
```

Problem Solving 5

Think about the `getSuccessors` method in your `Minesweeper` configuration.

- What does each successor represent?
- How many will there be?
- Will you have a pruning strategy?

Write pseudocode that describes the algorithm you think that you will need to implement in your implementation.

Remember to be detailed - each line of pseudocode should roughly correspond to one statement in a programming language.

```
Create a copy of the current minesweeper
Increment the location index of the board
Increment the count (if using one)
Update the value at the index to indicate if it was a
tile select.
```

```
Create a copy of the current MineSweeper
Increment the location index of the board
Increment the count(if using one)
Update the value at the index to indicate if a flag was
placed on that tile
```