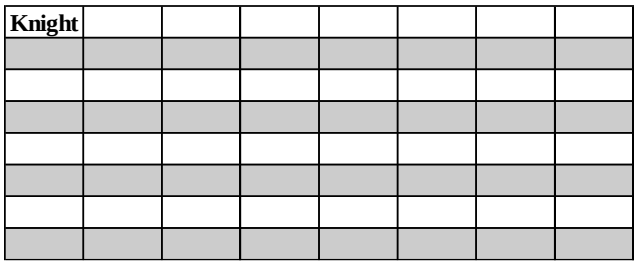
COP3503 Computer Science 2
Dr. Andrew Steinberg
Spring 2025

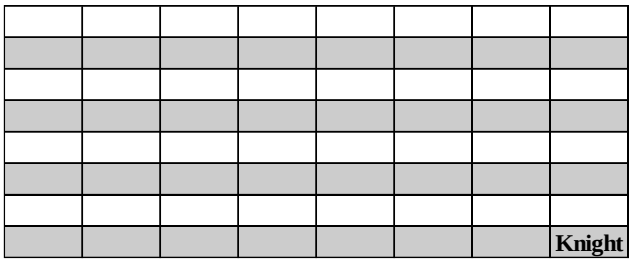
Programming Assignment 1
The Winning Strategy
Due: 1/31/2025 at 11:59pm

Objective: Students will apply concepts of clever problem solving in this assignment and warmup with basic Java skills.

Assignment Description: We are going to play a fun strategy game that only requires two players! In this game, we have an 8 x 8 board and a Knight chess piece (that both players share) that starts on the top left of the board. Each player gets to move the Knight piece one square over either down, diagonal (bottom right direction only), or to the right of its current position (a player cannot move the piece two or more squares). The Knight piece can keep moving until it reaches the bottom right corner of the board. The respective player that moves the Knight to the bottom right corner of the board wins the game! In this assignment you are going to implement the winning strategy for player 1 only.



When the game begins, the knight is on the top left square of the board.



When the game ends, the knight is on the bottom right square of the board.

For this assignment, you must follow these **requirements**.

1. You will create a public class called `Game`.

2. The `Game` Class will have the following attributes:
    a. 2D integer array that symbolizes the board to assist you with simulating the game.
    b. 1D primitive character array that stores the respective moves that will be used for player 2. **Please store them in order of how they are presented here.**
        i. ‘d’ for downward diagonal right direction (index 0)
        ii. ‘r’ for horizontal right direction (index 1)
        iii. ‘b’ for vertical bottom direction (index 2)
    c. A reference to a Random Class object. This will point to the object created by the driver and passed to the constructor. This should be used when selecting a random move for player 2 in the `selectPlayerTwoMove` method.

3. The `Game` Class has a constructor that takes one parameter.
    a. A reference to a Random Class object that is instantiated in the driver class.

4. The `Game` class has a public non-static method called `selectPlayerTwoMove`. The method will select a random move from the 1D primitive character array attribute that stores the respective move (from 2b). This method is intended to select player 2's move only. This should not be used for player 1. The method should return a primitive character.

5. The `Game` class has a public non-static method called `play`. `play` will simulate a round of the game where player 1 must win against player 2. The method has no parameters. The method should return a value 1 if the player 1 wins the game, if not return 2 or some other value. **NOTE: If a student just simply writes the statement that ONLY returns the player's number without any simulation to the game will receive an automatic score of 0 on the assignment with NO partial credit from the other categories in the rubric. You may also not apply backtracking to this assignment. If backtracking is applied, then you will receive a score of 0 on the assignment.**

6. You are allowed to create helper methods as long as they are not called directly from the driver file. The helper methods must be called from your solution file.

A driver file (`GameDriver.java`) has been provided for you to show you how the methods are called along with 10 test cases to see if you get the same scenario result.

What to submit: Submit a file called `Game.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. You do not need to submit the text files of moves.

Comment Header: Please make sure to provide the appropriate comment header (like in the Eustis assignment) as the first thing on the top of your file. Otherwise, if your comment header is done incorrectly including not being placed literally at the top of the file will result point deductions in accordance with the rubric.

Hardcoding Output: For this assignment it is expected that you implement an actual solution simulation to the game. If a student just places in their code to return the respective value without any sort of simulation of the game, the student will automatically receive a score of 0 with NO partial credit on any categories in the established rubric.

COP3503 Computer Science 2
Dr. Andrew Steinberg
Spring 2025

About the Driver Class

The `GameDriver` Class will test your play method with 10 different test cases. The first part of the code will create 10 different random objects with unique (seeds). Each object will be associated with a test case. Please remember Dr. Steinberg will change a few test cases when grading your code. After the random objects are instantiated, the Game class objects are instantiated with their respective random object reference. Last, all play methods with the respective game object is invoke and will determine if player 1 won.

Possible FAQs:

Here are some possible FAQs about the assignment.

1) Are there 2 knight pieces on the board that each player uses? **Answer**: No, there is only one knight piece that each player shares.
2) Can either player stall each other by moving the knight piece backwards? **Answer**: No, this will cause an infinite loop scenario. The only moves players can make are down, diagonal (bottom right direction), and right.
3) If the knight piece is on the last row/column of the board and the respective player wants to make an invalid move that would cause the knight piece to go out of bounds, does that count as a move for the player? **Answer**: No, invalid moves cannot be made in the game. If the knight piece is on the last row/column, then that means each player only has one valid option to make. You will have to include that scenario in your code.
4) What direction is the diagonal move? **Answer**: The direction is south east. This means down and to the right.

Important Note for running Eustis and Packages: 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 driver file which will result in points being deducted from the respective categories.

Code Style Requirements: It is expected that students follow the expected code style guide requirements in this course. Dr. Steinberg plans to release a guide that is expected for students to use on Programming Assignment 2 and beyond. Programming Assignment 1 will be a grace period to allow students to understand the guide and what it is expected. Dr. Steinberg will make a separate announcement for when the guide is available. Moving forward, it is expected that students will follow all code style guidelines specified for this class. Any violations to the code style guidelines will cause points to be deducted in your future programming assignments (programming assignment 2 and beyond).