Project 1: Board Game

Due date: Tuesday October 2, 11:55PM EST.

You may discuss any of the assignments with your classmates and tutors (or anyone else) but all work for all assignments must be entirely your own . Any sharing or copying of assignments will be considered cheating (this includes posting of partial or complete solutions on any public forum). If you get significant help from anyone, you should acknowledge it in your submission (and your grade will be proportional to the part that you completed on your own). You are responsible for every line in your program: you need to know what it does and why. You should not use any data structures and features of Java that have not been covered in class (or the prerequisite class). If you have doubts whether or not you are allowed to use certain structures, just ask your instructor.

In this project you will help determine the probability of winning a board game. Your program will represent the board in a double linked list. Your program will also simulate playing the game with 1,2,3 and finally 4 players. The program will output the average number of moves necessary for one of the players to "win" the game.

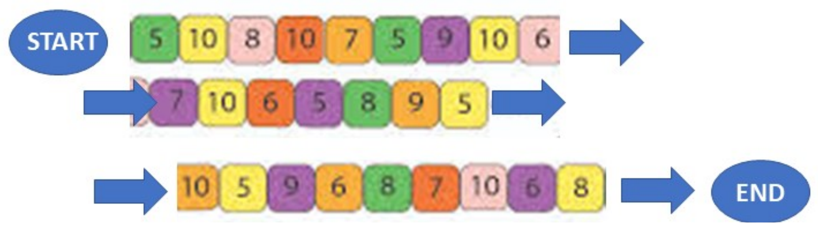
Objectives

The goal of this programming project is for you to master (or at least get practice on) the following tasks:

- Representing a real life board game within a data structure
- Modifying the code of an ADT
- Employing simulation techniques
- Writing classes
- Working with existing code

Start early! This project may not seem like much coding, but debugging always takes time. Analyze and plan now so questions are not being asked a day before the due date.

Board Game:



Board Game rules:

A player begins the game on the Start circle. The current player rolls a dice, that gives a number from 1 to 6. The current player moves that many colored blocks from the current player's position. If the square is unoccupied, then the current player adds the number on the square to the player's total. If the square already has a previous player on it, the previous player has to go back 7 spaces , and the current player adds the number on the square the current player's total. If a player moves back 3 spaces and shares a block with another player, that's ok, no additional processing. If the player is moved back beyond the first square, put the player on Start circle.

Use the Board Shown Above

When a player reaches the End circle or beyond , and has at least 44 points the game is over. When a player reaches the End circle or beyond , and has less than 44 points , the player must go back to the Start. A player can never go before the start square or after the end square.

Program Logic:

You must use a double linked list to represent the Board. Your program should start by creating nodes in the double linked list to represent the board. You need to determine the class (data type) of an element in a node, and the information that element needs to hold. That information should include attributes of the board space. Your program should also be able to print the Board, and show the current position of each player. The layout of this printout must have 3 rows as shown in the diagram.

Play 1000 games with only Player A in the game. Output the average number moves it takes to win. Next play 1000 games with player A and B. Note the average for each of them to win. Do this for up to 4 players. A sample matrix would be:

RESULTS TABLE

Player in game	Player A average moves / % winning	Player B average moves/ % winning	Player C average moves/ % winning	Player D average moves/ % winning
A	30, 100%			
A,B	25 / 62%	27 / 38%		
A,B,C	23/ 35%	24 / 62%	26 / 62%	
A,B,C,D	22 / 27%	20 / 25%	21 / 24%	24 / 24%

Expected Actual numbers will be provided during the project period for comparison. Print the game board for every 100th game at the end of the game, so one can see the final position of each player.. So you will have 10 board prints for each row above.

Notes on coding
1) You may use and modify the DoubleLinkedList class. This includes modifying the nested Node class.
2) Make sure to use plenty of loops
3) Be creative displaying the game board.

Outputs

1) Your program should print the board at the end of the game for the first game, 101st game, 201st game..etc. In all your output would have 40 prints of the board.
2) Your output should include the Results Table above .

Running the program

The zipped java project file, which contains all your source code, Eclipse related project files, is to be submitted. If you submit an incorrect file (say , just the source file) you will lose 10% of the project value. If you are unsure of what a zipped Eclipse Java folder means, you can submit a non-working early version to be sure.

Working on This Assignment

You should start right away! You should modularize your design so that you can test it regularly. Make sure that at all times you have a working program. You can implement methods that perform one task at a time. This way, if you run out of time, at least parts of your program will be functioning properly.

Grading Criteria – 20 points
1. (3) Program can use a double linked list to represent the board.
2. (3) The double linked list accurately reflects the board of the problem.
3. (2) One game can be properly played with one player
4. (4) One game can be properly played with multiple players
5. (4) 1000 games for each group of players , described above, can be played and statistics generated
6. (4) The board can be displayed after the 100th games of each set..