Homework 3

1 Introduction

This assignment will cover arrays, iteration, static and instance methods, as well as the Math and Random classes.

2 Problem Description

Games of chance and probability have captivated humans since ancient times. Whether we are rolling for crits in a board game or gambling in a casino, people love to place bets on the outcomes of random events. With the advent of computers we now have the ability to generate pseudorandom, or seemingly random numbers using advanced algorithms. For this homework you wont be creating your own random algorithm (although those are cool), but you will be making a game that supports a few random guessing games.

3 Game Mode Description

For this homework, you will be coding a random guessing game which will support all of the following games:

- **Guess the Card**
  The user must guess the suit and rank of the card. The suit must be entered as a String (either spades, clubs, diamonds, or hearts) and the rank of the card may be entered as a number (Ace is 1, Jack is 11, Queen is 12, and King is 13, with the number cards just represented as their number). The user should guess the suit, then the rank of the card. The user is given 2 points for guessing the suit and 6 points for guessing the rank.
- **Guess the Die Roll**
  The user must guess the outcome of a die roll. The user first enters how many faces they would like on the die, which can be any positive number. The user then enters a guess as to what the die will land on. If the user guesses correctly they should receive floor(# of faces/2) points.
- **Guess the Coin Toss**
  The user must guess which side of a coin will land face up. The user must guess the side of the coin, either 'h' (for heads) or 't' (for tails). Numerically (in your code), heads is represented as 1 and tails as 2. If the user guesses correctly they are given 1 point. This game is implemented for you to use as a reference or example.

4 Solution Description

Your task is to finish a skeleton implementation of the provided `RandomGame` and `GameDriver` files.

`RandomGame`
The `RandomGame` class provides a general model of a guessing game. You will use this class to implement the different games described above by implementing the following:

- Fields
  - `randomValues` is an array of integers which represents the possible random outcomes which the user will guess from. We prepopulate to these values to ensure fairness in `generateEvents()`. For example, when rolling a normal dice, the array would contain random values from 1 - 6 inclusive.
- Methods
  - `checkGuess(int guess)` checks if a user guess matches the next value in `randomValues`. Your code should start checking at index 0 in `randomValues`!
  - `getLastEvent()` returns the last value checked against the users guess. For example, `checkGuess(int guess)` is called and checks the users guess against the number 7, and then `getLastEvent()` is called, it should return 7 since that was the last value that we used to check the users guess
  - `generateEvents()` fills the random value array with new values in the appropriate range. This should be called whenever all the values in the random value array are used.

`GameDriver`
The `GameDriver` class implements the coin, card, and dice games described above and provides a command line interface for a user to select each game and accumulate points. Each game has its own method in `GameDriver` and should be self-contained in that method. These self-contained methods will be called from the main method based on which game the user decides to play. In addition, the `RandomGame` instances for the rank and suit are given to you in static variables called ”cardRankGame” and ”cardSuitGame”. You must use these instances when coding the card guessing game. You must implement each of the following methods:

- `playDiceGame(Scanner input)` - implements the dice game described above
- `playCardGame(Scanner input)` - implements the card game described above
- `main(String[] args)` - This method should initialize the `bufferSize` variable based on the command line argument or to 128 if one is not provided. It will also need to initialize necessary `RandomGame` objects for the coin and card games, which has been provided. Finally the method should process user input similar to what is shown below to allow the user to select which game to play and accumulate points.

5 Example Output

```
$ java GameDriver
Enter the game you would like to play (coins, dice, cards): coins
Enter your guess (h/t): h
Nope! Your guess: h; Correct: t
Would you like to play again? (y/n): y
Enter the game you would like to play (coins, dice, cards): coins
Enter your guess (h/t): t
Yep! Your guess: t; Correct: t
Would you like to play again? (y/n): y
Enter the game you would like to play (coins, dice, cards): dice
Enter the number of faces on the die: 1738
Guess which side will be rolled: 420
Nope! Your guess: 420; Correct: 1389
Would you like to play again? (y/n): y
Enter the game you would like to play (coins, dice, cards): cards
Guess the suit (spades, clubs, diamonds, hearts): spades
Guess the rank as an int (ace is 1; king is 13): 12
Nope! Your rank guess: 12; correct rank: 5
Your suit guess: spades; correct suit: clubs
Would you like to play again? (y/n): n
Your score is: 1
$
```

6 Helpful Hints

- You can assume the user will only enter valid data.
- The Math class contains methods which could be helpful when trying to fit random values into a specific range.

7 Checkstyle

You must run checkstyle on your submission. The checkstyle cap for this assignment is **15** points.
Review the Style Guide and download the Checkstyle jar. Run Checkstyle on your code like so:

```
$ java -jar checkstyle-6.2.2.jar *.java
Audit done. Errors (potential points off):
0
```

The message above means there were no Checkstyle errors. If you had any errors, they would show up above this message, and the number at the end would be the points we would take off.

The Java source files we provide contain no Checkstyle errors. For this assignment, there will be a maximum of **15** points lost due to Checkstyle errors (1 point per error). In future homeworks we will be increasing this cap, so get into the habit of fixing these style errors early!

8 Turn-in Procedure

Non-compiling or missing submissions will receive a zero. NO exceptions

Submit `RandomGame.java` and `GameDriver.java` to T-Square. Do not submit any compiled bytecode (`.class` files) or the Checkstyle jar file. When you're ready, double-check that you have submitted and not just saved a draft.

Please remember to run your code through Checkstyle!

8.1 Verify the Success of Your Submission to T-Square

Practice safe submission! Verify that your HW files were truly submitted correctly, the upload was successful, and that the files compile and run. It is solely your responsibility to turn in your homework and practice this safe submission safeguard.

1. After uploading the files to T-Square you should receive an email from T-Square listing the names of the files that were uploaded and received. If you do not get the confirmation email almost immediately, something is wrong with your HW submission and/or your email. Even receiving the email does not guarantee that you turned in exactly what you intended.
2. After submitting the files to T-Square, return to the Assignment menu option and this homework. It should show the submitted files.
3. Download copies of your submitted files from the T-Square Assignment page placing them in a new folder.
4. Recompile and test those exact files.
5. This helps guard against a few things.
   (a) It helps insure that you turn in the correct files.
   (b) It helps you realize if you omit a file or files. [1] (If you do discover that you omitted a file, submit all of your files again, not just the missing one.)
   (c) Helps find last minute causes of files not compiling and/or running.

[1] Missing files will not be given any credit, and non-compiling homework solutions will receive few to zero points. Also recall that late homework will not be accepted regardless of excuse. Treat the due date with respect. The real due date is midnight. Do not wait until the last minute!