CSCI 145 -- PA 9

More on OOD & Arrays

Feel free to discuss and help each other out but does not imply that you can give away your code or your answers!   You can work with a lab partner for this assignment. **You must always use the required template (JavaClassTemplate.java from Canvas) and output "Author: Your Name(s)" or "Modified by: Your Name(s)" for each program as applicable**.

You can work with a lab partner and each one must submit the same PDF file (include both names in the submission file). Each person must include a brief statement about your contribution to this assignment.

Perform the following exercises of lab book:

- **Exercise 1** -- A Biased Coin (from chapter 7)
- **Exercise 2** -- Reversing an Array (from chapter 8)
- **Exercise 3** – Magic Squares (from chapter 8); should rename data file MagicData to MagicData.txt; do not sort the list every time in the add() method and use a single loop to find the right spot to insert the value.

Exercise 4 -- Design and implement an application that processes the votes for 10 candidates with an id between 1 and 10 that will be used for the candidates (a value of 1 represents a vote for candidate 1). After all votes have been processed, print the number of votes, all candidates with at least 1 vote (id and votes), and the overall winner (in case of a tie, print one). Use 0 as sentinel and ignore values outside the range. *Hint: The counting should be very similar to LetterCount program in chapter 8 of the text and rolling dice in previous lab*.

Sample input/output:

```
Enter votes: 5 3 5 12 3 5 10 2 0<Enter>

Number of valid votes: 7
Results(candidate id and number of votes):
 2     1
 3     2
 5     3
 10    1

Winner: candidate 5
```

Question 1: Describe some differences between a **class** and an **interface**.

Question 2: Are there any good reasons for choosing an **ArrayList** class over an array? Explain.

Can only extra points for one of the 3 extra credit options below:

Extra Credit 1: Modify exercise 4 to handle a special case where multiple candidates have highest number of votes (2 or more candidates). Print all candidates with the highest number of votes and then randomly select a winner (make sure the chances are the same for all candidates). You can submit this one version to include exercise 3.

Sample input/output:
```
Enter votes: 5 3 5 12 3 5 10 2 3 0<Enter>

Number of valid votes: 8
Results (candidate id and number of votes):
 2     1
 3     3
 5     3
 10    1

Tie: 3 5
Randomly choosing a candidate …
Winner: 5
```

Note: winner could be 3 or 5, add a test case to include 3 candidates with highest votes.

Extra Credit 2: Use JUnit as discussed in class to unit test **BiasedCoin** class from exercise 1 above. You might need to set up a loop to count and then assert a certain value (e.g., if a coin has 0% percent of heads, then count should be 0 for 1000 flips; if a coin has 20% percent of heads, then count should be around 200 for 1000 flips so it would be reasonable to assert that the count should be between 175 to 225). Submit the unit test file and a screen shot showing all test cases are passed.

Extra Credit 3: Modify Roulette program from previous PA to allow a list of players (use an array of Player) to play at the same time and the game only stops when all players quit the game. *Hint: you should add an isActive() method to Player class to determine if a player is still playing the game.* Sample output using nextRandom method (must run this case):

```
Author: [Your Name]

Welcome to a simple version of roulette game.
You can place a bet on black, red, or a number.
A color bet is paid 2 the bet amount.
A number bet is paid 35 the bet amount.
You can bet on a number from 1 to 36.
Gamble responsibly.  Have fun and good luck!

Enter number of players: 3

Money available for player1: 100
Betting Options:
    1. Bet on black (even numbers)
    2. Bet on red (odd numbers)
    3. Bet on a number between 1 and 36
```

```
Enter a bet option, player1 (1, 2, or 3): 1
You chose option 1
How much to bet: 5
You chose to bet $5 on Black color.

Money available for player2: 100
Betting Options:
    1. Bet on black (even numbers)
    2. Bet on red (odd numbers)
    3. Bet on a number between 1 and 36

Enter a bet option, player2 (1, 2, or 3): 2
You chose option 2
How much to bet: 15
You chose to bet $15 on Red color.

Money available for player3: 100
Betting Options:
    1. Bet on black (even numbers)
    2. Bet on red (odd numbers)
    3. Bet on a number between 1 and 36

Enter a bet option, player3 (1, 2, or 3): 1
You chose option 1
How much to bet: 20
You chose to bet $20 on Black color.

Spinning …
Current number: 20, color: Black

player1 won 10.
player2 lost this round.
player3 won 40.

Play again, player1? [y/n] n
Play again, player2? [y/n] y

Money available for player2: 85
Betting Options:
    1. Bet on black (even numbers)
    2. Bet on red (odd numbers)
    3. Bet on a number between 1 and 36

Enter a bet option, player2 (1, 2, or 3): 1
You chose option 1
How much to bet: 10
You chose to bet $10 on Black color.

Play again, player3? [y/n] n

Spinning …
Current number: 5, color: Red

Player2 lost 10.
Play again, player2? [y/n] n

player1 won 5 for this game.
player2 lost 25 for this game.
player3 won 20 for this game.
Game over! Thanks for playing.
```

Fill out and turn in the PA submission file for this assignment (save as PDF format).
