CSCI 145 -- PA 5

Conditionals, Loops, and GUI Applications

Feel free to discuss and help each other out but does not imply that you can give away your code or your answers!   You cannot 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**.

Perform as many exercises from chapter 5 of lab book as possible, but the following lab exercises must be completed. You are not required to turn in written answers to various questions, but it is very helpful in understanding important concepts. You might see those questions on quizzes and exams.

- **Exercise 1** -- Computing a Raise
- **Exercise 2** -- A Guessing Game

Exercise 3 -- Vote Counter, JavaFX – must use JavaFX and start with VoteCounterFX.java and VoteCounterPaneFX.java file provided via this PA; only include source code for VoteCounterPaneFX.java in your submission. Follow instructions below:

> 1. Add variables for Sam - a vote counter, a button, and a label.
>
> 2. Add the button and label for Sam to the panel; add the listener to the button.
>
> 3. Modify the handle() method of the ButtonHandler class to determine which button was pressed and update the correct counter. (See example in textbook for how to determine the source of an event.)
>
> 4. Test your program.
>
> 5. Now modify the program to add a message indicating who is winning. To do this you need to instantiate a new label, add it to the panel, and add an if statement in handle() method that determines who is winning (also test for ties) and sets the text of the label with the appropriate message.

If you cannot run JavaFX you can perform an alternate exercise below:

Alternate Exercise 3: Perform exercise **Vote Counter, Revisited,** with Swing Components from the lab manual**.** Use the files VoteCounter.java and VoteCounterPanel.java from lab source code zip file and modify VoteCounterPanel.java to count votes.

Exercise 4 – Modify the Roulette game from previous PA to add the following:

1. Make sure user selects a valid betting option in makeBet method. User is not allowed to bet on zero, double zero, or green color in this game. Note that the method betOptions is static in the Roulette class, therefore its invocation is through the Roulette class. Make sure to utilize public named constants from class Roulette when perform data validation as much as possible. We need a check to ensure that the bet amount is valid (between MIN_BET which is a named constant in Roulette, and the available money of that player). Print an error message and re-prompt for invalid betting amount by inserting that check in the makeBet method.
2. We need to make one more modification to the *makeBet* method. If the user wants to bet on a number, we must prompt for and read that number. Be sure to validate that number so it would be between **Roulette**.*MIN_NUM* and **Roulette**.*MAX_NUM* . Store it in a variable called *number* in the player object. Use the constant **Roulette**.*NUMBER* to test for that *betType*.
3. We need to be able to spin the roulette wheel. Complete the static method called *spin* in the **Roulette** class that randomly determines the number and sets the predefined static variables *ballPosition* and *color* (can use method nextRandom() to get next number for testing). Make the color corresponds to the *RED*, *BLACK*, or *GREEN* constants already defined in the class. At the end of the spin method, print the results of the spin: both color and number. Move the two println after the call to spin() in main inside the spin() method.

Provide two sets of input/output. For this input/output, use nextRandom() to generate random number so your output should match this output.

```
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!

Money available for Jane: 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, Jane (1, 2, or 3): 1
How much to bet: 5
You chose to bet $5 on Black color.

Spinning …
Current number: 20, color: Black

Play again, Jane? [y/n] y
Money available for Jane: 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, Jane (1, 2, or 3): 4
Invalid betting option. Try again.
Enter a bet option, Jane (1, 2, or 3): 2
How much to bet: 10
You chose to bet $10 on Red color.

Spinning …
Current number: 5, color: Red

Play again, Jane? [y/n] y
Money available for Jane: 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, Jane (1, 2, or 3): 3
Enter a number between 1 and 36: 40
Invalid number. Try again.
Enter a number between 1 and 36: 1
How much to bet: 200
Invalid betting amount. Try again.
How much to bet: 1
You chose to bet $1 on number 1.

Spinning …
Current number: 0, color: Green

Play again, Jane? [y/n] n
Game over! Thanks for playing.
```

Provide another set of input/output that uses your own code to generate a random number in the spin method. Play at least 3 rounds and test some other situations that were not covered in the previous run before stopping playing. Check to make sure your spin result is reasonable.

Question 1: We cannot use int value 0 to represent 00 since 0 is used for 0 in exercise 4. What are two reasonable int values that we can be used for 00? Which one did you use for your code and how you tested it?

Question 2: List and briefly explain the four typical components of a loop.

Extra Credit: Perform exercise **Baseball Statistics** from the lab manual.

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