CS1083 – Programming 1 for Computer Scientists

Class Project – Dice Game

Objectives

The goal of this project is to help the student bring together all of the concepts learned throughout this course. The following are the main concepts for the project:

- Use of local and global variables
- Understanding data types and data type conversions
- Use of arithmetic expressions
- Use of if/else/switch statements
- Use of for loops, while loops
- Use of arrays
- Use of Scanner for user inputs
- File I/O
- State driven logic
- Use of methods

Hand-in Requirements

All projects will be submitted electronically only via Canvas. Students will compress their project directory and submit as a zip file. The zip file must contain at least the following file:

- DiceGameAbc123.java

Where "Abc123" should be replaced by your actual abc123. If your project requires any other files for loading saved state resources, you must include those as well.

Game Logic

The program will emulate a ***modified*** version of the dice game like "craps". Craps use two six-sided dice containing numbers 1 through 6. Craps is played according to the following rules:

1. The Come-out Roll aka "the first roll" of the two six-sided dice:
    - If the sum of the numbers on the dice is either 7 or 11; the player wins and the game is over.

- If the sum of the numbers on the dice is either 2, 3 or 12; the player loses, and the game is over. (aka crap out)
- If the sum of the numbers on the dice is any other number, that number becomes the point, and the game continues

2. Follow-up rolls, aka "rolling the point":
    - If the initial roll results in a point, the player will continue to roll the dice and win every time they roll the same point.
    - If the player rolls any other number that is not "the point" or "7", the game continues to the next roll
    - If the player rolls a "7", the game is over. (aka 7 out)
3. Betting: *to keep things simple, betting and winning will be modified from traditional rules*
    - Before the initial roll, the player must place a bet up to their current bank roll. The program must check to see if they still have money in their bank roll.
    - Once a bet has been placed, the player can win an equal amount (1:1) based on the rules set up in steps 1 and 2.
    - A player will lose only the amount they bet, not their cumulative winnings, based on the rules in steps 1 and 2.

Tasks

1. Create method(s) game menus with the following information:
    a. An intro message: i.e. "Welcome to Juan's Dice Game: Main Menu" (replace Juan with your name)
    b. Main Menu Options
        i. "Log In"
            1. Allow the user to enter a username. No password required
        ii. "Register"
            1. Allow the user to create a new account
        iii. "Quit"
            1. Ends the game and quits the program
    c. Game Menu Options; update the intro message to reflect this menu location
        i. "View Bank Roll"
        ii. "Place the bet"
        iii. "Roll the dice"
        iv. "Back to Main Menu"

2. Create a method(s) to handle the game logic based on the rules defined in the ‘Game Logic’ section of this document.
3. Create a method(s) to handle the registration
4. Create a method(s) to handle the log in

Details

Register Logic:

The user will be allowed to register using an email address, first and last name. This information will be stored in a file, along with an initial bank roll of $500. Your program must check to make sure that each user’s email address is only used once, duplicates are not allowed. First and last names must also only contain alphabet letters. Appropriate error messages must be displayed for any input errors encountered. Once the user is registered, the main game options must be displayed.

Log in Logic:

The user will be allowed to log in by entering an email address. The program must look for this data stored in a file; if the data is found, the main game options are displayed. Your program must load the remaining bank roll from file. If the data is not found, an appropriate error message must be displayed instructing the user to register before proceeding. If the user’s bank roll is $0, an error message must be displayed letting the user know they can no longer play. If the data is found and the bank roll is greater than $0, proceed to the main game menu.

View Bank Roll Logic:

The user will be allowed to view their current bank roll only in between current playing sessions, that is prior to the initial roll or after they win or crap out/7 out. The bank roll must reflect current wins/losses based on game play. This data should be stored in a file to prevent data loss.

Place bet logic:

The user will be allowed to enter any positive whole integer, up to their current bank roll. This is the number that will be used to calculate winnings and losses. The program must

check to make sure this number is not greater than their bank roll; present an error message otherwise. The program must also check for valid integer values, present error message otherwise.

Roll the dice logic:

Based on the game logic section of this document, the program must calculate the winnings based on the amount of each bet.

The Come-Roll example:

If the bet is $50 and the player rolls a 7 on the first roll, a message will be displayed letting the user know they rolled a 7 (with each individual number rolled: i.e.: 6-1, 4-3, 2-5) on the come-out roll and the player has won an amount equal to their bet, which is immediately added to their bank roll. Their bet is also added back to their bank roll and the player is presented with the game menu options.

The Craps-out roll example:

If the bet is $50 and the player craps out on the initial roll, a message is displayed to the user letting them know what was rolled (each individual dice) and that they crap out and lost, then the $50 is immediately deducted from their bank roll and the player is presented with the game menu option.

Follow-up (“point”) Roll example:

If the bet is $50 and the player rolls an 8, this number becomes the point; a message must be displayed to the player displaying the number they rolled (each individual dice) to let them know this is the point. The player will be prompted to roll again, each roll will be displayed to the player. If this new number matches the point, a message must be displayed to the player that they rolled the point number and have won an amount equal to their bet, this amount is immediately added to their bank roll. The player must roll again. This process continues until the player rolls a 7 (7-out), at which point a message is displayed to the player with the dice roll and that they 7-out and lose their initial bet. This initial bet is immediately deducted from their bank roll and the player is presented with the game menu option.

Back to main menu logic:

This menu option will be treated as the end to the gaming session, meaning that the user is done playing and will be logged out. The program must save all appropriate data to file to be used if the user decides to log in later. This menu option does not end the program, only the game session.

Quit Logic:

Stop the program.

Rubric

- Methods: [35 points]
    - All your program logic must be done within method calls, that is your public main should only contain global variable/constant declarations as well as method calls. [30 points]
    - Methods should call other methods for code re-usability, for example, there should be a roll dice method that returns each individual dice rolled and their sum. [5 points]
- Game Logic Implementation: [60 points]
    - Register Logic: [10 points]
        - Your users must be saved to a file for later processing. You must create a users.txt file containing username, first name, last name, bank roll (using comma separated values) [5 points]
        - Input must be error checked [3 points]
        - Duplicate users must be error checked [2 points]
    - Log In Logic: [10 points]
        - User must be prompted to enter a username (email) [2 points]
        - If the username is not found, display an error message [2 point]
        - If the username is found, load data into variables and proceed [6 points]
    - View Bank Roll Logic: [2 points]
        - The user will be able to see how much money is in their bank roll, updated with by most current gaming session [1 point]
        - Bank roll can only be viewed in between gaming sessions [1 point]
    - Place Bet Logic: [3 points]

        - Check to make sure input is positive integer only [1 point]
        - Display error message on invalid input [1 point]
        - Display error message when bet placed is larger than current bank roll [1 point]
    - Roll Dice Logic: [35 points]
        - The “come roll” [10 points]
            - Check to make sure there is a valid current bet [1 point]
            - Display error message if there is no valid current bet [1 point]
            - Display each dice rolled along with their sum [1 point]
            - Display the message when the user wins and how much they won [5 points]
            - Add both the winnings and their bet back to their bank roll [1 point]
            - Send user back to main game menu [1 point]
        - The “Craps out” roll [10 points]
            - Check to make sure there is a valid current bet [1 point]
            - Display error message if there is no valid current bet [1 point]
            - Display each dice rolled along with their sum [1 point]
            - Display the message when the user loses and how much they lost [5 points]
            - Subtract their losses their bank roll [1 point]
            - Send user back to main game menu [1 point]
        - The “point” roll [15 points]
            - Check to make sure there is a valid current bet [1 point]
            - Display error message if there is no valid current bet [1 point]
            - Display each dice rolled along with their sum [1 point]
            - Whenever the point is rolled, display the message that the user won and how much they won [5 points]
            - Add the winnings to their bank roll [1 point]
            - Continue until the user rolls a “7 out”, at which point display a message that the user has “7 out” and subtract their bet amount from their bank roll [5 point]
            - Send user back to main game menu when they “7 out” [1 point]
- Back to main game menu Logic: [15 points]
    - Take the current bank roll amount and update the users.txt file with the new bank roll to the appropriate user. [14 points]
    - Display the main menu [1 point]

- Quit Logic: [2 point]
    - Present the user with a message that the program is now over [1 point]
    - Stop the program [1 point]
- Code comments: [23 point]
    - You must use comments on all your code, in particular logic explaining functional tasks [10 points]
    - Each method must have a description of what arguments are passed and what data and data type is returned (if any) [13 points]
- Algorithm flow chart: [15 points]
    - You must submit a PDF outlining the flow of your program and include it in your zip file along with your source code.

About Submission

Late work will not be accepted, NO EXCPETIONS. You may not collaborate or share code with fellow students. You must submit your flow diagram as a PDF.

Use of AI of any kind will result in an IMMEDIATE ZERO on this project and may result in expulsion from this class.

Be prepared to explain your code upon request. Failure to do so will result in a zero for this project, no exceptions.