Assignment 1 - LRC Report

Jeremy Miller

CSE 13S - Winter 24

1 Purpose

This script simulates the game LRC. The user inputs a number of players between 3 and 10 and a random seed, then the game plays out until there is a winner.

2 Questions

2.1 Randomness

The randomness in this script is handled by the srandom() and random() functions. Using a random seed inputted by the user, srandom() function creates then possibilites of the random(), so that when two games are played with the same seed, the outcomes are identical.

2.2 Abstraction

Abstractions in programming refers to the different scopes within a script. These different scopes are created by the use of functions, as something defined inside a function cannot be altered from outside of the scope of the function.

2.3 Why?

Abstractions make debugging easier as it makes your code more readable and organized. It does this by compartmentalizing information based on their functions. In addition, if you have a problem with a function that gets called multiple times, you would only have to edit the function itself once and every instance of the function would work. If you didn't use a function, you would have to edit every instance of that particular block of code, which would cause a real headache.

2.4 Functions

Writing our code to use functions is a good idea as it makes debugging and readability much easier. In order to use functions in our main(), we would

have to declare them outside the function. I chose to write all of the function prototypes at the top above main(), then the full function declarations under main(). This makes it easy to open up the script, read the main() function right away, and understand what is happening. In my script I had 6 functions.

2.5 Testing

For this assignment, I want to test the output of my script to make sure the game works. To do this, I will test my script up against the provided lrc script. Since the number generation is pseudo random, I can run inputs "3 3" on both scripts and it should be identical. In the end, my script was not a perfect match, but I got as close as I could before the deadline.

2.6 Putting it all together

Using a pseudo random number generator (PRNG) and abstractions make debugging and testing a lot easier. PRNGs make testing easier because the randomness relies on a predetermined seed, therefore, whenever the game is played with a certain seed the output will always be the same. For example, the LRC game with a seed of 3 will always have the same results no matter how many times you play it. Abstractions make debugging easier because it makes your code easier to read and understand, as well as reducing the amount of duplicate code sections.

3 How to Use the Program

To compile the lrc.c source file, use the Makefile. Then run the program lrc. Once started, you will be required to input a number between 3 and 10 to indicate how many players are playing. You will then need to input a random seed for the game. Once both pieces of information are recorded, the game will automatically simulate, printing every turn. Once the game is over, the program will print out the winner.

4 Program Design

This script has several functions as well as a main() function that handles the main game play loop. At the top of the script all of the function prototypes are defined. Then there is the main() function, and under that the full logic for the each function.

4.1 Pseudocode

The script defines functions. Then in main, it starts off by getting the player count using scanf, and prints it. Then the function gets the random seed also using scanf. Then teh chips get setup and distributed. Once all this initialization

is done, the main game loop begins, which is a while loop. The condition is that the while the game isn't over, play the game. The game is a for loop that takes care of turn, and then each turn the player rolls a dice and the chips are changed accordingly. This continues until the while loop breaks and the game is over. Under the main function are all the logic for each function.

4.2 Function Descriptions

- **int getPlayerCount()** - no input, output: the amount of players playing the game based off the user input.
- **void getSeed()** - no input, output: the seed based off the user input, then sets that seed for the game.
- **void setupChips(int numPlayers, int chips[])** - input: amount of players, and the array of chip totals, no output
- **void playerTurn(int currentIndex, int numPlayers, int chips[], char roll)** - input: the index of the current player, the amount of players, the array of chip totals, and a string of the output of the dice roll, no output
- **int isGameOver(int numPlayers, int chips[])** - input: the amount of players, the array of chip totals, output: 1 or 0 depeding on if the game is over or not
- **char rollDice()** = no inputs, output: the result of the roll as a string