Assignment 1 – LRC Report

Caden Roberts

CSE 13S – Winter 24

Purpose

Audience for this section: Pretend that you are working in industry, and write this paragraph for your boss. You are answering the basic question, "What does this thing do?". This section can be short. A single paragraph is okay.

Do not just copy the assignment PDF to complete this section, use your own words.

ANS:

The purpose of the code will be to stimulate the game Left, Right, and Center. A dice of 6 sides will be rolled for x players who all start with 3 chips. A roll of 1, 2, and 3 will be a "dot", where the player keeps their chip. 4 will be "left", having the player give the chip to the player to the left, 5 will be "center", where the chip is discarded into a "pot", and 6 will be "right", having the player give the chip to the player to the right. If a player has no chips they may not roll the die when it is their turn. When only one player has chips the game is over and won. We will use pseudo random numbers and C code to do this stimulation.

Questions

Please answer the following questions before you start coding. They will help guide you through the assignment. To make the grader's life easier, please do not remove the questions, and simply put your answers below the text of each question.

Randomness

Describe what makes randomness. Is it possible for anything to be truly random? Why are we using pseudorandom numbers in this assignment?

ANS:

The numbers are not truly random, but pseudo random. This has to do with specific seeding of the random functions built into the C language. This is useful for this assignment because exact output can be expected, something not possible with truly random variables.

What is an abstraction

When writing code, programmers often use "abstractions". Define an abstraction in non computer science terms (Don't google it!)

ANS:

An abstraction in computer science is a user created variable, an enumeration.

Why?

The last assignment was focussed on debugging. How can abstractions make debugging easier? What other uses are there for abstractions? Hint: Do you have to be the one to write the abstraction?

ANS:

Abstractions can be used to model and test complex problems, which is similar to the definition of debugging. You must write your own debugging test cases, and you must write your own "abstractions".

Functions

When you write this assignment, you can chose to write functions. While functions might make the program longer, they can also make the program simpler to understand and debug. How can we write the code to use 2 functions along with the main? How can we use 8 functions? Contrast these two implementations along with using no functions. Which will be easier for you? When you write the Program design section, think about your response to this section.

ANS:

Functions will be useful in cases where a series of code needs to be written in multiple places in the main function. Creating 2 functions will allow simplification of a dice roll and a chip move. Creating 8 functions will allow extreme specification of the dice roll and where the chip moves. Using no functions will require writing a massive main function that handles all the logic of the game. Using 2 functions seems to be the best option, containing most of the logic flow in main and improving readability. However, with a task as simple as this, a single main function may be a reasonable approach.

Testing

The last assignment was focused on testing. For this assignment, what sorts of things do you want to test? How can you make your tests comprehensive? Give a few examples of inputs that you will test.

ANS:

We will want to test the seeding of the pseudo random numbers as well as edge cases of players and different valid and invalid inputs.

Putting it all together

The questions above included things about randomness, abstractions and testing. How does using a pseudo random number generator and abstractions make your code easier to test?

ANS:

Exact output can be expected and tested for when using set abstractions and pseudo random numbers.

How to Use the Program

Audience: Write this section for the user of your program. You are answering the basic question, "How do I use this thing?". Don't copy the assignment exactly; explain this in your own words. This section will be longer for a more complicated program and shorter for a less complicated program. You should show how to compile and run your program. You should also describe any optional flags that your program uses, and what they do.

ANS:

The program can be ran with any C compiler, for example repl.it can easily run this program. If we want to run it from our own terminal/VM, we will run the following:

```
cadenroberts@cse13svm:~/resources/asgn1$ ls
'asgn1 design template.pdf' lrc     lrc.o     names.h
'asgn1 design template.zip' lrc_arm lrc_x86   runner.sh
 asgn1.pdf                 lrc.c    Makefile  tests
```

First, we ls in the folder containing our lrc.c file to confirm it and the Makefile are in the same directory. In this case, the lrc executable is already present. However, if it wasn't we run:

```
cadenroberts@cse13svm:~/resources/asgn1$ make
clang -Wall -Wextra -Wstrict-prototypes -Werror -pedantic -c lrc.c
clang -Wall -Wextra -Wstrict-prototypes -Werror -pedantic lrc.o -o lrc
# bash runner.sh
```

Now, we can simple run the executable as follows, and give inputs 3 and 4823, for example:

```
cadenroberts@cse13svm:~/resources/asgn1$ ./lrc
Number of players (3 to 10)? 3
Random-number seed? 4823
Ada Lovelace: ends her turn with 2
Margaret Hamilton: ends her turn with 0
Katherine Johnson: ends her turn with 3
Ada Lovelace: ends her turn with 1
Margaret Hamilton: ends her turn with 0
Katherine Johnson: ends her turn with 2
Ada Lovelace: ends her turn with 3
Katherine Johnson: ends her turn with 1
Ada Lovelace: ends her turn with 2
Margaret Hamilton: ends her turn with 1
Katherine Johnson: ends her turn with 1
Ada Lovelace: ends her turn with 2
Margaret Hamilton: ends her turn with 1
Katherine Johnson: ends her turn with 1
Ada Lovelace: ends her turn with 1
Margaret Hamilton: ends her turn with 0
Katherine Johnson: ends her turn with 2
Ada Lovelace: ends her turn with 2
Katherine Johnson: ends her turn with 2
Ada Lovelace: ends her turn with 0
Margaret Hamilton: ends her turn with 0
Katherine Johnson won!
```

If we would like to test the class program, we can run the executable lrc_arm (for M2 users). If we would like to test our executable against the class executable with a series of user-defined tests, using the runner.sh present in the asgn1 directory, that looks like this:

```
cadenroberts@cse13svm:~/resources/asgn1/tests$ ls
test_0seed.sh      test_bigseed.sh test_inputdefault.sh test_overedge.sh
test_badletterin.sh test_edge.sh   test_negin.sh       test_smallseed.sh
cadenroberts@cse13svm:~/resources/asgn1/tests$ cd ..
cadenroberts@cse13svm:~/resources/asgn1$ bash runner.sh
PASS:  test_bigseed
Message:
Working Correctly
PASS:  test_inputdefault
Message:
Working Correctly
```

```
PASS:  test_smallseed
Message:
Working Correctly
PASS:  test_negin
Message:
Working Correctly
PASS:  test_edge
Message:
Working Correctly
PASS:  test_overedge
Message:
Working Correctly
PASS:  test_badletterin
Message:
Working Correctly
PASS:  test_0seed
Message:
Working Correctly
```

Program Design

Audience: Write this section for someone who will maintain your program. In industry you maintain your own programs, and so your audience could be future you! List the main data structures and the main algorithms. You are answering the basic question, "How is this thing organized so that I can have a chance of fixing it?". This section will be longer for a more complicated program and shorter for a less complicated program.

ANS

We use an enumeration and arrays, along with several integer variables to help with logic flow. The game, after gathering inputs num_players and seed, essentially is:

```
bool winner=false;
int n=0, win=0;
while (!winner) {
        int rolls = player[n];
        if (rolls>3) rolls=3;
        if (rolls>0){
                while(rolls-- > 0){
                        Position roll = die[random() % 6];
                        if (roll == LEFT){
                                if(n+1<num_players) player[n+1]++;
                                else player[0]++;
                        }
                        else if (roll == DOT) player[n]++;
                        else if (roll == RIGHT){
                                if(n==0) player[num_players-1]++;
                                else player[n-1]++;
                        }
                        player[n]--;
                }
        printf("%s: ends her turn with %d\n", player_name[n], player[n]);
        }
        if (n+1<num_players) n++;
        else n=0;
        int checker = -1;
        for (int i=0; i<num_players; i++){
                if (player[i]>0){
                        checker++;
                        win=i;
                }
        }
        if (checker == 0) winner=true;
}
printf("%s won!\n", player_name[win]);
```

The main algorithm is the while(rolls greater than 0) loop that rolls and does the chip shifts. The game is contained within the while(not a winner) loop.

Pseudocode

Give the reader a top down description of your code! How will you break it down? What features will your code have?

ANS

The psuedocode for the game is as follows:

```
Gather players and seed input
While no one has won:
    While current player has rolls left:
        Roll the dice
        Move the chip accordingly
    Print player and how many chips they have
    Move to the next player
    Check if someone has won
Print winner
```

Function Descriptions

For each function in your program, you will need to explain your thought process. This means doing the following

- The inputs of every function (even if it's not a parameter)
- The outputs of every function (even if it's not the return value)
- The purpose of each function, a brief description about a sentence long.
- For more complicated functions, include pseudocode that describes how the function works
- For more complicated functions, also include a description of your decision making process; why you chose to use any data structures or control flows that you did.

Do not simply use your code to describe this. This section should be readable to a person with little to no code knowledge.

ANS

I did not write any separate functions for this game apart from main. Initially, an array of 10 integers ”players” starting with value 3 ”chips” is created regardless of integer input ”num_players”, which will be used to control the max index we look at within that array of 10 players. Integer ”seed” is created as 4823. Both ”num_players” and ”seed” will be turned to their initial values if bad input is given. The main logic is handled by a while loop that uses the number of rolls a player has as ”input” and rolls dice using random(), moving chips accordingly using if statements which increment the current player in the array(dot), or the player before (right) or after (left). Center is not checked for, then the current player is decremented in all cases. When this loop ends, printing of the current player and the chips they ended with occurs. The next ”function” is a for loop that checks through the array of players to see if only 1 has chips left. An integer ”checker” of -1 is created and incremented if a player[i] is greater than 0, and an integer ”win” is updated to that index, and used in the final player win statement after the main while loop ends. ”checker” is checked to be exactly 0 after the loop finished, in which case boolean ”winner” will be turned true. ”winner” is passed to a ”function” while loop that is controlling player shift and the dice roll ”function” while loop.