Merilyn Kuo
CSE13S/L
Spring 2021

Assignment 1: Left Right and Center
Design Document

Introduction:

Left, Right, Center is a dice game. Every player starts with $3 at the beginning of the game. Players in the game roll a number of dice based on the amount of money they have. The maximum number of rolls a player can do is three. If the player has no money, they cannot roll until they receive money. Based on each player's roll or rolls, players in the game can lose or gain money. The possible outcomes a player can roll are left, right, center, or pass. Three of the dice faces are pass. Rolling left means the player gives a dollar to the person on their left. Rolling a right means the player gives a dollar to the person on their right. Rolling center means the player puts a dollar in the pot. Rolling a pass keeps the player safe. The game ends when only one player has money left and the player then wins the money in the center.

Provided Code Used:

For the following, the code was provided to us for use and can be found in the Assignment 1 program specification.

- Array of die faces
- Function to determine the position of the player on the left
- Function to determine the position of the player on the right
- philos.c (which contains names of the players as a character array)
- Makefile code to build our program.

Initial Setup For Game Program:

Define:
- maximum number of rolls
- amount of money each player starts off with

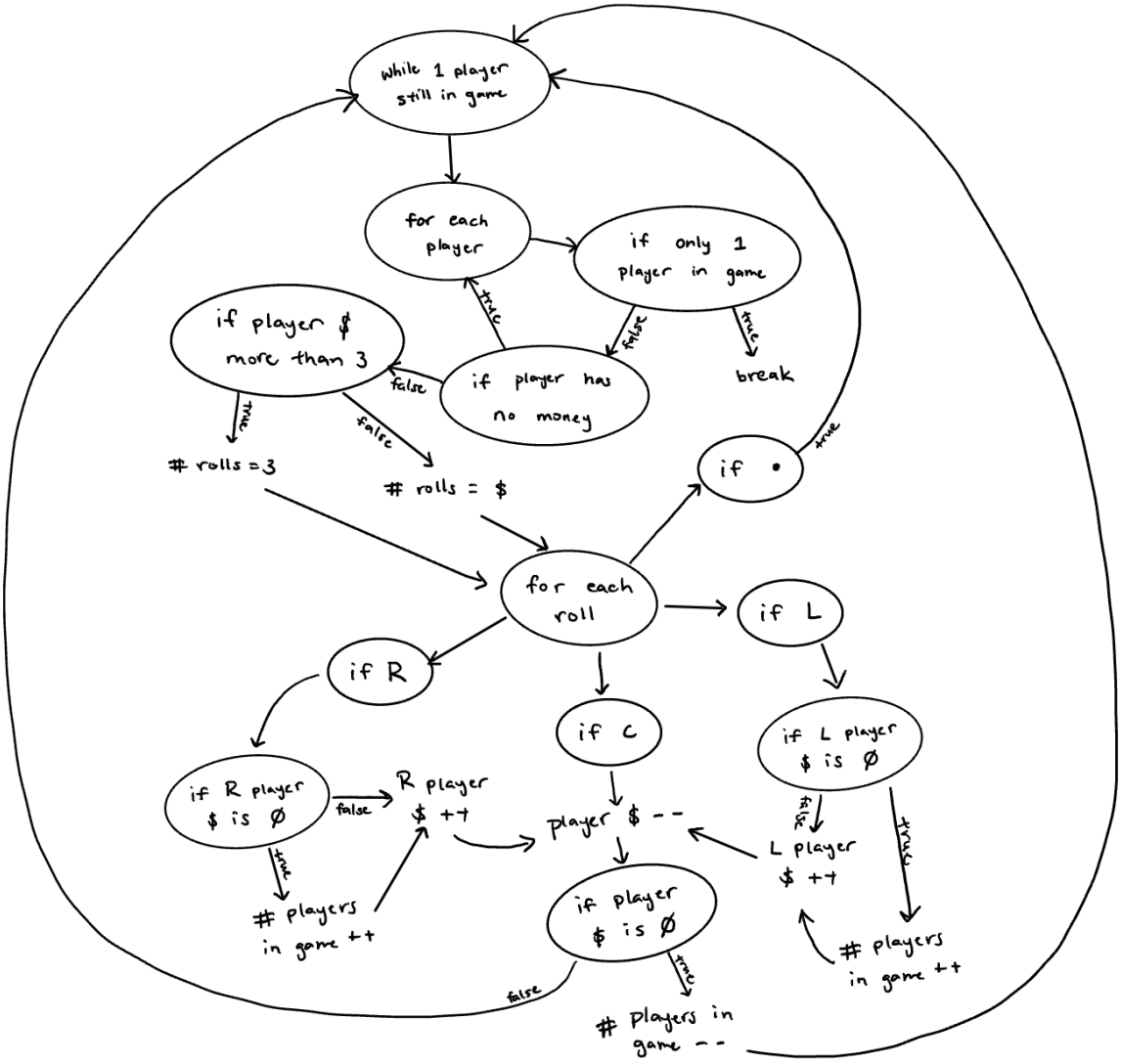
Initialize:
- Integer array for amount of money each player has
- Integer variable number of rolls
- Integer variable random number generated
- Integer variable for number of players at the start of game
- Integer variable for number of players in the game
- Integer variable for amount of money in the center pot

Concepts Or Topics Used:

The main part of the game, after any necessary initial set up, is a loop that runs until there is one player left in the game. In my program I used a while loop. Within this loop, I also implemented two nests for loops, where one for loop is inside the other.. The outer for loop iterates through each of the players in the game and the inner for loop iterates through the number of rolls each player gets. Upon the exit of the main while loop, there is one more for loop to find the player that has money left. I also implemented several if statements and if/else statements in my program. I used if statements to check how many rolls a player has, if a player has no money, and if there is only one player left in the game. I used a series of if and else if statements to represent the rolling of the dice and the result(s) of the outcome(s).

Top Level Diagram:

This is the flow chart diagram I drew when thinking about the design of my program.



Pseudocode:

This pseudocode was inspired by vampires.c code the professor posted.
Please note that there are print statements implemented in the program that describe the simulation of the game to the user. This part was not included in the pseudocode. Also, a small portion of the pseudocode was developed with tutor Mary.

The pseudocode is as follows:

```
Take in user input for # of players and random seed

Number players in game = numbers of players user inputed

While # players in game > 1

      For each player:
            If players in game == 1:
                  Break

            If money ==  0:
                  Continue to next player

            Int rolls = money
            If rolls > 3:
                  Rolls  =3

            for (i = rolls; i  > 0; i - -)
                  Roll dice (generate random number)

                  If random # corresponds to index of L in die faces array:
                        Current player $ minus 1

                        If player $ == 0:
                              # players in game - 1

                        Get player on left (using function)
                        If player on left $ == 0:
                              # players in game + 1

                        Player on left $ plus 1

                  Else If R:
                        Current player $ minus 1
                        If player $ == 0:
                              # players in game - 1

                        Get player on right (using function)
                        If player on right $ == 0:
                              # players in game + 1
```

```
                        Player on right $ plus 1

                  Else if C then
                        Current player $ minus 1
                        If player $ == 0:
                              # players in game - 1

                        $ in center plus 1

                  Else if P:
                        Do nothing

Loop through money players have array
      Find which player still has money
            Print name and amount they win from center
```

Process and Changes to Design:

TBD

Credits and Resources Used:

TBD