Assignment 1 – Pass the Pigs

Noah Meisel

CSE 13S – Spring 2023

Purpose

Pig.c is a simplified version of the dice game called ”Pass the Pigs”. The game plays with each player rolling a pig-shaped die to get points. If a player rolls the pig so it lands on its side, the turn passes to the next player. This cycle continues until someone gets to one hundred points.

This program replicates this game structure utilizing the random function to simulate a random roll of the pig.

How to Use the Program

Utilizing the pig.c program is quite simple. Run the program in the terminal and it will prompt you to enter a number of players and then a seed value. The number of players is exactly what it sounds like, the number of people who the program will simulate to be playing the game, and the seed value is the value that determines which value the random function will start with.

The program will then run through an entire game of Pass the Pigs, printing the name of the player who is rolling as well as the the subsequent rolls of that player. Once a winner is found, the program ends.

Program Design

This program has a simple two loop design to create the structure of a continuous game with looping turns inside that larger game cycle. Before the loops begin, however, the program has multiple short code blocks that captures the number of players and the seed value the user is choosing, along with an array of 0's that represent the scores of each player. An enumerated list of values that correspond to the different roll possibilities, and a corresponding array to replicate their respective probabilities of being rolled, is created as well. This was done using code given in the CSE13S Assignment 1 project description.

The flow of the main game is as follows

```
While game is not over, keep playing:
    Print the name of the current player
    Roll the dice and determine how many points the player gets.
    Check if the roll breaks us out of the loop. If it does not, go to the next players turn and repeat
```

Data Structures

The only data structures used in this program are arrays. These are already implemented by default in C.

Algorithms

No particular algorithm's were used in this program

Function Descriptions

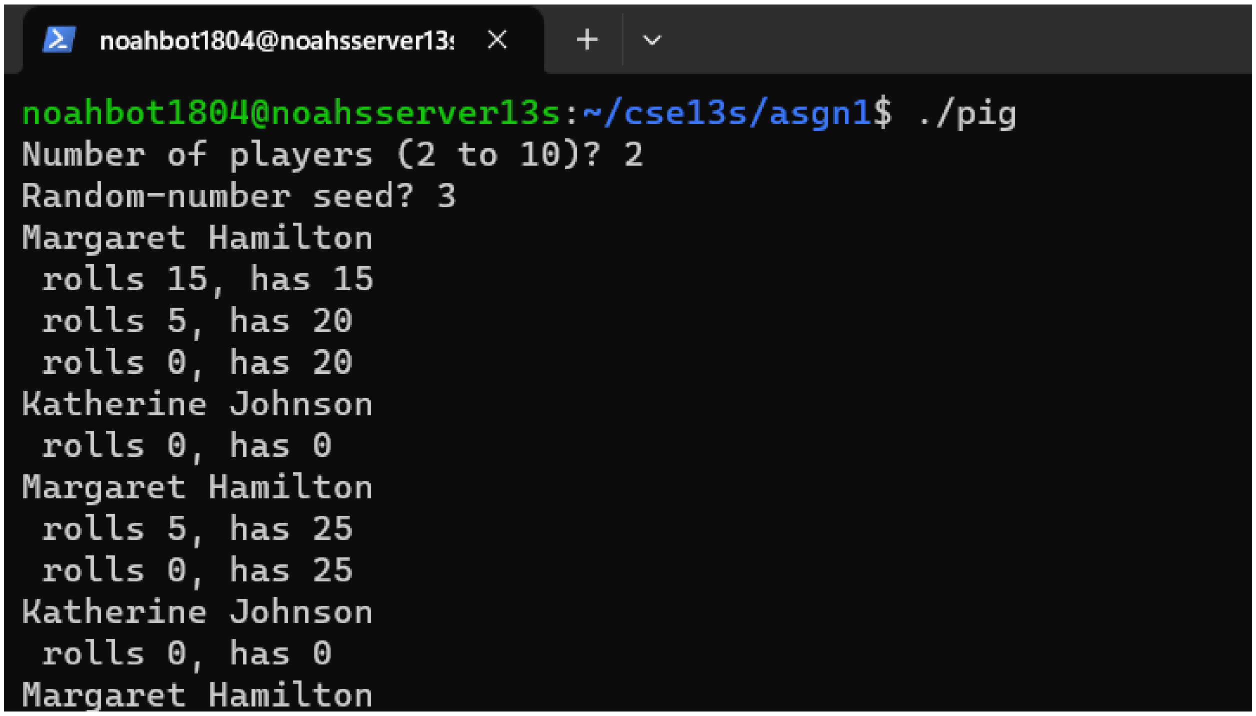
Two custom functions are used in this program; is_game_over and dice_roll.

- is_game_over takes two integers as its inputs: turn number modulus the number of players and the score of the current player. dice_roll takes in a number between 0 and 6, and the current players score
- is_game_over returns either the number 1 or 0. 1 represents the game being over. dice_roll outputs the score that the new player receives, and prints the roll and new score to the console.
- is_game_over checks whether the game is over, and returns true or false depending on this question. It determines this by seeing if the current players score is greater than or equal to 100. If this is true the program will set the return value to 1 instead of 0, as well as store the index of the current player in a global variable winner_index that can be used in the main game loop to print the winner. dice_roll simulates the roll of a pig, and returns the the score that this roll provides. this function consists of a moderate if-else tree that checks the rolls numeric value, as can be found in the array of different position, and depending on the value prints the rolls score value, the players score, and returns the score of this roll so it can be used outside of the function to update the players score.

Results

Overall the program runs as I intended. The do-while loops structure of the inner game loop works very well for this type of game, as each player is guaranteed to go once each round and then every roll after that is conditional on their previous roll.

One area of my code that could definitely be improved is the dice_roll function. It's rather messy because it's implementation is done with an if-else tree, which works but is likely not the most elegant solution. I also used a lot of hard-coded values for the dice rolls because I was having issues with function scope and couldn't figure out, in a short period of time, how to edit arrays inside separate functions. I am relatively confident the solution involves pointers, but don't remember the proper implementation.



```
noahbot1804@noahsserver13s:~/cse13s/asgn1$ ./pig
Number of players (2 to 10)? 2
Random-number seed? 3
Margaret Hamilton
 rolls 15, has 15
 rolls 5, has 20
 rolls 0, has 20
Katherine Johnson
 rolls 0, has 0
Margaret Hamilton
 rolls 5, has 25
 rolls 0, has 25
Katherine Johnson
 rolls 0, has 0
Margaret Hamilton
```

Figure 1: Screenshot of the program running.

Bibliography

Kernighan, Brian. Ritchie, Dennis. The C Programming Language, 2nd Edition. Murray Hill, New Jersey, 1988.

Long, Darrel. Veenstra, Kerry. asgn1.pdf, Santa Cruz, California. 2023.