Assignment 1 – Pass the Pigs

Eric Lee

CSE 13S – Spring 2023

Purpose

This program plays the game 'Pass the Pig'. This game can be played with 2 to 10 players and involves players rolling a asymmetrical dice, or the 'pig', to earn points. There are 5 total sides to the pig, with the Jowler and Side having a probability of 2/7 to land and the rest having a probability of 1/7. There are also different points earned for each side. All sides let the player roll the pig again except for 'Side', which moves the dice to the next player. The game ends when a player gets 100 points.

How to Use the Program

To play the game, the user is first prompted to enter the number of players between 2 and 10. The program will return an error and use the default value of 2 players if the user doesn't enter a valid input. Then the user enters the random seed, which again will return an error and will use default value 2023 with invalid input. The program will then print out the results of the game of all players until the game is over.

Program Design

Data Structures

In order to provide an abstraction of the 'pig' in the program, I used enumeration to have a constant defined for each side. Then, the constants defined in the enumeration are used in an array of size 7 to provide an abstraction of the different odds each side has of landing. I also use an array to keep track of each player's points. Each player is represented by an index in the array that is updated when they earn points. I also use many variables for holding data like the number of players, the random seed, and which player is currently rolling the pig.

Algorithms

The overall game part works with a while loop that has a switch statement nested inside. The while loop runs as long as none of the players have won yet, and the switch statement figures out what happens for each side possible.

This is the first case for if the player rolls a 'Side'. The array 'pig' holds all possible sides of the pig. The integer 'roll' has the random number that has been selected

```
switch(pig[roll]){
       case SIDE:
           printf(" rolls 0, has %d\n", players[turn]);
               if (turn == (num_players-1)){
                   turn = 0;
               }else{
                   turn = turn + 1;
               }
```

```
                printf("%s\n", player_name[turn]);
                    break;
```

This for-loop sets each players score to '0' before the game starts playing.

```
int players [num_players];
       for (int i = 0; i < num_players; i++){
               players[i] = 0;
       }
```

Function Descriptions

srandom():

- It takes the input of the user-inputted seed .
- It is void and has no output.

main():

- It takes no inputs and outputs 0.
- It first creates an array to hold a list of all the possible sides the pig can land on.
- Then it will ask the user for the number of players and random seed and output an error message if need.
- Then it creates an array that holds a list of each player's points and is created using the user inputted number of players.
- The game itself runs in a while loop that keeps the game running as long as none of the players have reached 100 points.
- With each loop, a random number is generated, and then the case in the switch statement that matches that side recalculates the player's new point total and prints it out.
  - The switch statement was an easier route than a lot of if-else statements and could use the 'pig' array already created

Results

The code successfully runs the game ever time, and matches the output of the tester code. I think it could also use a buffer so that the entire game isn't printed at once in the future. Other than that, I do not think anything else is lacking.