Dmitrius Agoston
Prof. Darrell Long
3 October 2021

CSE 13S Fall 2021
Assignment 1: Pass the Pigs
Design Document

Description of Program:
The purpose of this program is to simulate David Moffat's dice game *Pass the Pigs*. This consists of asking the user for the number of players (between 2 and 10) as well as a random seed for input to determine dice rolls. The dice rolls are to simulate a pig landing in various positions, and receiving points based on said positions. Rolling Side yields 0 points and the end of the players turn. Rolling Snouter yields 15 points, Razorback or Trotter yields 10 points and rolling Jowler earns 5 points. The game is over once a player has reached a total score of 100 points or greater.

Layout/Pseudocode:
The Libraries/files that should be included in this program are names.h, stdbool.h, stdio.h, stdlib.h

The layout of this program is built in a way where the user's input is prompted first. The user will first be prompted for the number of players. The input has to be an integer between 2 and 10, if not, 2 will be used as a default value.

```
Create variable for amount of players
Players set equal to user input
If user input is less than 0 or greater than 10 or a non integer
    Print “Invalid number of players. Using 2 instead”
    Players set equal to 2
```

The user will then be prompted to input a seed for the dice rolls. If the input does not meet the requirements for a seed, the default seed 2021 will be used instead.

```
Create variable for seed
Seed set equal to user input
If user input is not a suitable seed
    Print “Invalid random seed. Using 2021 instead.”
    Seed set equal to 2021
Input seed value into srandom
```

After this the positions are defined that will be used during the game through enumerations. This allows names to be provided for integers that will be used later.

```
Enumerators created as Side, Razorback, Trotter, Snouter, Jowler, and named position
Constant position named pig with 7 values assigned
Side, Side, Razorback, Trotter, Snouter, Jowler, Jowler
```

Then the program should create player totals in an array based on the number of players input.

```
Create array for players scores
For the number of players
    Create player total 0, 1, ...
```

With the players and sides defined the dice will be rolled for player 0 until they either reach 100 points or the pig lands on one of its sides. While this happens there will be two if statements checking if it is the next player's turn or if the current player has won. The rolls will be determined through the previously defined enumerators and the values assigned with the position named pig. There will be a total of five cases implemented through a switch statement for the rolls though the likelihood of rolling a “side” or “jowler” has a 2/7 chance in comparison to the others that only have a 1/7 chance of being rolled. This process will continue until a player triggers the if statement looking for a player's points to reach or exceed 100 points. Once triggered the player will be given a congratulatory message and the game will be over.

```
Create and set variable for current player to 0
Create and set variable for previous player to -1
While true
    If current player’s points is greater than or equal to 100
        print congratulatory message
        Break from while loop
    If the value of current player does not match previous player
        Print current player rolling
        Set previous player equal to current player
    Switch statement with (pig[random number mod 7])
    Case 0 (side case)
        Print pig lands on side
        Increment current player value by 1
        Break from switch statement
    Case 1 (razorback case)
```

```
        Print pig lands on back
        Add 10 points to current player’s total
        Break from switch statement
    Case 2 (trotter case)
        Print pig lands upright
        Add 10 points to current player’s total
        Break from switch statement
    Case 3 (snouter case)
        Print pig lands on snout
        Add 15 points to current player’s total
        Break from switch statement
    Case 4 (jowler case)
        Print pig lands on ear
        Add 5 points to current player’s total
        Break from switch statement
```

Error Handling:
The main two places that I had to watch for errors was the user input for the number of players as well as the input for a random seed. For the player output I initially was going to make a case for if the user input was a non number like, “abc”, for example. Though when I went to the code with the correct output it appeared that if a user input a non number the program is supposed to use the default value of 2 as well as the default seed. Other than that error I only had to make sure the number of players was between 2-10. For the seed I ended up only checking if it was less than 0 since srandom takes an unsigned integer as the argument. My reason for not having a check for the max value is because the maximum positive value of a signed integer is less than what the bounds of an unsigned integer can handle. To be safe I even tested my code with the numbers 2,147,483,647 and 2,147,483,648. When these numbers are used as an input the first one works as a valid seed and the second will trigger the default seed to be used since the second is greater than the maximum positive signed integer can handle. That causes scanf to return a negative number and trigger the if statement.