

## Assignment 1 - Pass the Pigs

Teresa Joseph

CSE 13S - Professor Long

Fall 2021 - RD September 30/ FD October 3

### Purpose

Implement a game of “Pass the Pigs” where players take turns rolling a pig to score points.

### Game Breakdown

- k number of players (2 min and 10 max)
- Each player rolls a pig of 5 possible positions to gain points
  - 0 points for side (2/7 probability)
  - 5 points for jowler/ear (2/7 probability)
  - 10 points for razorback/back (1/7 probability)
  - 10 points for trotter/upright (1/7 probability)
  - 15 points for snouter/snout (1/7 probability)
- Continue rolling until side is rolled (then pass to next player) or reach 100 or more points (then win and end the game)

### Pseudocode

Libraries/files to include

<stdio.h> for general commands

<stdlib.h> for random generator

<limits.h> for seed range

“names.h” for accessing player names

Main function:

[Obtaining the number of players]

Prompt user to enter the number of players and label the input as *players*

If *players* is less than 2 or greater than 10:

Print error message and assume 2 players (so *players* = 2)

Else, continue with *players* = user’s input

[Obtaining the seed value]

Prompt user to enter a valid unsigned seed value and label the input as *seed\_value*

If *seed\_value* is greater than unsigned int max ( $2^{32}-1$  or `UINT_MAX`):

Print error message and assume value of 2021 (so *seed\_value* = 2021)

If *seed\_value* is less than unsigned int min (0/indicates negative sign):

Print error message and assume value of 2021 (so *seed\_value* = 2021)

Else, continue with *seed\_value* = user's input

[Creating arrays for the pig roll]

Create array of ten 0s (the 0s represents each player's initial points, and there are 10 elements to represent the max number of players), call array *points*

Create another array that enumerates the 5 positions (side, jowler, razorback, trotter, and snouter), call it *Positions* (as stated in assignment document)

Create another array of the roll possibilities (side, side, jowler, jowler, razorback, trotter, and snouter), call it *pig* (as stated in assignment document)

[Simulating the pig roll]

Set `random()` to *seed\_value* (similar to example in assignment document)

Set counter for player turn (calling it *current\_player*) to 0

While *current\_player* is less than *players*/the total number of players (up to last player's:

Print corresponding player name from `names.h` and "rolls the pig"

Randomly select value from *pig* with `random() % 7` -> represents roll

While *pig*[roll] is not equal to side:

If *pig*[`random() % 7`] equals jowler:

Print that the player rolled jowler

Increment *points*[*current\_player*] by 5

If *pig*[`random() % 7`] equals razorback:

Print that the player rolled razorback

Increment *points*[*current\_player*] by 10

If *pig*[`random() % 7`] equals trotter:

Print that the player rolled trotter

```

        Increment points[current_player] by 10
    If pig[random() % 7] equals snouter:
        Print that the player rolled snouter
        Increment points[current_player] by 15
    If current player's points are 100 or more:
        Break out of this while loop
    Otherwise, roll again and go back to the start of this while loop
Check again if current player's points are 100 or more:
    Print name of winner and their points using the current index
    Break out of the outermost while loop if so
If pig[random() % 7] equals side:
    Print that the player rolled side
    If i equals current_player-1:
        Set current_player back to 0 (essentially starts a new round)
    Else:
        Increase current_player by 1 (moves to next player)
End the main function

```

### **Notes**

- “random() % 7” makes sure that the randomly generated values are between 0 and 6 (where each number corresponds to a position in *pig* according to its index)
- Checking if the input is an integer might not be necessary (scanf should take care of this)

### **Overall Description**

The code I created is structured so that my main function has everything I need. First, I prompt the user to enter the number of players, making sure that it satisfies the requirements described in the assignment document. I did the same with the random seed, prompting the user to enter a value and ensuring that it fits the necessary conditions.

Then, I created the arrays that I need. The array called *points*, which will be initialized to have *players* amount of zeros, will keep track of each player's points as the game progresses. As

the assignment document suggests, I also enumerated *Positions* and created an array called *pig* that represents the probability of each pig position (side, jowler, razorback, trotter, and snouter).

Next is the biggest portion of my code: the pig rolling simulation. This would require setting up `srandom()` with *seed\_value* and then entering a loop where each player takes turns rolling the pig. Though I originally wanted to use a for loop to keep track of each player's turn, I realized it would be much simpler and easier to follow along with a while loop. Ultimately, I restructured my code to do this. This loop allowed me to alter the iterator and restart the round as needed (namely when a player rolled side). In this loop, I created another while loop where I would randomly select a value with `srandom()` and had it represent the index of the *pig* array, thereby randomly selecting a pig position. I then checked to see which of the five pig positions it is and incremented the points array accordingly (by 0, 5, 10, or 15 points). I also checked to see if the points for each player after each roll is greater than or equal to 100. If this is true, I will break the two loops and state them as the winner. Otherwise, I will continue the loop until the winner is decided upon.

### **Simple Summary**

- Function: `main(void)`
- Inputs: number of players and random seed
- Variables: *players* (number of players) and *seed\_value* (random seed)
- Outputs: names of players as they take turns, which positions their rolls resulted in, and who ultimately won the game
- Process: prompt for both inputs, create arrays, simulate the pig rolls, and end the game

### **Goals/Intended Process**

- Set up this pseudocode in C
- Address possible errors
- Swap loops for more efficient methods once running properly
- Add sufficient comments and clean up format

## Other - Visual Representation of Roll

