asgn1: DESIGN.pdf - Pass the Pigs

Serjo Barron

September 30, 2021

Program:

This program simulates a simplified version of David Moffat's *Pass the Pigs* game where the user can decide the number of players (2 to 10 inclusive) and SEED the game will use to pseudo-randomly decide the rolls, and outcome (winner) of the game. Each player begins with a total of 0 points and wins when they reach or go above 100. The number of times a player can roll is infinite but ends when that current player lands a side. If each player has taken a turn, the program will return to the initial player (player 0) and continuously cycle until someone wins.

Rules of the game:

1. There are 5 ways for the pig to land (imagine it as an asymmetrical dice): side (2), back (1), upright (1), snout (1), and ears (2). With the (number) indicating if that particular land has more than one instance, thus meaning the sample space of the pig is 7.

2. Point allocation:
➢ rolling a side ends the current players turn, and rewards 0 points
➢ rolling back or upright rewards 10 points
➢ rolling snout rewards 15 points
➢ rolling ears rewards 5 points

3. The game ends when a player reaches a total of 100 or more points.

Needed files in asgn1 directory:

- names.h
  - ➢ A header file containing an array of length 10 which holds 10 names (of the players) which is to be included in the pig.c file. Names are assigned via index, so player 0 will always be Wilbur, player 1 will be Charlotte, and so forth.
- pig.c
  - ➢ The source file that contains the majority of the C code to run the game *Pass the Pigs*
- Makefile

    - ➢ A file that will format the code (pig.c) into clang-format and compile all the required files (pig.c and names.h) into an executable.
- README.md
    - ➢ A text file in markdown that will describe the program (what it does), the instructions of how to build and run the program, and how the program handles erroneous inputs.
- DESIGN.pdf
    - ➢ Describes the design of the program (the current pdf you are reading)

Pseudocode and Structure:

Get user input for the number of players (must be between 2 and 10 inclusive)
Get user input for random seed (must be between 32-bit unsigned integer)

Start off each player with 0 points

While there is no winner:
    Cycle through the players, starting with players[0]:
        Player rolls the pig... (to land on side, back, upright, snout, or ear)
            While there is no side_roll:
                Switch case for result of the players roll:
                    Case 0:
                        Pig lands on side
                    Case 1:
                        Pig lands on back
                    Case 2:
                        Pig lands upright
                    Case 3:
                        Pig lands on snout
                    Default:
                        Pig lands on ear

                If the player's score is ≥ 100 the player wins and the program terminates.

Functions:

- srandom(SEED) and random()
    - ➢ srandom(SEED) sets its argument as the SEED (an unsigned integer) to create a sequence of pseudo-random integers which will be returned when calling the random() function.

    - ➢ The sequences are repeatable for the function srandom() for as long it is called with the same SEED value.
- typedef enum{ SIDE, RAZORBACK, TROTTER, SNOUTER, JOWLER } Position; const Position pig[7] = { SIDE, SIDE, RAZORBACK, TROTTER, SNOUTER, JOWLER, JOWLER };
    - ➢ The following enumeration was provided in the corresponding asgn1 specifications (© Prof. Long). “Position” represents the type of the first array containing all 5 possible rolls being SIDE, RAZORBACK, TROTTER, SNOUTER, and JOWLER. The second array “pig[7]” is a constant array containing the full sample space of possible outcomes, which is why SIDE and JOWLER appear twice.

Variables:

- players
    - ➢ Assigned by a user-inputted value after the message “How many players? “, the value must be between 2 and 10 (inclusive).
- SEED
    - ➢ Assigned by a user-inputted value after the message “Random seed: “, the value/valid SEED must be an unsigned integer.
- scores[players]
    - ➢ scores[players] is an array of length “player” which each value in that array will be initialized to 0 via a for loop indicating that each player begins with 0 points.
- side_roll
    - ➢ Keeps track of whether the current player rolls a “side” on the pig, which if so, the current player's turn will end.
- roll
    - ➢ roll is the variable assigned to random(), which is used to pseudo-randomly generate a number imitating the player rolling the pig.
- result
    - ➢ The result is calculated by taking the pseudo-randomly generated number “roll”, and taking the modulus of it by 7 (roll % 7). The result/remainder will always be an integer ranging from 0 to 6 inclusive, representing the 7 possible outcomes of the player rolling the pig.
    - ➢ (ex: SEED = 2021, with the first 3 pseudo-randomly generated numbers)
      1644198542 % 7 = r0
      653251166 % 7 = r1
      273593510 % 7 = r1

Credit:

- I attended Brian's tutoring session on 9/27/21 and he pointed out that my original plan of using the enumerations in if conditionals (comparing the enum value to the result of the roll) was not the "clear/easier method" of designating what the player rolled. To which he encouraged me to look more closely at the enumeration of the pig and think about how to take those special values and use them to print a string. To which I then looked more into the flow-control of C and decided to use switch cases as opposed to if and if-else conditionals.