Assignment 1 Design – Jack Vento (1816097)

The Game

Left, Right, Center is traditionally played at a table with three die where players take turns in a clockwise manner. They will roll all three die if they have $3 or more or only 1 or 2 die if they hold less than that. Depending on their rolls, they will either give money left, right, or center (the pot) or pass. This cycle continues until only one player has money left where they then win the pot.

The program will have three main sections:

- Introduction: initialize variables, allocate memory, get the seed + number of players, check that those inputs are valid
- Gameplay: loop clockwise where each player rolls dice 0<x<4 times depending on their currency, transfer their money left, right, or center if they didn't roll passes, update all affected balances, take any players out who don't have money, and then find the next player whose turn it is.
- Conclusion: print the winning message and free any dynamic memory used.

Variables

- Currency: *pot* (money in the pot), *balances* (dynamic array of uint32_t holding each player's currency).
- Players: *num_players* (total players), *current_player* (player currently rolling), *players_left* (players still with currency),
- Misc: *seed* (the random seed), *names* (array of predetermined names), *die* (array of possible rolls), *die*_rolls (amount of times current_player will roll), die_*result* (enum result from rand()).

Pseudocode

Helper Functions

left(pos, players)
Output: Position of the player to the left.

right(pos, players)
Output: Position of the player to the right.

transfer_money(fromPos, toPos, balArray)
Input: Positions to transfer money from and to, pointer to balances array
Output: Whether we need to update the player count (did a player lose all their money in this transfer?
Decrement balArray[fromPos]
start_bal ← balArray[toPos]
Increment balArray[toPos]
Print the transfer
if start_bal ← 0 then
  return true

Main

```
Initialize currency and player variables
Print random seed prompt, check input validity, store input in seed if good input
Print player count prompt, check input validity, store input in num_players if good input
Initialize balances[] with length num_players and values equal to the starting amount (macro)
while players_left > 1 do
    Store results of right() and left()
    Determine die_rolls based on current_player's balance
    for die_rolls do
        die_result = result of rand() % 6
        switch die_result
            Pass or transfer money either left, right, or to the pot using transfer_money()
            Print each result of the die
    if balances[current_player] ← 0 then
        Decrement players_left
    if balances[right_player] > 0 then
        current_player ← right_player
    else
        current_player ← next player clockwise with currency (for loop using right())
Print the win message
Free balances[]
End
```