Isaac Flores
Prof. Miller
CSE13S
October 13, 2022

General Idea:

Dreidel.c: get the number of players and number of coins from the arguments. Create an array that has each player's amount of coins. Begin playing the game. Use the Mersenne Twister to calculate a random number and assign a modulo value to a letter on the dreidel. Perform the respective task for what the respective player landed on. If the player has 0 coins and needs to add a coin to the pot they are out of the game. Add a round to the total rounds when the next player is alphabetically before the last player. Repeat until one player remains. Return the index of the winner and the number of rounds played.

Play-dreidel.c: Take the parameters given when the function is called and fill with the default value if left empty. Check if the given parameters fit inside the allowed range and if they do perform the respective task. Plug the parameters into the play_game function. Take the output and use it to display the winner. Display the name of the winner, players in the game, starting amount of coins, the number of rounds, and the seed used to play.

Pseudocode:

Dreidel.c:

```
# function spin_dreidel():
    # get a random number from the Mersenne Twister
    # divide the number by 4 and get the remainder
    # if the remainder is 0, return the letter ‘G’
    # if the remainder is 1, return the letter ‘H’
    # if the remainder is 2, return the letter ‘N’
    # if the remainder is 3, return the letter ‘S’

# function play_game():
    # if n_rounds is equal to 1, this means the -v parameter was given. If so, assign a variable named v_case to 1 so that when a player is eliminated, their name will be printed. If n_rounds isn’t equal to, set v_case to 0. Then reset the n_rounds to 0.
    # for each player in the game
        # assign their respective index in an array to the number of coins each player starts with. This will be the number of coins they currently have. The 0 index will be Aharon, the 1 index will be Batsheva, and so on.
    # for each turn, loop until there is 1 player remaining
        # if the player's coins are not negative, spin the dreidel
```

```
            # spin the dreidel
            # if the letter is ‘N’, do nothing
            # if the letter is ‘G’, add the entire pot to the current player's coins, and then reset the pot to 0
            # if the letter is ‘H’, add half of the pot rounding down to the current player's coins. Then subtract half of the pot rounding down from the pot.
            # if the letter is ‘S’ check if the current player has 0 coins.
                # If they have 0 coins and v_case = 1, assign their amount of coins to -1 meaning they are out. Also, print the player's name that was eliminated on the current rounds in a game with the total amount of players.
                # If they have 0 coins and v_case = 0, assign their amount of coins to -1 meaning they are out.
                # If they have more than 0 coins, subtract 1 coin from the current player’s coins and add a coin to the pot.
    # When the game is over, iterate through all the player's coins. The one player without a negative amount of coins won so assign the winner’s index to a variable.
    # Calculate the number of rounds by dividing the number of turns by the number of players
    # return the winner’s index

# helper function check_winner():
    # iterate through the player’s coins
        # if their coin amount isn’t negative add 1 to the number of players still in
    # if the amount of players still in is equal to 1 return a 0 or else return a 1
```

Play-dreidel.c:

```
# check for arguments given by the caller
    # if a -p number is given, check if the number is less than 2 or greater than 8. If it is, return a non-zero error code. If not, assign the number of players to the given number. If no number is given, set the number of players to 4.
    # if a -c number is given, check if the number is less than 1 or greater than 20. If it is, return a non-zero error code. If not, assign the number of coins to the given number. If no number is given, set the number of coins to 3.
    # if a -s number is given, check if the number is less than or equal to 0 or has more than 10 digits. If it is, return a non-zero error code. If not, assign the seed to the given number. If no number is given, assign the seed to 4.
    # If a -v parameter is given, assign n_rounds to 1 so that the play_game function knows to print when a player loses. If -v is not given, set the variable to 0 so the play_game function knows not to print when a player loses.
# feed the seed into the Mersenne Twister
# play a round of dreidel and print out the winner of the game, the number of players, the number of starting coins, the number of rounds it took to complete, and the seed.
```