

Liam Murray

Lijamurr

CSE13s

Assignment 3

### I Have a Little Dreidel General Idea

The goal of this project is to create a pseudo-random dreidel game simulator, using the Mersenne Twister random number generator. Once the game is created, one will use a bash script to run the program multiple times, store the output in a file, then plot the data and perform statistical analysis on it. By the end of the deadline, a finished product will contain a file of dreidel game logic, a file that accepts command line arguments and creates games of dreidel with them, a bash script used to do analysis on the programs, and a write up describing the results.

The basic implementation is to have one file, `dreidel.c`, containing all of the game logic and called functions that are used when actually playing a game. Then, use a different file, `play-dreidel.c`, to hold the main function and use the implemented functions to actually simulate the games of dreidel being played. This will also require a header file, which will contain all the global variables and called functions.

### Pseudocode

#### Dreidel.c requirements

// Required Functions, their purpose, parameters //

1. `spin-dreidel(void)`: returns g, h, n, or s based on the random number generated by `mt_rand()`. Accepts no parameters.

2. dreidel\_game(): plays a game of dreidel, returns the number of the player who won. Also returns data such as number of rounds via call by reference pointers. Accepts number of players, number of coins, and the pointer to the variable that stores the number of rounds.

Dreidel.c

Initialize an array with the names of 8 players, [Aharon, Batsheva, Chanah, David, Ephraim, Faige, Gamaliel, Hannah]

Spin Dreidel (void)

Declare an array with the characters [g, h, n, s] in that order

Declare num as an int

Run mersenne twister rng machine to get a random number

Store that number in num

Set num = num % 4

This gives a random number between 0-3

Return character array index[num]

This should return a random character g, h, n, or s

Play\_dreidel(players, coins, pointer, print)

Print\_message = print (true or false, based on command line input)

Set num\_players = players

Starting\_coins = coins

Pot = 0

Initialize an array "coin\_array" of size [num\_players] with starting\_coins in each slot

Set counter = num\_players

While num\_players > 1;

player\_id = counter % players //gives the index of the player whose turn it is

If [player\_id]coin\_array < 0; //if the players coins are less than zero (they are out)

Break //skip the turn

Roll = spin dreidel()

If the roll is G: //player takes pot

Set [player\_id]coin\_array += pot

Pot = 0

If the roll is H: // player takes half the pot

Set [player\_id]coin\_array += (pot // 2) //floor division, or set pot to an int

Pot = pot // 2

If the roll is N: //nothing happens, skip turn

break

If the roll is S: //player adds to the pot

If [player\_id]coin\_array == 0, //if they have no coins to give

Set [player\_id] = -1 //make sure player is eliminated

num\_players -= 1 //remove on player from the count

If print\_message == true //if the user selected to print messages

Print elimination message

Set [player\_id]coin\_array -= 1

Pot += 1

Counter += 1

For n in range(players) //check which player won

If coin\_array[n] > 0 //if they have positive number of coins

Winner = n // they won!

Set variable at pointer address = (counter - players)

Return winner

Play-dreidel.c

- Requirements: accepts command line arguments for the number of coins, players, and whether or not a message is to be printed when playing the game.
- Runs a game of dreidel, printing out the winner of the game, the number of players, how many rounds it took, and the seed used for the random number generator.

Play-dreidel pseudocode

Include header files for random number generator and dreidel program

Use getopt() to parse command line arguments

Initialize all arguments to their default values (players, coins, print message, seed)

While loop, exiting when the command line argument list is empty:

    Use switch command:

        If the case is p:

            Set players to p parameter

        If the case is c:

            Set coins to c parameters

        If the case is v:

            Set print\_message to true

        If the case is s:

            Set seed to s parameters

Set rounds = 0

Call play\_dreidel with the arguments(players, coins, \*rounds (pointer), print\_message);

Store play\_dreidel into variable winning\_number

Print (name\_array[winning\_number], players, rounds, seed)

Return 0

Header file for dreidel.c

Include dreidel.c files

Add guard

Declare spin\_dreidel()

Declare play\_dreidel()