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   // either 1 or 0, based on command line input
    Set num_players = players
    Starting_coins = coins
    Pot = 0
    Initialize an array “coin_array” of size 8 with 0 coins in each slot
    Set counter = num_players
    set round counter = 0
```

```
Use a while loop, iterating as many times are there are players
    Set coin_array[current iteration number] = starting coins
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
    If player_id = 0;
        Increment round counter
    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
                Check if value is between 2-8
                If the case is c:
                        Set coins to c parameters
                        Check if value is between 1-20
                If the case is v:
                        Set print_message to true
                If the case is s:
                        Set seed to s parameters
                        Check if value is between 1 - 999999999
        Set rounds = 0
        Set pointer to point at rounds
        Set randint seed = seed parameter
        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()
```