Project 1:
Roulette
Game

Curtis Stephens
CIS 5- Program Logic 42641
April, 2017

INTRODUCTION:

Over the years I have always been attached to the game of roulette. I have spent countless hours (and dollars) trying to figure out systems that would increase my likelihood of walking out of the casino with more money than I walked in with. Video roulette has always been my favorite because not only is it a faster game play, you can get quicker and on more numbers. Though my game does not go as in depth as the games you see in casinos, it is the basic construct of what these games aim to emulate.

For those living under a rock, roulette is a basic guessing game. Traditionally played on a big table where a ball is spun around a wheel until it lands on a number of a certain color grouping, my game ends to achieve this by randomizing numbers and matching results. To increase the difficulty of game play I made it so you can only guess on 10 numbers per spin, but hitting this number will pay out greatly! I did include a system of betting but did not get as far as choosing the betting increment due to lack of time so by default the bets are in $1 increments.

I went in a few different directions some might find odd (like the switch menu at the end for ratios) but this was only to include what was required of me by the professor to meet all the requirements for this project.

SUMMARY:

Project Size: About 240 Lines

Included I have made two basic game types. You can play by color (which essentially could have been even or odd as well) or to play by number. The color play is a basic Boolean style game. A user is asked whether he would like to play either Black or Red. The program then generates the right answer between the two and compares it to your guess. Depending on whether you win or lose, your bank is adjusted while factors like numbers of wins, losses, and games are tallied up for statistical purposes. You can end game play at any time by entering -1

The second (and more complex) game is playing by number. On a true roulette there are no limits to how many plays you can play per spin. As I mentioned before my game is just a little bit different as I limit the number of plays to make the game more difficult. First the user will enter the number of plays they wish to play on this spin (a value of -1 ends game). A random number between 0 and 37 is generated because there are 38 different places to play on a traditional table. The program then loops the number of times you want to play for this spin. The loops will ask the user to play a number between 0-37, then compares to correct answer. If your play is correct your bank will gain $40, wins and just like before wins and games played are tallied. If you lose your bank will decrease a dollar.

The final part of the game is the ratio aspect. As explained, your wins, losses, and number of game plays are tallied up using increment counters throughout the game. The final part will present you with a menu that asks which of the ratios you would like to see!

Room for Improvement:

This project is nowhere near perfect and I realize this. With only a little over a weeks worth of time to work on it, I attempted to get the game flow as seamless as possible. A betting system where the user gets to input how much they would want to bet is something worth adding to the next phase of this project. I also was going to add a odd/even function but the black/red function is virtually the same thing. The adding of the more rare green numbers would offset the code being repetitious in that nature but limits on time kept me from attempting this phase. All in all I am confident in what I produced and feel that I met all the requirements of what you wanted us to demonstrate. Keeping bank totals going through input/output would be interesting too.

Pseudocode:

```
/*
File:   main.cpp
Author: Curtis Stephens
Created in 2017
```

```
Purpose:  Pseudo code for roulette game
*/

//System Libraries
//Input Output library
//Random numbers
//Time to set the Seed
//String Functioning
//Math Library
//Precision library
//Read Write Library
//File stream library
//Format Library

//Namespace std of system libraries

//User Libraries
//Global Constants
//Such as PI, Vc, -> Math/Science values
//as well as conversions from system of units to another
//Percentage Conversion
//Million
//100,000

//Function Prototypes

//Main -> Executable code begins here!!!

   //Set the random number seed

   //Declare Variables and Initialize

      //int for guesses, floats for ratios
      //char and string for choices
      //Set Constants for limits
      //Counters/indicators initialize for wins, loss, and $

  //Intro
    //Instantiate and Open files for header
```

```
  //Retrieve and Display Header
  //Close file
  //Get Player Name
  //Explain betting system
  //Display Menu
    //Choose Game Type

//Play By Color
  //Explain Gain/Loss
  //Ask User to Bet on Black or Red
  //Validate Input
  //Play by Do While Looping
    //Randomize Choice
    //Compare Choice
    //If Win
      //Display Winning Message
      //Add Dollar to Bank
      //Add 1 to Win Tally
      //Add 1 to Games Tally
      //Display Bank Total
    //Else Lose
      //Display Losing Message
      //Subtract Dollar From Bank
      //Add 1 to Loss Tally
      //Add 1 to Games Tally
      //Display Bank Total
      //If Money = 0
        //Display Bankrupt Message
        //Break! End Game
    //Play Another Game.
    //Validate Input
    //Do While Loop Ends Game With -1

//Play By Number
  //Explain Gain/Loss
  //Ask User to How Many Plays on This Spin
  //Validate Input
  //Play by Do While Looping
    //Ask for Number of Plays on This Spin
```

```
    //Validate Input
    //Generate Random Number Between 0-37
    //Initiate For Loop for Number Guess
      //Choose Number
      //Validate Input
      //Compare Choice To Random Number
      //If Win
        //Display Win Message
        //Add $40 to Bank
        //Add 1 to Play Tally
        //Add 1 to Win
        //Display Bank Total
      //Else Lose
        //Display Loss Message
        //Subtract Dollar from Bank
        //Add 1 To Loss Tally
        //Add 1 to Play Tally
        //If Money = 0
          //Display Bankrupt Message
          //Break! End Game
      //Display Winning Number
      //Do While Loop Ends With -1 Entry

//Validate Menu Entries
  //Display Thank You For Coming Message

//Ratios
  //Algebraic and Static Expressions for Answers

//Output Data
  //Set Precision for floats
  //Display Win Total
  //Display Loss Total
  //Ask For Ratio Display
  //Switch Menu for Ratio Display
    //W Displays Win vs Plays
    //L Displays Loss vs Plays
    //O Displays Win Over Loss
    //T Displays All Three
```

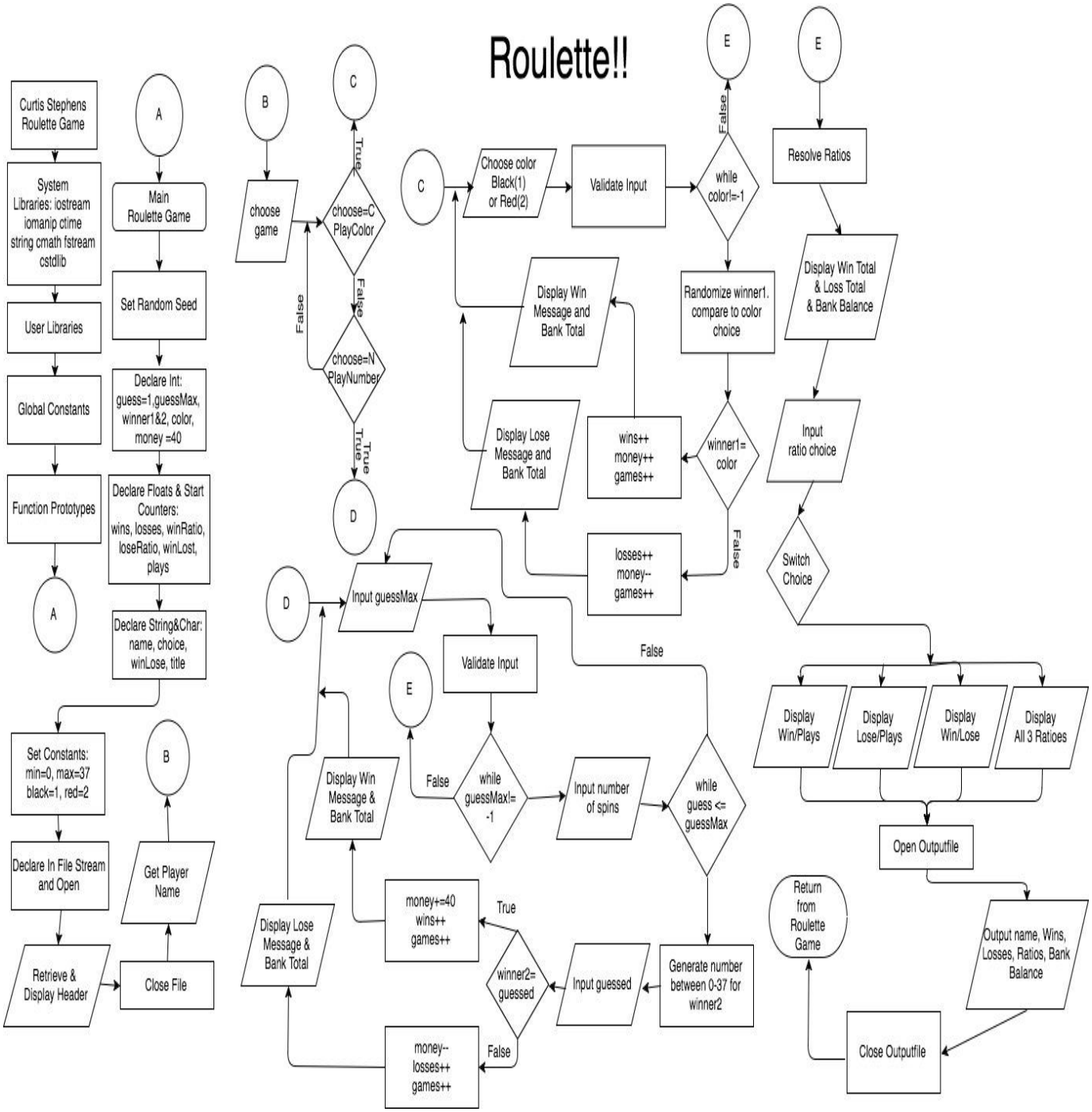
```
  //Display Ending Bank Balance
  //Display Come Again Message

//Write File
  //Open Output File
  //Output Win Total In Output File
  //Output Loss Total In Output File
  //Output Win Ratio In Output File
  //Output Loss Ratio In Output File
  //Output Win/Loss In Output File
  //Output End Balance In Output File
  //Close File

//Exit!!!
```

Cross Reference Check Sheet Project 1 :

Chapter	Section	Topic	Line number
2	2	cout	Used throughout entire project
	3	libraries	iostream, iomanip, cmath, cstdlib, fstream, string, ctime
	4	variables/literals	Lines 31-44
	5	Identifiers	Break, const, string, switch, unsigned, static, int, if, for, while, char, float, etc
	6	Integers	Follows my variables
	7	Characters	Lines 38, 62, 122
	8	Strings	Lines 37, 54, 58, 225
	9	Floats No Doubles	Wins, losses, winRatio, loseRatio, winLost, plays
	10	Bools	Lines 61 and 70
	11	Sizeof *****	Achieved
	12	Variables 7 characters or less	Keep this consistent throughout project
	13	Scope ***** No Global Variables	Achieved
	14	Arithmetic operators	+ - * / %
	15	Comments 20%+	Used throughout, better displayed in pseudo code

	16	Named Constants	Lines 33-36 Min, max, black, red
	17	Programming Style ***** Emulate	Achieved
3	1	cin	Lines 54, 58, 67, 72, 111, 116, 130, 135, 144, 149, 199
	2	Math Expression	Lines 187 -189
	3	Mixing data types ****	Demonstrated through ratio usage
	4	Overflow/Underflow ****	
	5	Type Casting	Line 187
	6	Multiple assignment *****	Line 155
	7	Formatting output	Line 192
	8	Strings	Lines 37, 44, 54, 58, 225
	9	Math Library	Line 13
	10	Hand tracing ******	Achieved
4	1	Relational Operators	Lines 62, 68, 79, 87, 103, 112, 118, 122, 131, 140, 145, 153, 170, 179
	2	if	Lines 62, 79, 152
	4	If-else	Lines 95, 162, 183
	5	Nesting	Most of the project is different nested statements. Primarily demonstrated in the
	6	If-else-if	Line 122
	7	Flags *****	Line 183
	8	Logical operators	Lines 62, 68, 112, 122, 145
	11	Validating user input	Lines 68, 112, 131, 145, 183, 217
	13	Conditional Operator	Lines 68, 112, 131, 145
	14	Switch	Lines 200-218
5	1	Increment/Decrement	Lines 82-84, 98-100, 155-157, 165-167
	2	While	Lines 68-73, 112-117, 131-136, 145-150
	5	Do-while	Lines 74-118, 126-179,
	6	For loop	Lines 140-176
	11	Files input/output both	Lines 47-51, 223-230

Flowchart:


Roulette!!
Curtis Stephens Roulette Game
System Libraries: iostream iomanip ctime string cmath fstream cstdlib
User Libraries
Global Constants
Function Prototypes
A
A
Main Roulette Game
Set Random Seed
Declare Int: guess=1,guessMax, winner1&2, color, money =40
Declare Floats & Start Counters: wins, losses, winRatio, loseRatio, winLost, plays
Declare String&Char: name, choice, winLose, title
B
Get Player Name
Close File
B
Set Constants: min=0, max=37 black=1, red=2
Declare In File Stream and Open
Retrieve & Display Header
C
choose game
choose=C PlayColor
True
False
choose=N PlayNumber
False
True
D
D
Input guessMax
Validate Input
E
while guessMax!= -1
False
Input number of spins
while guess <= guessMax
Generate number between 0-37 for winner2
Input guessed
winner2= guessed
True
money+=40 wins++ games++
False
money-- losses++ games++
Display Win Message & Bank Total
Display Lose Message & Bank Total
C
Choose color Black(1) or Red(2)
Validate Input
while color!=-1
False
E
Randomize winner1, compare to color choice
winner1= color
False
losses++ money-- games++
wins++ money++ games++
Display Win Message and Bank Total
Display Lose Message and Bank Total
E
Resolve Ratios
Display Win Total & Loss Total & Bank Balance
Input ratio choice
Switch Choice
Display Win/Plays
Display Lose/Plays
Display Win/Lose
Display All 3 Ratioes
Open Outputfile
Output name, Wins, Losses, Ratios, Bank Balance
Close Outputfile
Return from Roulette Game

Major Variables:

Type	Variable Name	Description	Use
Int	guess	Starts the guess counter	Used for limiting the for loop
	guessMax	Upper limit for number of plays for this spin	Used as cin object in number game
	guessed	Number you are playing for this play	Compared against winner2
	winner1	Winning answer for color game	Randomized between 1&2
	Winner2	Winning answer for number game	Randomized between 0-37
	color	Similar to guessed, except for the choice in color game	Used as cin var. in color game
	money	This is your bank!	Used as counter to track funds
Const int	min	Minimum number you are allowed to guess	Used as lower limit in number game
	max	Maximum number you are allowed to guess	Used as upper limit in number game
	black	Choice for color game	Emulates playing black
	red	Choice for color game	Emulates playing red
string	name	Players name	Used to track who is playing
	choice	Game choice. (C) for color of (n) for number	Cin used to choose game
	title	Header	Used to pull header from input file
char	winLose	Switch menu choice to display which ratio you want to view	Used as switch menu entry
float	Wins	Track how many plays you have won	Increment counter
	Losses	Track how many plays you have lost	Increment counter
	winRatio	Wins over plays	Algebraic expression
	loseRatio	Losses over plays	Algebraic expression
	winLost	Wins over losses	same
	plays	Track how many games you have played	Increment Counter

Program:

```
/*
         * File:   main.cpp
         * Author: Curtis Stephens
         * Created in 2017
         * Purpose:  Simulate a roulette game
         */

        //System Libraries
        #include <iostream>  //Input - Output Library
        #include <ctime>     //Time for rand
        #include <cstdlib>   //Srand to set the seed
        #include <string>    ///string function
        #include <cmath>     // math functions
        #include <iomanip>   //set precision
        #include <fstream>   //read file
        using namespace std; //Name-space under which system libraries exist

        //User Libraries

        //Global Constants

        //Function Prototypes

        //Execution begins here
        int main(int argc, char** argv) {
            //Set the random number seed
            unsigned seed = time(0);
            srand(seed);

            //Declare & Initialize variables
            int guess = 1;
            int guessMax, guessed, winner1, winner2, color; //random variables
            const int min = 0;  //number min
            const int max = 37; //number max
            const int black = 1; //black
            const int red = 2;  //red
            string name, choice, title;  //strings
            char winLose;  //choose ratio
            unsigned int money = 40;  //starting bank
            float wins = 0;   //set win counter
```

```
 float losses = 0;  //set lose counter
 float winRatio, loseRatio, winLost;  //ratios
 float plays = 0;//set play counter

 //Intro
 ifstream inputFile;  //input file
 inputFile.open("Title.txt");  //source file
 inputFile >> title;  //input info
 cout << title << "!!" << endl;  //display info
 inputFile.close();   //close file
 cout << "_______________" << endl;
 cout << "Player Name: ";
 cin >> name;                    //enter name
 cout << "Each Play Will Cost $1." << endl;  //explain betting system
 cout << "You start with $40." << endl << endl;
 cout << "Play by color (C) or play by number (N)? ";
 cin >> choice;                  //choose game type
 cout << "\n";

//play by color;
if (choice == "C" || choice == "c")   //defining color choice
{
     cout << "Each time your number hits your bank will gain $1!" << endl;
     cout << "Each time you miss your bank will lose $1.\n" << endl;
     cout << "Choose a color! Black (1) or Red (2). To quit enter (-1): ";
     cin >> color;  //choose color
     while (color < -1 || color > 2)   //validate input
     {
         cout << "Not a valid choice! Choose either Black (1) or ";
         cout << "Red (2): ";
         cin >> color;
     }
     do               //loop game
     {
         //generate winning color
         winner1 = (rand() % (red - black + 1)) + black;
         //win!!!!
         if (color == winner1)
         {
             cout << "You win!" << endl;
             money++;  //adjust bank
             wins++;    //add win
             plays++;    //add play
```

```
            cout << "You have $" << money;
            cout << " left. \n" << endl;      //funds left
            cout << endl;
        }
        //lose!!!
        else
        {
            cout << "You lose." << endl;
            money--; //adjust bank
            losses++; //add loss
            plays++;  //add play
            cout << "You have $" << money;
            cout << " left.\n" << endl;      //funds left
            if (money == 0)  //bankrupt
            {
                cout << "You are out of money!! Come Again!" << endl;
                break;    //end game from lack of funds
            }
            cout << endl;
        }
        cout << "Enter another color. Black (1) or Red (2): ";
        cin >> color;  //repeat loop
        while (color < -1 || color > 2)   //validate input
        {
            cout << "Not a valid choice! Choose either Black (1) or ";
            cout << "Red (2): ";
            cin >> color;
        }
    } while (color != -1);  // ends games
}

//play by number
else if (choice == "N" || choice == "n")  //choose number game
{
cout << "Each time your number hits your bank will gain $40!" << endl;
cout << "Each time you miss your bank will lose $1.\n" << endl;
    do   //start number game
    {
        cout << "How many plays do you want for this spin? (up to 10 per ";
        cout << "spin enter -1 when done): ";
        cin >> guessMax;  //how many guesses for this spin
        while (guessMax < -1 || guessMax > 10)
        {
```

```
    cout << "Not a valid number of plays! Reenter ";
    cout << "number of plays (1-10): ";
    cin >> guessMax;
}
//generate winning number
winner2 = (rand() % (max - min + 1)) + min;
//guess and compare
for (guess = 1; guess <= guessMax; guess++)  //guess loop
{
    cout << "What number would you like to play for ";
    cout << "play number " << guess << "? (0-37): ";
    cin >> guessed;  //guess number
    while (guessed < 0 || guessed > 37)  //validate input
    {
        cout << "Not a valid number to play! Please reenter ";
        cout << "The number that you want to play (0-37): ";
        cin >> guessed;
    }
    //win
    if (guessed == winner2)  //winning number
    {
        cout << "This number hit!" << endl; //win
        money += 40;  //add winnings
        plays++;  //add play
        wins++;   //add wins
        cout << "You have $" << money;
        cout << " left. \n" << endl;       //funds left
    }
    //lose
    else    // losing number
    {
        cout << "This number missed." << endl;  //lose
        money--; //subtract losings
        losses++;  //add losses
        plays++; //add plays
        cout << "You have $" << money;
        cout << " left.\n" << endl; //funds left
        if (money == 0)  //bankrupt!!
        {
            cout << "You are out of money!! Come Again!" << endl;
            break;
        }
    }
```

```
        }
        cout << "The Winning Number is ";
        cout << winner2 << "\n\n";        //display winning number
    } while (guessMax != -1);  //to end game
}

// validate entry from menu
else
    cout << "Not a valid game choice. Please come again!";

 //ratios
 winRatio = static_cast<double>(wins) / plays;
 loseRatio = (losses / plays);
 winLost = (wins / losses);

 //output data
 cout << "Your total number of wins is " << wins << endl;  //total wins
 cout << "Your total number of losses is ";
 cout << losses << endl << endl;//total losses
 cout << setprecision(2) << fixed;
 cout << "Would you like to see Win ratio (W), ";
 cout << "Loss ratio (L), Win over Loss ratio (O), ";
 cout << "or All Three (T)?: ";
 cin >> winLose;     //choose ratio
 switch (winLose)   //switch option for ratios
 {
     case 'W':          //win ratio
     case 'w': cout << "Your Win ratio is " << winRatio << endl;
             break;
     case 'L':      //loss ratio
     case 'l': cout << "Your Loss ratio is " << loseRatio << endl;
             break;
     case 'o':     //win loss ratio
     case 'O': cout << "Your Win over Loss ratio is " << winLost << endl;
             break;
     case 'T':
     case 't':    //display all three
         cout << "Your Win ratio is " << winRatio << endl;
         cout << "Your Loss ratio is " << loseRatio << endl;
         cout << "Your Win/Loss ratio is " << winLost << endl;
         break;
     default: cout << "That is an invalid choice. \n"; //validation
 }
```

```
    cout << "Your ending balance is $" << money << endl << endl; //end balance
    cout << "Come Again!";

   //write file
    ofstream outputFile;
    outputFile.open("Output.txt");
    outputFile << "Player: " << name << endl;
    outputFile << "Your number of Wins is" << wins << endl;
    outputFile << "Your number of Losses is" << losses << endl;
    outputFile << "Your Win ratio is " << winRatio << endl;
    outputFile << "Your Loss ratio is " << loseRatio << endl;
    outputFile << "Your Win/Loss ratio is " << winLost << endl;
    outputFile << "Your ending balance is $" << money << endl;
    outputFile.close();

   //Exit stage right!
    return 0;
}
```