EUROPEAN ROULETTE WITH C++ FINAL PROJECT

CSC-5-40107

Scott Parker

spfrood@gmail.com | January 30, 2017

Table of Contents

Introduction

This project will reproduce the core functionality of a European roulette game using C++ and make a playable version on the console. It will include most of the betting options, wheel spinning, and odds found in traditional roulette. Roulette is a gambling game based on spinning a wheel and betting on the results. The wheel is divided into 37 different sectors (or 38 for American roulette). The odds are paid as if there are only 35 different sectors which gives the house its advantage for single-number bets. For other bet types the house maintains an advantage by not including the 0 space in any of the win conditions or odds calculations on the other bets.

Gameplay and Rules

This program will model European roulette. European roulette is very similar to American roulette. The main difference is that European roulette only uses A single '0' rather than '0' and '00' which are included on American roulette tables. This means that even though the house still maintains a slight odds advantage, it is slightly less on European tables.

To play this roulette game there is no playfield. Instead the player is offered bets via a menu. In order to select a bet the player chooses that bet from a menu and then enters the amount of the bet. For bets that are not taken the default bet amount is $0. Entering a bet more than once on the same spin will overwrite the previous bet and not add to it.

Once the player indicates that they are finished entering bets the spin-phase will begin automatically. After the spin the game will calculate the win or loss amounts and keep track of internal record-keeping accordingly. For a win the odds amount will be paid for the value of the bet. On a loss the amount wagered will be deducted from the player's bank. It is at this point that the player has the option to put the game on auto-play and select a number of games.

Development Summary

Project Size: 1200+ Lines of Code

Global Constants: 1

Variables: 35+

Functions: 21

This project was coded over many hours on a very long weekend. The project includes the concepts went over during the first four weeks of the class (covered in the first 5 chapters of the Gaddis book). Specific examples of referenced concepts from various chapters can be found in the program at locations listed in the following table:

Concepts Used

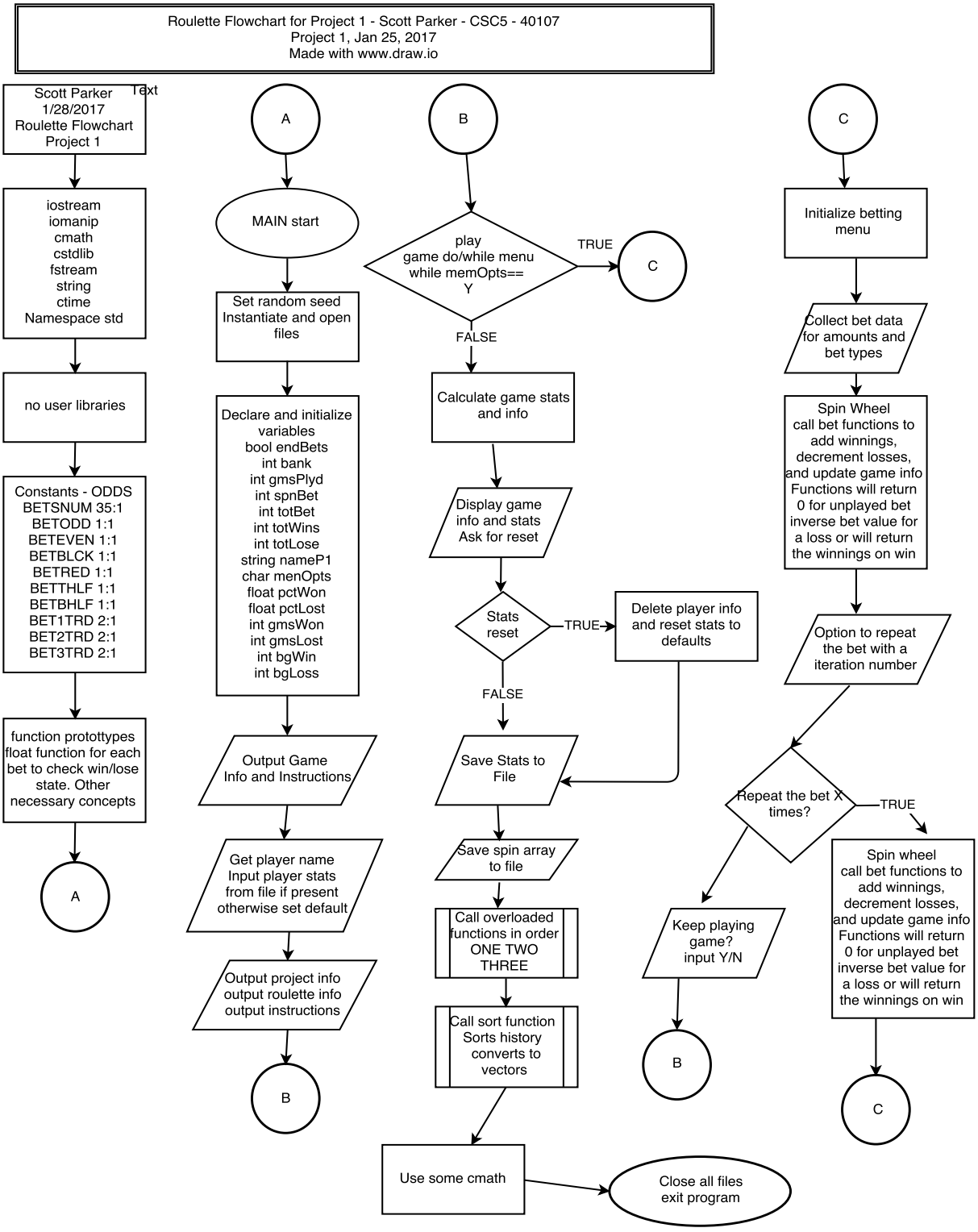
Cross-List

Cross Reference for Project 1

Chapter	Section	Topic	First used in code Line number
2	2	cout	Line 90 first instance
	3	libraries	Lines 10-16 iostream, iomanip, cmath, cstdlib, fstream, string, ctin
	4	variables/literals	Lines 67 to 86
	5	Identifiers	Lines 67-86 amongst others
	6	Integers	Line 77
	7	Characters	Line 83
	8	Strings	Line 82
	9	Floats No Doubles	Line 67
	10	Bools	Line 86
	11	Sizeof *****	
	12	Variables 7 characters or less	Lines 67 to 86
	13	Scope ***** No Global Variables	
	14	Arithmetic operators	Line 128 first instance (+)
	15	Comments 20%+	First comment on line 2
	16	Named Constants	Line 32-42
	17	Programming Style ***** Emulate	
3	1	cin	Line 94 first instance
	2	Math Expression	Line 128 (adding a string), Line 645 multiply
	3	Mixing data types ****	
	4	Overflow/Underflow ****	
	5	Type Casting	Line 573 int to float
	6	Multiple assignment *****	
	7	Formatting output	First used Line 159
	8	Strings	input Line 123, used line 128
	9	Math Library	Line 619, sqrt -- Line 625, pow
	10	Hand tracing ******	
4	1	Relational Operators	Line 95 first instance
	2	if	Line 144 first instance
	4	If-else	Line 523
	5	Nesting	
	6	If-else-if	Line 520
	7	Flags *****	
	8	Logical operators	Line 95 first instance
	11	Validating user input	Line 274
	13	Conditional Operator	Line 645 first used
	14	Switch	Line 233
5	1	Increment/Decrement	Line 302
	2	While	Line 95
	5	Do-while	Line 176
	6	For loop	Line 411
	11	Files input/output both	Line 128 input, line 598 output
	12	No breaks in loops ******	
6	3	Prototypes	Line 41 first instance
	4	Pass by Value	Line 316 first occurance
	8	Returning Values	Line 316 first occurance
	9	Boolean return values	Line 511
	11	Static Local Variable	Line 743 best first example
	12	Default Arguments	Declared Line 42, function line 994
	13	Reference Variables	Function line 909
	14	Overloading functions	Line 742, 782, 801
7	4	Array Initialization	First instance Line 94
	7	Parallel Arrays	First used in function starting on Line 742
	8	Arrays in function arguments	Passed in line 290
	9	2 Dimensional Arrays	line 851
		2 D arrays as function argument	line 851 (passed from 310)
	12	STL Vector	Line 699
8	1	Search Linear/Binary	Line 713
	3	Sorting Bubble/Selection	Line 669
	5	Applied to Vectors	Line 713

Specifications

Flowchart * Flowchart shows pre-code data



Variables * taken from program

```
float betODD=0; //betting amount on ODD numbers
float betEVEN=0; //betting amount on EVEN numbers
float betBLCK=0; //betting amount on BLACK numbers
float betRED=0; //betting amount on RED
float betTHLF=0; //betting amount on TOP HALF of field (1-18)
float betBHLF=0; //betting amount on BOTTOM HALF of field (19-36)
float betFSTT=0; //betting amount on FIRST THRID of field (1-12)
float betSNDT=0; //betting amount on SECOND THIRD of field (13-24)
float betTRDT=0; //betting amount on THIRD THIRD of field (25-36)
//These could technically all be unsigned shorts unless the player plays a
//LOT of roulette
int gmsPlyd=0, spinVal=0, spinWin=0, betWin=0, gmsWon=0, gmsLost=0;
unsigned short gmsPush=0, resetBK=0, resetL=0, resetW=0, resetBT=0, switMen=0;
string playerN;  //player name used in saving stats and loading stats
char menuOpt, yesNo;  //menu option choices that get reused
float bank=100.00, totBet=0.00, totWin=0.00, totLoss=0.00, hiWin=0.00,
      mostBet=0.00, betSpin=0.00, winPcnt=0.00;  //currency values
bool playMor=true; //boolean used in menu choices later on
const int SPINNER=3000;
int utilize = 2000;
int spinRec[SPINNER]={};
int numBets[37][BETCOLS]={};
vector<int> sRecord(utilize, 0); //vector of int
```

Datafiles

The program is capable of keeping track of player profiles and information so long as the players enter a unique name. The game attempts to read the players profile information from a datafile once the player inputs their name. The datafile is also saved with the player information just prior to the program exiting. The information in the datafile is read in sequence so the data must match up to the appropriate field. For example a datafile with the following contents will match up to the listed field and will have the matching output. The datafile ONLY contains the first column the field shows what is requested in the game.

101009	gmsPlyd
672	gmsWon
100317	gmsLost
19	gmsPush
2279	resetBK
2280	resetL
0	resetW
13751	resetBT
-921905	bank
3.01008e+06	totBet
308471	totWin
-444455	totLoss
6454	mostBet
175	hiWin

The above data results in the following statistical output in the game:

You have played 101009 games so far!

You have won money in 672 games so far!

You have lost money in 100317 games so far!

You neither won or loss in 20 games to date.

You have been saved from bankruptcy 2279 times so far.

This is your bank value reset counter

You have been saved from organ harvesting 2280 times so far.

This is your total losses reset counter

You have been saved from the IRS 0 times so far.

This is your total winnings reset counter

You have been saved from yourself 13751 times so far.

This is your total bet amount reset counter

Your bank currently has $-921905.00 in it.

Your total bets are $3010080.00 as of now.

Your winnings come to $308471.00 as of now.

Your total losses are $-444455.00 as of now.

$6454.00 is the most you have ever bet on a game.

Your biggest win was $175.00 on a single game.

Your win ratio is: 0.7%

The program also keeps track of the previous 2000 spins of the wheel in a separate data file. The player's games are unique to their own save datafile but the spins of the wheel are shared across all players unless the files is manually deleted.

Pseudocode

```
/*
 * File:   main.cpp
 * Author: Scott Parker
 * Created on January 28, 2017, 2:30 PM
 * Purpose: Template to be used for all programming projects
 */

//System Libraries
//cin, cout, basic math, etc
//used to set random seed to spin roulette wheel
//used for the random number function to play roulette
//used to input/output player records
```

```
//Used for player name (in conjunction with fstream)
//used to format output to be like currency
//This program really doesn't use CMATH but it's here for
        //the grade. SQRT and POW at end of program. Otherwise unused.
//standard namespaces

//Unsure if the following will work on WINDOWS machines. Test in class before
//submitting project. ** seems to work on windows 10 in class so leaving colors in
//For red text
//for green text
//for cyan text
//reset color to black

//User Libraries

//Global Constants
//Such as PI, Vc, -> Math/Science values
//as well as conversions from one system of measurements to another
//Column row number in 2D array
//Function Prototypes (for returning the amounts of the bets)
//Determine winning numbers
//return value of win betting ODD
//return value of win betting EVEN
//return value of win betting BLACK
//return value of win betting RED
//return value of win betting 1-18
//return value of win betting 19-36
//return value of win betting 1-12
//return value of win betting 13-24
```

```
//return value of win betting 25-36
//Rotate spin history in array
//Print spin history
//Pass by reference to udpate player stats variables
//To enable betting more than 1 number per spin
//function that returns boolean
//Overload function
//Overload function
//Overload function
//Function to demo search/sort and vector
//Binary search function
//sort function
//Executable code begins here! Always begins in Main
//Set random number seed
   //Instantiate files - they are opened and used later
   ifstream in;
   ofstream out;

   //Declare Variables

   float betODD=0; //betting amount on ODD numbers
   float betEVEN=0; //betting amount on EVEN numbers
   float betBLCK=0; //betting amount on BLACK numbers
   float betRED=0; //betting amount on RED
   float betTHLF=0; //betting amount on TOP HALF of field (1-18)
   float betBHLF=0; //betting amount on BOTTOM HALF of field (19-36)
   float betFSTT=0; //betting amount on FIRST THRID of field (1-12)
   float betSNDT=0; //betting amount on SECOND THIRD of field (13-24)
   float betTRDT=0; //betting amount on THIRD THIRD of field (25-36)
```

```
   int betNum=99; //set start value of betNum outside spin range
   //These could technically all be unsigned shorts unless the player plays a
   //LOT of roulette
   int gmsPlyd=0, spinVal=0, spinWin=0, betWin=0, gmsWon=0, gmsLost=0;
   unsigned short gmsPush=0, resetBK=0, resetL=0, resetW=0, resetBT=0, switMen=0;
   string playerN;  //player name used in saving stats and loading stats
   char menuOpt, yesNo;  //menu option choices that get reused
   float bank=100.00, totBet=0.00, totWin=0.00, totLoss=0.00, hiWin=0.00,
         mostBet=0.00, betSpin=0.00, winPcnt=0.00;  //currency values
   bool playMor=true; //boolean used in menu choices later on
   const int SPINNER=3000;
   int utilize = 2000;
   int spinRec[SPINNER]={};
   int numBets[37][BETCOLS]={};
   vector<int> sRecord(utilize, 0); //vector of int
   //Input Values
   //Displaying instructions for program and rules of roulette
   //Giving option to skip instructions and information
//Get player to enter name
//This will load a data file with the player's name
                  //if such exits. It will then populate the fields based
                  //on previous wins, losses, etc, unless they were reset
   //Getting data from player file if available
   //total games played
   //number of games won
   //number of games lost
   //Games with no win or los
   //Counter of bank value resets
   //counter of loss amount resets
```

```
   //counter of win amount resets
   //counter of total money bet resets
   //Amount of money player has
   //Total amount player has bet
   //Total amount player has won
   //Total amount player has lost
   //THe most the player has ever bet on a single spin
   //The most a player has ever won on a single bet
//closing the data file
//displaying the players info if it existed previously
//Getting spins history from file
   //Running loop to import 2000 history values
//Process by mapping inputs to outputs
//Drawing the roulette table in ASCII with colors if possible on machine
//This is the betting options. Displayed for player to choose bet type
   //Switch menu to place the bets of each type
   //betting THIRD THIRD of field (25-36)
   //betting SECOND THIRD of field (13-24)
   //betting FIRST THRID of field (1-12)
   //betting BOTTOM HALF of field (19-36)
   //betting TOP HALF of field (1-18)
   //betting BLACK numbers
   //betting RED
   //betting EVEN numbers
   //betting ODD numbers
   //function to bet individual numbers
//Repeat? Bet again
//always reset variable to unused value after an input is done being used
//Spin the wheel
```

```
//random value from 0 to 36 for roulette wheel spin
//Editing player data and/or incrementing appropriate counters FUNCTION
//reset the win variable for this bet to zero.
//Find win or loss for ODD bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for EVEN bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for BLACK bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for RED bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for TOP HALF (1-18) bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for BOTTOM HALF (19-36) bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for FIRST THIRD (1-12) bet
//Increment with every bet amount
```

```
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for SECOND THIRD (13-24) bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//Find win or loss for SECOND THIRD (13-24) bet
//Increment with every bet amount
//testing and setting highest win
//reset the win variable for this bet to zero.
//settling variables to end this round of bets. Updating data for
//player profile info
//ternary operator test to update win for this spin
//Put the game on auto-play for a while for a user chosen number of games
    //Editing player data. Needs to be edited every time wheel is spun to increment
            // counters and update where needed if out of minimums or maximums
    //random value from 0 to 36 for roulette wheel spin
    //Find win or loss for single number bet
    //Increment with every bet amount
    //testing and setting highest win
    //reset the win variable for this bet to zero.
    //Find win or loss for ODD bet
    //Increment with every bet amount
    //testing and setting highest win
    //reset the win variable for this bet to zero.
    //Find win or loss for EVEN bet
    //Increment with every bet amount
    //testing and setting highest win
    //reset the win variable for this bet to zero.
```

//Find win or loss for BLACK bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//Find win or loss for RED bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//Find win or loss for TOP HALF (1-18) bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//Find win or loss for BOTTOM HALF (19-36) bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//Find win or loss for FIRST THIRD (1-12) bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//Find win or loss for SECOND THIRD (13-24) bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//Find win or loss for SECOND THIRD (13-24) bet

//Increment with every bet amount

//testing and setting highest win

//reset the win variable for this bet to zero.

//settling variables to end this round of bets. Updating data for

```
            //player profile info


    //set highwin if applicable
    //Reset spin to zero
    //Reset bets to defaults
    //Set number selector to out of bounds number
    //Reset bet for ODD to zero
    //Reset bet for EVEN to zero
    //Reset bet for BLACK to zero
    //reset bet for RED to zero
    //Reset bet for TOP HALF (1-18) to zero
    //Reset bet for BOTTOM HALF (19-36) to zero
    //Reset bet for FIRST THIRD (1-12) to zero
    //Reset bet for SECOND THIRD (13-24) to zero
//Outputting player info and statistics
//asking to reset the values in player data file to defaults
//Output values to file - save player profile
//Saving spins history to file
//Calling overloaded function ONE
//Calling overloaded function TWO
//Calling overloaded function THREE – This function uses parallel arrays
//Search and sort demos - search spin history and sort spin history
//Put in a Few things that require <cmath> since this program really doesn't need it
//Exit stage right! - This is the 'return 0' call
   //DONT FORGET TO CLOSE FILES if they were not closed already

//00000001111111111222222222233333333334444444445555555555666666666677777777778
//34567890123456789012345678901234567890123456789012345678901234567890123456789
//**************************  Binary Search  *********************************
//Description: Function that uses vectors and binary search methods
```

```
//
//Inputs: Array, size of array, object of search
//
//Outputs: Returns first instance of search item in array
//*************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//******************************  Sort Array  **********************************
//Description: Sort function to arrange an array from least to greatest
//
//Inputs: array, size of array
//
//Outputs: none
//*************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//**************************  Sort and Search  *********************************
//Description: Function that uses vectors to demonstrate sort and search
//           methods
//
//Inputs: vector of int and the array of spin history (to copy to vector)
//
//Outputs: none
//*************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//****************  Parallel Array in Overload Function  ***********************
//Description: Function that has parallel arrays just as an example
//
//Inputs: none
//
//Outputs: none
//*************************************************************************
```

```
//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//************************* Overload Function **********************************
//Description: two of two functions with the same name to overload
//
//Inputs:
//
//Outputs: none
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//************************* Overload Function **********************************
//Description: One of two functions with the same name to overload
//
//Inputs:
//
//Outputs: none
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//************************* Boolean Function **********************************
//Description: Function that returns a boolean
//
//Inputs: None
//
//Outputs: Returns a boolean
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//********************** Pay for Single Number Bet *******************************
//Description: Determines single-number win and returns winnings value
//
```

```
//Inputs: amount bet, number chosen, winning number (spin results)
//
//Outputs: The amount of winnings for the individual bet or the amount lost
//        if bet did not win (loss is only the amount bet, wins can be a
//        multiple of amount bet depending on the type of bet
//********************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//*************************** Fill Number Bets **********************************
//Description: Takes the bets for Single Number bets
//
//Inputs: Single Number bet array, number of bets made
//
//Outputs: None
//********************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//******************** Set Player Stat Data via Reference ************************
//Description: Pass info to function by reference to change the values in
//          MAIN even though in a void function
//
//Inputs: Player statistics and gambling variables
//
//Outputs: None
//********************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//************************* Print Spin History **********************************
//Description: prints the spin history to the screen
//
//Inputs: spin history array
//
//Outputs: None
```

```
//*****************************************************************************

//00000001111111111222222222233333333334444444444555555555566666666667777777778
//34567890123456789012345678901234567890123456789012345678901234567890123456789
//************************ Rotate Array Values *********************************
//Description: Rotates the array so that each number is moved down a slot
//
//Inputs: spin history array, current spin value
//
//Outputs: None
//*****************************************************************************

//00000001111111111222222222233333333334444444444555555555566666666667777777778
//34567890123456789012345678901234567890123456789012345678901234567890123456789
//********************** Pay for ODD number bet ******************************
//Description: Determines ODD number win and returns winnings value
//
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the odd number bet or the amount lost
//        if bet did not win (loss is only the amount bet, wins can be a
//        multiple of amount bet depending on the type of bet
//*****************************************************************************

//00000001111111111222222222233333333334444444444555555555566666666667777777778
//34567890123456789012345678901234567890123456789012345678901234567890123456789
//********************** Pay for EVEN number bet ******************************
//Description: Determines EVEN number win and returns winnings value
//
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the even number bet or the amount lost
//        if bet did not win (loss is only the amount bet, wins can be a
//        multiple of amount bet depending on the type of bet
//*****************************************************************************
```

```
//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//********************** Pay for BLACK number bet ******************************
//Description: Determines BLACK number win and returns winnings value
//
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the black number bet or the amount lost
//        if bet did not win (loss is only the amount bet, wins can be a
//        multiple of amount bet depending on the type of bet
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//************************ Pay for RED number bet ******************************
//Description: Determines RED number win and returns winnings value
//
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the red number bet or the amount lost
//        if bet did not win (loss is only the amount bet, wins can be a
//        multiple of amount bet depending on the type of bet
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//********************** Pay for TOP HALF number bet ***************************
//Description: Determines Top Half (1-18) number win and returns winnings value
//
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the top half (1-18) number bet or the
//        amount lost if bet did not win (loss is only the amount bet, wins
//        can be a multiple of amount bet depending on the type of bet
//*****************************************************************************
```

```
//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//******************* Pay for BOTTOM HALF number bet **************************
//Description: Determines Bottom Half (19-36) number win and returns winnings
//            value
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the bottom half (19-36) number bet or the
//        amount lost if bet did not win (loss is only the amount bet, wins
//        can be a multiple of amount bet depending on the type of bet
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//******************* Pay for FIRST THIRD number bet **************************
//Description: Determines First Third (1-12) number win and returns winnings
//            value
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the First Third (1-12) number bet or the
//        amount lost if bet did not win (loss is only the amount bet, wins
//        can be a multiple of amount bet depending on the type of bet
//*****************************************************************************

//0000000111111111122222222223333333333444444444455555555556666666666777777777 8
//3456789012345678901234567890123456789012345678901234567890123456789012345678 90
//******************* Pay for SECOND THIRD number bet **************************
//Description: Determines Second Third (13-24) number win and returns winnings
//            value
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the Second Third (13-24) number bet or the
//        amount lost if bet did not win (loss is only the amount bet, wins
//        can be a multiple of amount bet depending on the type of bet
//*****************************************************************************
```

```
//00000001111111111222222222233333333334444444445555555556666666666777777777 8
//345678901234567890123456789012345678901234567890123456789012345678901234567890
//******************* Pay for THIRD THIRD number bet **************************
//Description: Determines Third Third (25-36) number win and returns winnings
//            value
//Inputs: amount bet, winning number (spin results)
//
//Outputs: The amount of winnings for the Third Third (25-36) number bet or the
//        amount lost if bet did not win (loss is only the amount bet, wins
//        can be a multiple of amount bet depending on the type of bet
//*****************************************************************************
```