**PROJECT 2**

*<SLOT MACHINE v2>*

CSC5 - 40107

Cay, Shienne Patricia

Date: 02/09/2017

**Introduction**

Title: Slot Machine v2

Slot Machine is played just as how people play it in casinos. Once a pattern comes out, the player wins the bet. Amount of money won varies by bet amount and the pattern it won with. This is the main idea of the game.

However, I added some features so I could utilize topics in class that were not used version 1. The game starts as soon as it is debugged. The program will display the winning patterns and prompt the user for a decision. Before, it was only to play or to cancel. But now, user can choose to play a mini-game within the program. It’s a number guessing game with a twist. User don’t necessarily guess the number but takes a chance if input number occurs 5 times in printing, user will then earn $10 even with no cash insertion. However, one wouldn’t be able to get the money from mini-game if player does not play the main game. Mini game winnings are added to cash bonus.

Another feature I added is the data feature that is only accessible to casino “staffs”. I set the game up to only accommodate 25 play count and if the number of maximum players is reached, the program will prompt for a Staff Login and reset the data. Even though it can be reset in the program, saved file of user’s game report can still be retrieved as it is in an external file. Inside the staff access are the players game information like the total cash involved, cash loss, cash win and cash bonus. Another thing is, the only way for the program to stop is when the “staff” resets the game.

For the player list, if a user comes back and plays again, user will have the option to look at his/her last game report before it is overwritten by new game play.

**Note:** Just like how it goes in slot machines in any casinos, bet amount is first deducted to the current money before it generates the random numbers. Once the player wins, the bet amount is added to current money, and is multiplied depending on the pattern won.

Example: If user has $50 and bets $25 dollars, and it wins a 2x bet pattern. Though user won $50 dollars, only $25 dollars will be added to his/her account. The money that the user will have on his/her account will be $75.

It is a game won purely by luck.

**Summary**

Project Size: 950+ lines

The Number of Variables: about 60

I have applied every topic required to show for the cross reference list even the input fstream that I was not able to include in the version 1. However, as I lack the time and understanding of the topic, I was not able to include vectors in the version 2. It took me almost a week to complete version 2 without errors as much as possible. I only added one part to the project that we have not learned yet which is outputting the current time and date as the program ends and debugs. I find it necessary to add that for the cashout voucher since I would not be able to create a barcode. The voucher looks a bit like a check then because of it.

I did not change much from the version 1 but only applied newly learned concepts from class.

Other than that, the game is still full of nested do-while loops and switch. I had a hard time formatting the 2D array which is paralleled to a string array because for some reasons, <iomanip> did not work the first time. I also had to adjust a lot of values especially passing values and making a copy because I encountered playing the game with the money from before still intact. It was a bit rigorous but it works now.

**Description**

The main point I programmed in this project is formatting the 2D array paralleled to a string array to make a table of information for staff access. I also had to make changes to make the mini-game possible.

**Pseudo-code**

/\*

\* File: main.cpp

\* Author: Shienne Cay

\* Created on February 8, 2017, 7:49 PM

\* Purpose: PROJECT 2 SLOT MACHINE

\*/

//System Libraries

//User Libraries

//Global Constants

//Such as PI, Vc, -> Math/Science values

//as well as conversions from one system of measurements

//to another

//Percentage Conversion

//Two Dimensional Arrays

//Function Prototypes

//Function to reset all variables for use to 0 after one cancels game

//View game report after a game

//View game report from previous play

//Bonus cash calculation from number of spins reached

//Search a number

//Display voucher

//Fill array with random number elements

//Sorting array for easier win clarification

//Print the array or the random numbers in mini-game

//Name and current money

//Password validation for administrator settings

//Round off Budget

//Insert budget/money

//Display Winning Patterns

//Prompt for game report

//Executable code begins here! Always begins in Main

//Set the random seed using time

//Declare Variables

//Input Values

//Display Winning patterns

//Display times played count

//Money so far added from mini-game

//Prompt user to play game or not

//If Yes, start loop.

//If players count is greater than size for array, prompt administrator for reset

//If user chooses to play the game

//Prompt user for name to be used later for cashout voucher using string datatype

//If user played before, retrieve play count from game report saved on file

//Prompt returning user if he/she wants to see previous game report

/If yes, retrieve whole game report and display

//Upon displaying, reset all retrieved data back to 0

//Add player's name to players list

//If user is a new player, add player's name to list

//Welcome user

//Prompt for cash deposit/insert

//Add the amount inserted to total cash inserted but subtract money from mini-game

//Give cash bonus if Player decided to play

//If inserted amount is below 1000, get 1% of cash and multiply by itself

//If inserted amount is >1000 but <=2000, get 10% of cash

//If over 2000, get 5% of cash and add to the cash of user

//Round off Cash Bonus to two decimal places to get exact value

//Tell the user that the system added cash bonus to account

//Add the cash bonus to overall cash bonus in record

//Call rndOffB function, status function

//Start loop for budget condition. If bool x for budget is true, continue loop

//If budget is greater than 0, start loop.

//Prompt user to Spin or Cancel game

//Any other input will not be accepted

//If user chose to spin and budget is greater than 0

//Prompt user for betting amount

//User input validation of float datatype for bet amount

//If bet is greater than the budget, display invalidity

//If bet is equal to 0, display invalidity

//If bet is less than or equal to budget and budget is greater than 0

//Increment total number of spins

//Generate slot machine numbers

//Determine if user won

//If user won, add corresponding price to budget, increment won and add win to total cash win

//If user lost, subtract bet from budget, increment loss and add loss to total loss

//Determine if gameplay reached

//If reached add cash bonus, increment gameplay, add bonus to total cash bonus

//Call rndOffB function, status function

//Loop for spin or cancel if budget>0, bet is less than 0 and budget is > 0.

//If user chose cancel at the start

//If total spin count is 0 and user chose not to play game after money inserted

//Deduct cash bonus from inserted amount

//Deduct cash bonus added for total record

//Exit loop for spin

//If spin count is >1, exit loop

//If user input is not 'S' or 'C', display invalidity, ask again

//Case statement for choice in variable Begin

//Only accepts 'Y' and 'N' as input

//If user chooses to play the game

//Prompt user for name to be used later for cashout using string datatype

//Start loop

//Prompt user to insert cash amount for the game

//Validate user input for float datatype

//Loop if cash amount inserted is less than 1 or above the limit $20000

//Continue loop while budget/cash inserted <1 or >20000

//Add the amount inserted to total cash inserted

//Give cash bonus if Player decided to play

//If inserted amount is below 1000, get 1% of cash and multiply by itself

//If inserted amount is >1000 but <=2000, get 10% of cash

//If over 2000, get 5% of cash and add to the cash of user

//Round off Cash Bonus to two decimal places to get exact value

//Tell the user that the system added cash bonus to account

//Add the cash bonus to overall cash bonus in record

//Round off Cash amount of user to get exact cash value

//Display current progress

//Name and current amount of money on system

//Start loop for budget condition. If bool x for budget is true, continue loop.

//If budget is greater than 0, start loop.

//Prompt user to Spin or Cancel game

//If user chose to spin and budget is greater than 0

//Prompt user for betting amount

//Validate user input for float datatype

//If bet is greater than the budget, display invalidity

//If bet is equal to 0, display invalidity

//If bet is less than or equal to budget and budget is greater than 0

//Increment total number of spins

//If number of spin cash bonus is reached, add cash bonus

//If no conditions are met, cash amount remains the same

//If spin cash bonus is triggered, calculate amount to be added

//based on same conditions for cash addition above

//Let user know that he/she won cash bonus for reaching specific number

//of spins that the game requires

//Add game bonus spins by itself or multiply required by 2 for next bonus

//Add the cash bonus earned to total cash bonus for record

//Start rolling and generating numbers for the slot machine

//Value range from 4 to 8 //Value range from 4 to 8 //Value range from 4 to 8

//Value range from 4 to 8 //Value range from 4 to 8 //Value range from 4 to 8

//Value range from 4 to 8 //Value range from 4 to 8 //Value range from 4 to 8

//Display the generated numbers

//Bet is deducted first before game generate wins or loss

//If all number patterns come out, display win

//If all 777s come out, user wins 100x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If all same numbers come out, user wins 50x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If square pattern comes out, display win

//if 777s come out in square, user wins 15x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If normal numbers come out in square, user wins 10x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If cross pattern comes out, display win

//If cross pattern 777s come out, user wins 10x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If cross pattern normal numbers come out, user wins 5x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If X Pattern comes out, display win

//If X pattern 777s come out, user wins 10x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If X pattern normal numbers come out, user wins 5x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If straight horizontal pattern in middle comes out, display win

//If straight 777s in middle come out, user wins 10x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If normal straight numbers come out, user wins 2x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If straight horizontal pattern comes out, display win

//If straight 777s come out on top, user wins 3x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If normal number come out on top, user wins 2x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If straight horizontal pattern comes out, display win

//If straight 777s come out below, user wins 3x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If normal number come out below, user wins 2x

//Increment number of wins

//Add winning amount to total amount of wins for record

//If diagonal pattern comes out, display win

//If 777s diagonal pattern come out, user wins 3x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If normal number diagonal pattern come out, user wins 2x bet

//Increment number of wins

//Add winning amount to total amount of wins for record

//If no winning pattern comes out, display bet deduction

//Increment number of loss

//Add loss amount to total amount of wins for record

//Round off budget to get exact value

//Display Name and current money

//Loop for spin or cancel if budget>0, bet is less than 0 and budget is > 0.

//If user chose 'C' to cancel game

//If total spin count is 0 and user chose not to play game after money inserted

//Deduct cash bonus from inserted amount

//Deduct cash bonus added for total record

//Exit loop for spin

//If spin count is >1, exit loop

//If user input is not 'S' or 'C', display invalidity, ask again

//Continue loop til budget is 0 or choice is not to spin

//and boolean for boolean for budget remains true

//If budget is 0 or turns 0, or if cash is used up

//Start loop for cash insert

//Prompt user if players wants to insert more cash to continue playing

//If yes, ask how much

//Start loop for add amount

//Validate user input for float datatype

//If add amount is is greater than 0, add to budget or current money

//If add amount is <= 0, diplay invalidity

//Amount inserted is added into total cash inserted record

//Round off budget for exact value

//Display name and current money

//Continue loop if added inserted cash input is less than 0

//Boolean for add game turns false.. exit loop for add cash

//If choice is 'n' or 'N'

//Boolean for add game turns false.. exit loop for add cash

//Boolean for budget turns false, exit game play loop

//If user input is not 'Y' or 'N', display invalidity

//Continue loop if bool z for add cash remains true

//If no conditions are met exit game play loop

//Continue loop if bool x for budget remains true or budget is greater than 0, continue loop

//Exit game play

//If begin is 'n' or 'N', exit game prompt

//Game play exit

//If begin is not 'N' or 'Y', display invalidity

//Start loop exit condition

//Calculate total money involved

//Loop voucher one time

//If chosen to exit game and there's money left on account

//Start printing vouchers

//Ticket number for voucher generator

//Number generator for validation code of voucher

//Number generator for machine number

//Display voucher in a specific set of format

//Set iomanip/header format

//Prompt user if wants to see game report

//If yes, display report

//Calculate percentage of loss and win based on total number of spins

//Display percentage result

//If user does not want to see game report, save report to a .dat file

//If user input is not either 'Y' or 'N', display invalidity

//If no money left on account

//Prompt user if he/she wants to see game report

//If yes, display report

//If user does not want to see game report, save report to a .dat file

//Once condition reached, end game.

//If user decides not to play game, end game

**Program**

/\*

\* File: main.cpp

\* Author: Shienne Cay

\* Created on January 27, 2017, 3:57 PM

\* Purpose: PROJECT 1 SLOT MACHINE

\* Problem:

\*/

#include <iostream>

#include <string>

#include <cstdlib>

#include <ctime>

#include <iomanip>

#include <cmath>

#include <fstream>

using namespace std;

const short PERCENT = 100; //Percentage Conversion

int main(int argc, char\*\* argv) {

srand(static\_cast<unsigned int>(time(0)));

time\_t now = time(0);

char\* dt = ctime(&now);

string name,

dummy;

const float bonCsh1 = 0.05f,

bonCsh2 = 0.10f,

bonCsh3 = 0.15f,

bonCsh4 = 0.25f,

bonCshP = 0.01f,

minVal = 0;

float budget,

add,

bet,

loss = 0,

win = 0,

bonWin = 0,

cshIns = 0,

totCash,

winPer,

lossPer;

int numWin = 0,

numLose = 0,

play = 0,

g8mBon = 10,

rndOff = 100;

char begin,

load,

choice,

report;

unsigned short rn1, rn2, rn3,

rn4, rn5, rn6,

rn7, rn8, rn9;

bool x = true;

bool y = true;

bool z = true;

bool rep = true;

//Input Values

cout<<"\t CSC5 CASINO RIVERSIDE"<<endl<<endl;

cout<<"\tSLOT MACHINE 2017 (C++) Version"<<endl<<endl;

cout<<" Winning Patterns: JACKPOT: "<<endl<<endl;

cout<<" X X X - - - - - - X - X - - - X X X "<<endl

cout<<" - - - X X X - - - - X - 7 7 7 X - X "<<endl;

cout<<" - - - - - - X X X X - X - - - X X X "<<endl<<endl;

cout<<" X - - - - x - X - X X X "<<endl;

cout<<" - X - - X - X X X X X X "<<endl;

cout<<" - - X x - - - X - X X X "<<endl<<endl;

do {

cout<<"Would you like to play the game? "<<endl;

cout<<" Y - YES N - NO"<<endl<<endl;

cout<<"CHOICE: ";

cin>>begin;

getline(cin, dummy);

switch (begin) {

case 'y':

case 'Y': {

cout<<endl;

cout<<"Enter Player Name: ";

voucher

getline(cin, name);

do {

cout<<"Insert Cash: $";

cin>>budget;

getline(cin, dummy);

while (cin.fail()) {

cout<<"\nYou must only enter a number!"<<endl<<endl;

cout<<"Insert Cash: $";

cin.clear();

getline(cin, dummy);

cin>>budget;

}

if (budget < 1) cout<<"\nCash amount inserted cannot be less than 1!\n"<<endl;

else if (budget > 20000) {

cout<<"\nCash inserted is above the limit!"<<endl;

cout<<"Cash amount cannot go above $20000\n"<<endl;

}

} while ((budget < 1) || budget > 20000);

cshIns += budget;

float cshBon = (budget<=1000)?pow((budget\*bonCshP), 2):

(budget>1000&&budget<=2000)?budget\*bonCsh2: (budget>=2000)?budget\*bonCsh1:minVal;

if (budget >= 0 && budget <= 20000) {

int exCshBn = cshBon\*rndOff+0.5;

cshBon = exCshBn/100.0f;

budget += cshBon;

cout<<"\nThank you for choosing to play the game!"<<endl;

cout<<"We added $"<<cshBon<<" to your capital as cash bonus."<<endl<<endl;

cout<<"Good luck ^\_^"<<endl;

bonWin+=cshBon; }

int exBud = budget\*rndOff+0.5;

budget = exBud/100.0f;

cout<<endl

<<"----------------------------------------"<<endl;

cout<<"Player: "<<name<<endl;

cout<<"Current Money: $"<<budget<<endl;

cout<<"----------------------------------------"<<endl<<endl;

do {

if (budget > minVal) {

do {

cout<<"\tPLAY"<<endl<<endl;

cout<<"S - SPIN C - CANCEL"<<endl<<endl;

cout<<"CHOICE: ";

cin>>choice;

if ((choice == 'S' || choice == 's') && budget>minVal) {

do {

cout<<endl<<"BET: $";

cin>>bet;

while (cin.fail()) {

cout<<"\nYou must only enter a number!"<<endl<<endl;

cout<<"Bet: $";

cin.clear();

getline(cin, dummy);

cin>>bet;

}

if (bet>budget) {

cout<<"\nYour bet cannot be higher than your capital!"<<endl;

cout<<"Your remaining cash is $"<<budget<<endl<<endl;

}

else if (bet<=minVal) {

cout<<"\nBet must be greater than 0!"<<endl;

cout<<"Your remaining cash is $"<<budget<<endl<<endl;

}

else if (bet<=budget && bet>minVal) {

play++;

budget = (play==g8mBon && g8mBon<=50)?budget+=(loss\*bonCsh1):

(play==g8mBon && g8mBon<=100)?budget+=(loss\*bonCsh2):

(play==g8mBon && g8mBon<=300)?budget+=(loss\*bonCsh3):

(play==g8mBon && g8mBon>=301)?budget+=(loss\*bonCsh4): budget;

if (play==g8mBon) {

float amount = (g8mBon<=50)?(loss\*bonCsh1): (g8mBon<=100)?(loss\*bonCsh2):

(g8mBon<=300)?(loss\*bonCsh3):

(g8mBon>=301)?(loss\*bonCsh4):'\n';

cout<<"\nCongratulations!\n"

<<"You earned game bonus for reaching "<<g8mBon<<" number of

spins!"<<endl;

cout<<"$"<<amount<<" is added to your cash amount!"<<endl;

g8mBon+=g8mBon;

bonWin+=(g8mBon<=50)?(loss\*bonCsh1):

(g8mBon<=100)?(loss\*bonCsh2):

(g8mBon<=300)?(loss\*bonCsh3):

(g8mBon>=301)?(loss\*bonCsh4):minVal;

}

rn1 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn2 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn3 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn4 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn5 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn6 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn7 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn8 = (rand()%(8-4+1))+4; //Value range from 4 to 8

rn9 = (rand()%(8-4+1))+4; //Value range from 4 to 8

cout<<endl;

cout<<"\t"<<rn1<<" "<<rn2<<" "<<rn3<<endl;

cout<<"\t"<<rn4<<" "<<rn5<<" "<<rn6<<endl;

cout<<"\t"<<rn7<<" "<<rn8<<" "<<rn9<<endl<<endl;

if (rn1==rn2 && rn2==rn3 && rn3==rn4 && rn4 == rn5

&& rn5==rn6 && rn6 == rn7 && rn7 == rn8 && rn8 == rn9) {

if (rn1 == 7) {

budget += (bet\*99);

cout<<"All 777s earn 100x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*100)<<endl;

numWin++;

win+=(bet\*99);

}

else {

budget += (bet\*49);

cout<<"All same number earns 50x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*50)<<endl;

numWin++;

win+=(bet\*49);

}

}

else if (rn1==rn2 && rn2==rn3 && rn3==rn4 && rn4 == rn6

&& rn7==rn8 && rn8 == rn9) {

if (rn1 == 7) {

budget += (bet\*14);

cout<<"Square Pattern 777s earn 15x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*15)<<endl;

numWin++;

win+=(bet\*14);

}

else {

budget += (bet\*9);

cout<<"Square Pattern number earns 10x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*10)<<endl;

numWin++;

win+=(bet\*9);

}

}

else if (rn2==rn4 && rn4==rn5 && rn5==rn6 && rn6==rn8) {

if (rn2 == 7) {

budget += (bet\*9);

cout<<"Cross Pattern 777s earn 10x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*10)<<endl;

numWin++;

win+=(bet\*9);

}

else {

budget += (bet\*4);

cout<<"Cross Pattern number earns 5x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*5)<<endl;

numWin++;

win+=(bet\*4);

}

}

else if (rn1==rn5 && rn5==rn9 && rn9==rn3 && rn3 == rn7) {

if (rn1 == 7) {

budget += (bet\*9);

cout<<"X Pattern 777s earn 10x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*10)<<endl;

numWin++;

win+=(bet\*9);

}

else {

budget += (bet\*4);

cout<<"X Pattern number earns 5x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*5)<<endl;

numWin++;

win+=(bet\*4);

}

}

else if (rn4==rn5 && rn5==rn6) {

if (rn4 == 7) {

budget += (bet\*9);

cout<<"Straight 777s in the middle earn 10x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*10)<<endl;

numWin++;

win+=(bet\*9);

}

else {

budget += bet;

cout<<"Straight number earns 2x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*2)<<endl;

numWin++;

win+=bet;

}

}

else if (rn1==rn2 && rn2==rn3) {

if (rn1 == 7) {

budget += (bet\*2);

cout<<"Straight 777s earn 3x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*3)<<endl;

numWin++;

win+=(bet\*2);

}

else {

budget += bet;

cout<<"Straight number earns 2x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*2)<<endl;

numWin++;

win+=bet;

}

}

else if (rn7==rn8 && rn8==rn9) {

if (rn7 == 7) {

budget += (bet\*2);

cout<<"Straight 777s earn 3x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*3)<<endl;

numWin++;

win+=(bet\*2); }

else {

budget += bet;

cout<<"Straight number earns 2x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*2)<<endl;

numWin++;

win+=bet;

}

}

else if (rn1==rn5 && rn5==rn9) {

if (rn1 == 7) {

budget += (bet\*2);

cout<<"Diagonal 777s earn 3x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*3)<<endl;

numWin++;

win+=(bet\*2);

}

else {

budget += bet;

cout<<"Diagonal number earns 2x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*2)<<endl;

numWin++;

win+=bet;

}

}

else if (rn7==rn5 && rn5==rn3) {

if (rn7 == 7) {

budget += (bet\*2);

cout<<"Diagonal 777s earn 3x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*3)<<endl;

numWin++;

win+=(bet\*2);

}

else {

budget += bet;

cout<<"Diagonal number earns 2x bet!"<<endl;

cout<<"Congratulations for winning $"<<(bet\*2)<<endl;

numWin++;

win+=bet; }

}

else {

budget -= bet;

cout<<"Sorry, you lost $"<<bet<<endl;

numLose++;

loss+=bet;

}

exBud = budget\*rndOff+0.5;

budget = exBud/100.0f;

cout<<"\n"

<<"-----------------------------------"<<endl;

cout<<"Player: "<<name<<endl;

cout<<"Current Money: $"<<budget<<endl;

cout<<"-----------------------------------"<<endl<<endl;

}

else {

cout<<"\nYou can only enter numbers!\n"<<endl;

}

} while (budget>minVal && bet<minVal && budget!=minVal);

}

else if (choice == 'C' || choice == 'c') {

if (play == minVal) {

budget -= cshBon;

bonWin -=cshBon;

cout<<"\nNOTICE: Cash bonus is deducted because you have not played yet!"<<endl;

x = false;

}

else x=false;

}

else { cout<<"\nWarning: Enter only either 'S' or 'C' \n"<<endl; }

} while ((choice == 'S' || choice == 's') && x && budget != minVal);

}

else if (budget==minVal) {

do {

cout<<"You have used up all your funds. Would you like to insert again? "<<endl;

cout<<" Y - Yes N - No "<<endl<<endl;

cout<<"CHOICE: ";

cin>>load;

switch (load) {

case 'y':

case 'Y': {

do {

cout<<"\nInsert Cash: $";

cin>>add;

while (cin.fail()) {

cout<<"\nYou must only enter a number!"<<endl<<endl;

cout<<"Insert Cash: $";

cin.clear();

getline(cin, dummy);

cin>>add;

}

if (add > minVal) budget += add;

else cout<<"Value must be greater than 0!"<<endl;

cshIns+=add;

exBud = budget\*rndOff+0.5;

budget = exBud/100.0f;

cout<<"\n"

<<"-----------------------------------"<<endl;

cout<<"Player: "<<name<<endl;

cout<<"Current Money: $"<<budget<<endl;

cout<<"-----------------------------------"<<endl<<endl;

} while (add <= minVal);

z = false;

} break;

case 'n':

case 'N': z = false;

x = false;

break;

default: cout<<"\nYou can only enter 'Y' or 'N'\n"<<endl;

break;

}

} while (z);

}

else x=false;

} while (x);

y = false;

} break;

case 'n':

case 'N': {

cout<<"\nGoodbye!"<<endl;

y = false;

} break;

default: {

cout<<"\nWarning: You can only enter 'Y' or 'N'!\n"<<endl;

} break;

}

} while (y);

cout<<fixed<<setprecision(2)<<showpoint;

if (x == false) {

totCash = cshIns + bonWin + win;

for (int p = 1; p <= 1; p++) {

if (budget>minVal) {

cout<<"\nCashout voucher printing..."<<endl<<endl;

unsigned short ticket = (rand()%(9999-1000+1))+1000;

unsigned short v1 = (rand()%(99-10+1))+10;

unsigned short v2 = (rand()%(9999-1000+1))+1000;

unsigned short v3 = (rand()%(9999-1000+1))+1000;

unsigned short v4 = (rand()%(9999-1000+1))+1000;

unsigned short v5 = (rand()%(9999-1000+1))+1000;

unsigned int mNum = (rand()%(4294967294-1000000000+1))+1000000000;

cout<<"\t------------------------------------------------------"<<endl;

cout<<"\t CSC5 CASINO RIVERSIDE, CA"<<setw(27)<<ticket<<endl<<endl;

cout<<"\t\t\t CASHOUT VOUCHER"<<endl<<endl;

cout<<"\t\t\tFOR "<<name<<endl<<endl;

cout<<"\t\t Validation"<<setw(5)<<v1<<"-"<<v2<<"-"<<v3<<"-"<<v4<<"-"<<v5<<endl;

cout<<"\t\t "<<dt<<endl<<fixed<<setprecision(2)<<showpoint;

cout<<"\t\t\t "<<setw(5)<<"$"<<budget<<endl<<endl;

cout<<"\t Ticket void after 30 days"<<setw(17)<<"Machine #"<<mNum<<endl;

cout<<"\t------------------------------------------------------"<<endl<<endl;

cout<<"\tNOTE: Bring your ID with you to cash it out to verify \n"

<<"\t name in the voucher."<<endl;

do {

cout<<"\nWould you like to see your game report?"<<endl

<<" Y - YES N - NO"<<endl<<endl;

cout<<"CHOICE: ";

cin>>report;

if (report == 'Y' || report == 'y') { //

cout<<"\n\tSLOT MACHINE GAME REPORT"<<endl<<endl;

cout<<"Number of games played: "<<play<<endl;

cout<<"Number of wins: "<<numWin<<endl;

cout<<"Number of losses: "<<numLose<<endl<<endl;

cout<<"Total amount of cash inserted: $"<<cshIns<<endl;

cout<<"Total amount of cash won: $"<<win<<endl;

cout<<"Total amount of cash bonus: $"<<bonWin<<endl<<endl;

cout<<"Total amount of cash involved: $"<<totCash<<endl;

cout<<"Total amount of cash loss: $"<<loss<<endl<<endl;

winPer = (static\_cast<float>(numWin)/play)\*PERCENT;

lossPer = (static\_cast<float>(numLose)/play)\*PERCENT;

cout<<"Win Percentage: "<<winPer<<"%"<<endl;

cout<<"Loss Percentage: "<<lossPer<<"%"<<endl;

cout<<"\nThank you for playing the game! See you again next time!"<<endl<<endl;

rep = false;

}

else if (report == 'N' || report == 'n') {

ofstream out;

out.open(name+".dat");

out<<"\n\tSLOT MACHINE GAME REPORT"<<endl<<endl;

out<<"Number of games played: "<<play<<endl;

out<<"Number of wins: "<<numWin<<endl;

out<<"Number of losses: "<<numLose<<endl<<endl;

out<<"Total amount of cash inserted: $"<<cshIns<<endl;

out<<"Total amount of cash won: $"<<win<<endl;

out<<"Total amount of cash bonus: $"<<bonWin<<endl<<endl;

out<<"Total amount of cash involved: $"<<totCash<<endl;

out<<"Total amount of cash loss: $"<<loss<<endl<<endl;

winPer = (static\_cast<float>(numWin)/play)\*PERCENT;

lossPer = (static\_cast<float>(numLose)/play)\*PERCENT;

out<<"Win Percentage: "<<winPer<<"%"<<endl;

out<<"Loss Percentage: "<<lossPer<<"%"<<endl;

cout<<"\nGame report is saved to a file."<<endl;

cout<<"\nThank you for playing! See you again!\n"<<endl;

out.close();

rep = false;

}

else cout<<"\nWarning: You can only enter 'Y' or 'N'!"<<endl;

} while (rep);

}

else if (budget <=0) {

cout<<"\nSorry, you didn't win any amount this time."<<endl;

do {

cout<<"\nWould you like to see your game report?"<<endl

<<" Y - YES N - NO"<<endl<<endl;

cout<<"CHOICE: ";

cin>>report;

if (report == 'Y' || report == 'y') {

cout<<"\n\tSLOT MACHINE GAME REPORT"<<endl<<endl;

cout<<"Number of games played: "<<play<<endl;

cout<<"Number of wins: "<<numWin<<endl;

cout<<"Number of losses: "<<numLose<<endl<<endl;

cout<<"Total amount of cash inserted: $"<<cshIns<<endl;

cout<<"Total amount of cash won: $"<<win<<endl;

cout<<"Total amount of cash bonus: $"<<bonWin<<endl<<endl;

cout<<"Total amount of cash involved: $"<<totCash<<endl;

cout<<"Total amount of cash loss: $"<<loss<<endl<<endl;

winPer = (static\_cast<float>(numWin)/play)\*PERCENT;

lossPer = (static\_cast<float>(numLose)/play)\*PERCENT;

cout<<"Win Percentage: "<<winPer<<"%"<<endl;

cout<<"Loss Percentage: "<<lossPer<<"%"<<endl;

cout<<"\nThank you for playing the game! See you again next time!"<<endl<<endl;

rep = false;

}

else if (report == 'N' || report == 'n') {

ofstream out;

out.open(name+".dat");

out<<"\n\tSLOT MACHINE GAME REPORT"<<endl<<endl;

out<<"Number of games played: "<<play<<endl;

out<<"Number of wins: "<<numWin<<endl;

out<<"Number of losses: "<<numLose<<endl<<endl;

out<<"Total amount of cash inserted: $"<<cshIns<<endl;

out<<"Total amount of cash won: $"<<win<<endl;

out<<"Total amount of cash bonus: $"<<bonWin<<endl<<endl;

out<<"Total amount of cash involved: $"<<totCash<<endl;

out<<"Total amount of cash loss: $"<<loss<<endl<<endl;

winPer = (static\_cast<float>(numWin)/play)\*PERCENT;

lossPer = (static\_cast<float>(numLose)/play)\*PERCENT;

out<<"Win Percentage: "<<winPer<<"%"<<endl;

out<<"Loss Percentage: "<<lossPer<<"%"<<endl;

cout<<"\nGame report is saved to a file."<<endl;

cout<<"\nThank you for playing! See you again!\n"<<endl;

out.close();

rep = false;

}

else cout<<"\nWarning: You can only enter 'Y' or 'N'!"<<endl;

} while (rep);

}

}

}

else cout<<"\nPlay the game next time!"<<endl

return 0;

}

**Cross Reference for Projects 1**

|  |  |  |  |
| --- | --- | --- | --- |
| Chapter | Section | Topic | Where in code – Line Number |
| 2 | 2 | Cout | 77-85, 134-137, 143-147, 476, 486-496, 505-513, 518-521, 565-573, 578-581, 615 |
|  | 3 | Libraries | iostream, iomanip, fstream, cstdlib, cmath, string, ctime |
|  | 4 | Variables/Literals | 40-73, 125, 130, 140, 187 |
|  | 5 | Identifiers | 40-73, 125, 130, 140, 187 |
|  | 6 | Integers | 58-62 |
|  | 7 | Characters | 63-66 |
|  | 8 | Strings | 40-41 |
|  | 9 | Floats no doubles | 48-57, 125,30, 140, 187 |
|  | 10 | Bools | 70-73 |
|  | 11 | Sizeof \*\*\*\*\* | n/a |
|  | 12 | Variables 7 characters or less | checked |
|  | 13 | Scope \*\*\*\*\* no global variables | n/a |
|  | 14 | Arithmetic Operators | checked – 472… |
|  | 15 | Comments 20% + | checked |
|  | 16 | Named Constants | 26, 42-47 |
|  | 17 | Programming Style \*\*\*\*\* | n/a |
|  |  |  |  |
| 3 | 1 | Cin | 92, 100, 104, 155, 403, 417, 502, 562 |
|  | 2 | Math Expression | checked – 125-128… |
|  | 3 | Mixing data types \*\*\*\*\* | n/a |
|  | 4 | Overflow/Underflow \*\*\*\*\* | n/a |
|  | 5 | Type Casting | 515-516, 539-540, 575-576, 599-600 |
|  | 6 | Multiple Assignment \*\*\*\*\* | n/a |
|  | 7 | Formatting Output | 468 |
|  | 8 | Strings | 40-41 |
|  | 9 | Math Library | 125 |
|  | 10 | Hand Tracing \*\*\*\*\* | n/a |
|  |  |  |  |
| 4 | 1 | Relational Operators | checked – 125-128… |
|  | 2 | If | 420-421, 386-392… |
|  | 4 | If-else | 420-421, 386-392… |
|  | 5 | Nesting | 150-396 |
|  | 6 | If-else-if | 150, 398, 475, 555… |
|  | 7 | Flags \*\*\*\*\* | n/a |
|  | 8 | Logical Operators | checked – 104, 525… |
|  | 11 | Validating User Input | 107-113, 162-168, 412-418 |
|  | 13 | Conditional Operator | 125-128… |
|  | 14 | Switch | 95, 405 |
|  |  |  |  |
| 5 | 1 | Increment/Decrement | 179, 223… play++, numWin++, numLose++ |
|  | 2 | While | 107, 162, 412 |
|  | 5 | Do-while | 88, 102, 199, 158, 399, 408, 498, 558… |
|  | 6 | For loop | 474 |
|  | 11 | Files input/output | Output – 526-547, 586-607 |
|  | 12 | No breaks in loops \*\*\*\*\* | checked |
|  |  |  |  |