**PROJECT 1**

*<MONKEY GAME>*

CSC17A - 42636

Cay, Shienne Patricia

Date: 04/17/2017

**Introduction**

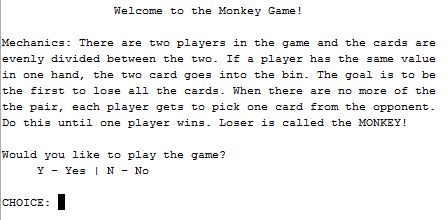
Title: Monkey Game

Monkey Game is played using a deck of cards. It can have players up to 4 people but in this case, only 2 players are accommodated. The one with the highest point after the end of the game wins, and the loser is called the Monkey of the game.

The game goes when the machine divides the shuffled cards to two players so each player holds 26 cards. The player will then discard all the ‘pairs’ in their hand to gather points. Pairs are the same face value of card disregarding its suit. So, if player one has a ‘4S’ and a ‘4C’, it is a pair and must be discarded. The player gets two points each time a pair is discarded. They get a point deduction if they input two cards that are not a pair. After the end of this round, if no one got deduction, players must have even points. If a player has not seen a pair in hand and proceeded to end his/her turn, he/she will have the option to discard later, although it will only be one point.

After all elimination, players will face each other’s cards and try to guess which is a pair of their card in the opponent’s hand. The player will choose a card in hand, and an opponent card. If it is a pair, 3 points is added to the player, and he/she gets another turn. If it’s not a pair, no point deduction happens. It will simply be the next player’s turn. If cards have not been eliminated after five turns of each players, game ends and machine will tally the score. Announcement of winner will be made and recorded to the “Records” text file.

**How It Works**

****

Once user hit play, the program

will explain the mechanics of

the game. The user will be

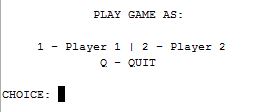
prompted to play the game or not.

Should the user choose not to play,

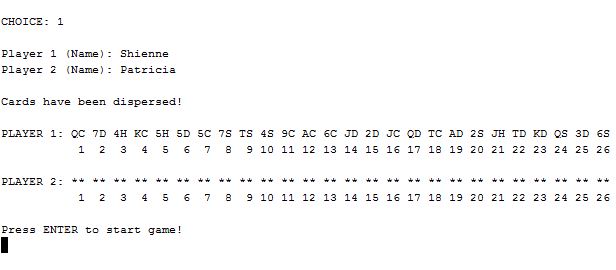
game ends and exits. If the user

decides to play, player will be

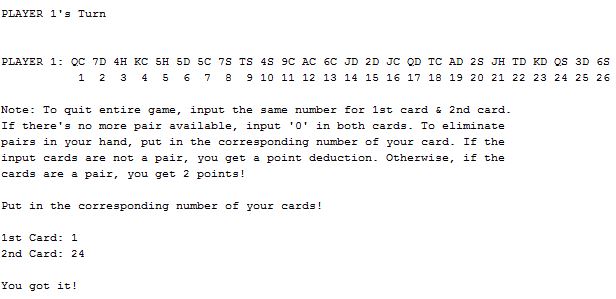
asked if he/she choose to be Player 1 or Player 2 of the game. Whoever the user chooses to be, he/she will have the first turn.



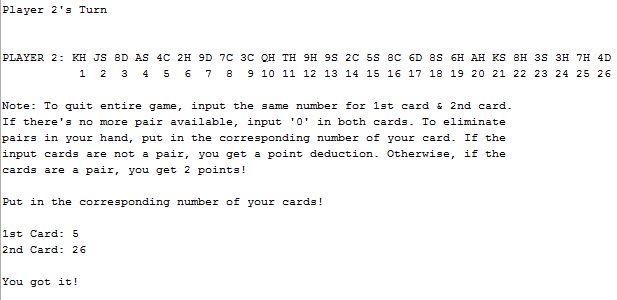
There is still option to quit game in choosing players should the player wants to end game. Once user selects a player, the machine will prompt user for the names of two players for recording purposes. The names are not used until the end of the game. After inputting the names for both players, machine will notify that cards have been dispersed and depending on the user’s player preference, the user’s card will be shown while the opponent’s cards are hidden.



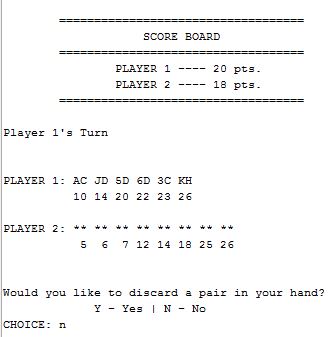
The game will immediately start once user presses ‘Enter’ key.



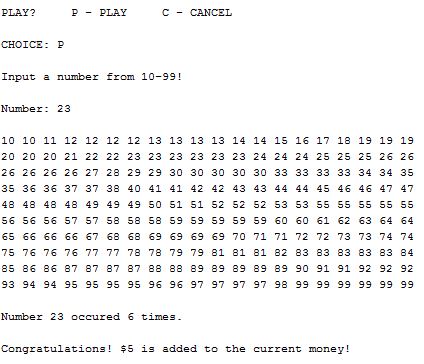
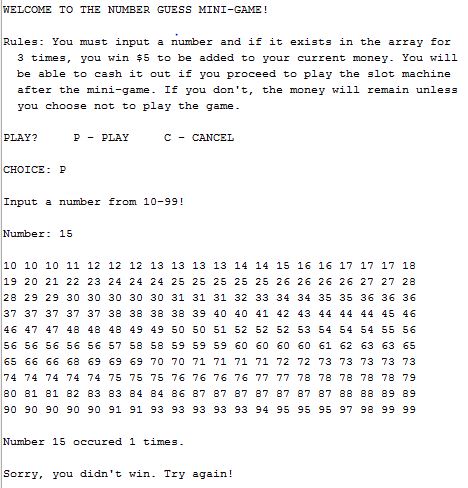
This loop will continue until player 1 sees no more pair and inputs 0 in both cards. And there it becomes player 2’s turn.

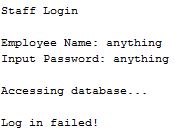
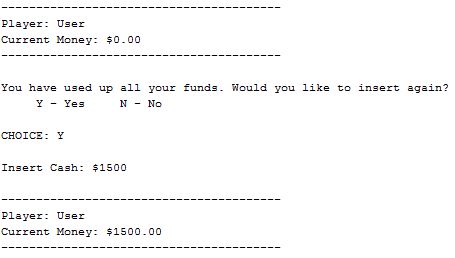
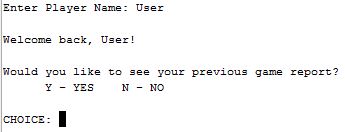
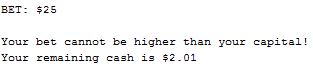


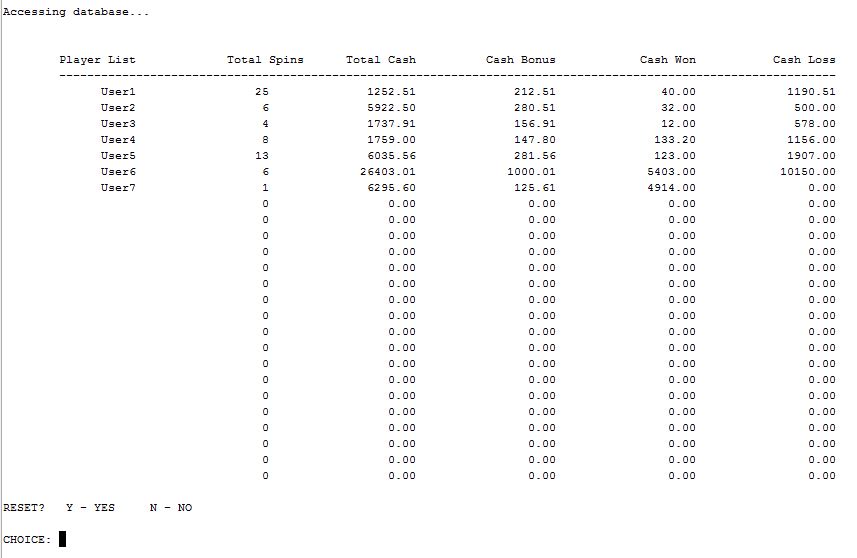
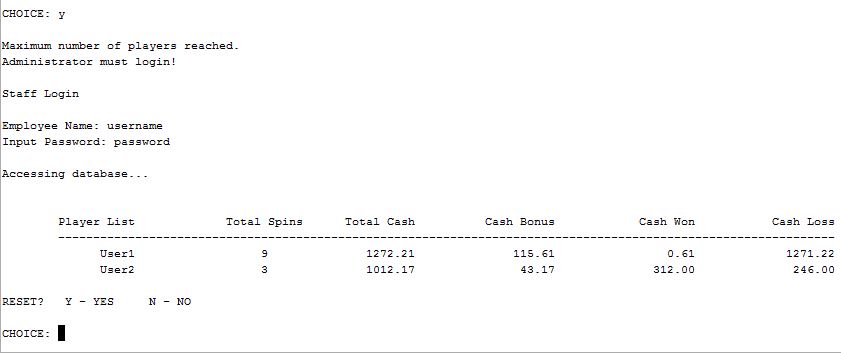
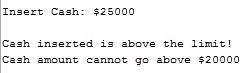
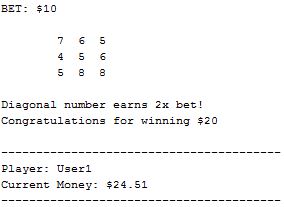
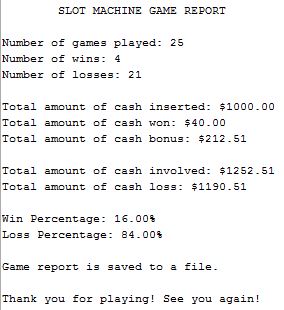
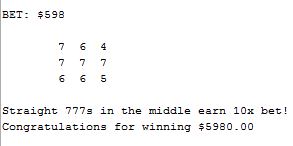
When players are done in eliminating pairs in their hands, score board will be shown and game will ask if player wants to discard a pair if ever they did not see a pair in first elimination.



**Additional Photos/Screenshots**

****

****

****

**Summary**

Project Size: 965 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. 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 cash out 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 and is also paralleled to a vector. 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 remain intact even after the previous player is done. 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 and vector to make a table of information for staff access. I also had to make changes to make the mini-game possible.

**Cross Reference for Projects 1**

|  |  |  |  |
| --- | --- | --- | --- |
| Chapter | Section | Topic | Where in code – Line Number |
| 2 | 2 | Cout | 104-110, 134-137, 160-162, 455-460 and more. |
|  | 3 | Libraries | iostream, iomanip, fstream, cstdlib, cmath, string, ctime, vector |
|  | 4 | Variables/Literals | 51-98, 468, 573 |
|  | 5 | Identifiers | 51-98, 468, 573 |
|  | 6 | Integers | 76-83, 468, 573 |
|  | 7 | Characters | 84-90 |
|  | 8 | Strings | 54-58 |
|  | 9 | Floats no doubles | 64-74 |
|  | 10 | Bools | 94-98 |
|  | 11 | Sizeof \*\*\*\*\* | n/a |
|  | 12 | Variables 7 characters or less | checked |
|  | 13 | Scope \*\*\*\*\* no global variables | n/a |
|  | 14 | Arithmetic Operators | checked – 148-149, 156-157… |
|  | 15 | Comments 20% + | checked |
|  | 16 | Named Constants | 51-52 |
|  | 17 | Programming Style \*\*\*\*\* | n/a |
|  |  |  |  |
| 3 | 1 | Cin | 111, 127, 138, 175, 187, 407, 420, 464 |
|  | 2 | Math Expression | checked – 148-149, 156-157 |
|  | 3 | Mixing data types \*\*\*\*\* | n/a |
|  | 4 | Overflow/Underflow \*\*\*\*\* | n/a |
|  | 5 | Type Casting | 698-699 |
|  | 6 | Multiple Assignment \*\*\*\*\* | n/a |
|  | 7 | Formatting Output | 570 |
|  | 8 | Strings | 54-58 |
|  | 9 | Math Library | 151 |
|  | 10 | Hand Tracing \*\*\*\*\* | n/a |
|  |  |  |  |
| 4 | 1 | Relational Operators | checked – 151-154… |
|  | 2 | If | 118, 155, 424… |
|  | 4 | If-else | 420-421, 386-392… |
|  | 5 | Nesting | 177-140 |
|  | 6 | If-else-if | 545-551, 470-496… |
|  | 7 | Flags \*\*\*\*\* | n/a |
|  | 8 | Logical Operators | checked – 217, 369… |
|  | 11 | Validating User Input | 182-187, 398, 441… |
|  | 13 | Conditional Operator | 151-154 |
|  | 14 | Switch | 122, 409 |
|  |  |  |  |
| 5 | 1 | Increment/Decrement | 199, 223, 395… play++, numWin++, numLose++ |
|  | 2 | While | 182, 475, 915 |
|  | 5 | Do-while | 102, 117, 169, 171, 178, 403, 412, 461 |
|  | 6 | For loop | 584 |
|  | 11 | Files input/output | Input – 128-131 Output – 687-703 |
|  | 12 | No breaks in loops \*\*\*\*\* | checked |
|  |  |  |  |
| 6 | 3 | Prototypes | 29-43 |
|  | 4 | Pass by Value | 29, 33 |
|  | 8 | Returning Values | 898, 931 |
|  | 9 | Boolean return values | 884-885 |
|  | 11 | Static Local Variable | 883, 896, 910 |
|  | 12 | Default Arguments | 32 |
|  | 13 | Reference Variable | 29 |
|  | 14 | Overloading functions | 30-31 |
|  |  |  |  |
| 7 | 4 | Array Initialization | 56, 75, 81 |
|  | 7 | Parallel Arrays | 534-541 |
|  | 8 | Arrays in function arguments | 33, 35-37 |
|  | 9 | 2 Dimensional Arrays | 75 |
|  |  | /as function argument | -- |
|  | 12 | STL Vector | 53 |
|  |  |  |  |
| 8 | 1 | Search Linear/Binary | 484 |
|  | 3 | Sorting Bubble/Selection | 483 |
|  | 5 | Applied to Vectors | -- |
|  |  |  |  |

**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

//Continue loop til budget is 0, 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

//User input validation of float datatype for adding funds

//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

//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 user chose Mini game, welcome user and explain rules

//Prompt user to play or cancel/exit mini game

//Start to fill array

//Prompt user to input number

//User input validation of float datatype for number

//Sort the filled array

//Find out how many times user input's number appeared

//Print sorted array

//Display result

//If user input occurred 3 times or more, add $5 to money

//Add cash won from mini-game to total cash bonus

//If user didn't win, money remains 0

//Reset found count

//If user chooses to cancel game, exit mini-game and back to main menu

//Else, warn user to input right character. Goes back to main menu

//Continue mini-game as long as user chooses to play

//Exit mini-game back to main menu

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

//Set money to 0 if player did not choose to play even if player won in mini game

//Game play exit, back to main menu

//If user enters staff login access

//Prompt user for username and password to log in and access player's important records

//If logged in, display game player's information

//Prompt for reset after viewing records

//If chose to reset, end game

//If not, back to main menu

//If user input is not Y or N, repeat process

//If employee name and password is wrong, display login failed. Go back to main menu

//If begin is not 'N' or 'Y' or ‘M’ or ‘S’, display invalidity

//If game play remains true, continue loop

//Start loop exit condition

//Loop count for 2d array

//Calculate total money involved

//Add important information to database for staff settings

//Increase row

//Decrease column back to 0

//Increment total player count

//Loop voucher one time

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

//Display voucher

//Prompt user for game report

//If yes, display report – function call

//Save report to file

//Reset all data and back to main menu

//If user does not want to see game report

//Save report to file

//Reset all data and back to main menu

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

//If no money left on account

//Prompt user to see game report

//If yes, display report

//Save report to file

//If user does not want to see game report

//Save report to file

//Reset all data and back to main menu

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

//Once condition reached, end game.

//If user decides not to play game, back to main menu and reset the bearned money from mini-game if there is.

**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;

const int ROWS = 25, COLS = 4;

void resetA(int &, int &, int &, float &, float &, float &, float &, float &, float &, float &, float&);

void gameRep(string, char, int, int, int, float, float, float, float, float, float, float);

void gameRep(string, int, int, int, float, float, float, float, float, float, float);

float bonCash(float, float, float, float, float, float, int, int, float = 0);

void find(int [], int, int &, int);

void voucher(string, float);

void filAray(int [], int);

void mrkSrt(int [], int);

void prntAry(int [], int);

void status(float, string);

bool valPass(string, string);

float rndOffB(float, int);

float insert(float &, string);

void winPat();

void askRep();

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

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

const int SIZE = 25;

const int MINSIZE = 200;

vector <int> spins(SIZE);

string name,

dummy;

const float bonCsh1 = 0.05f,

bonCsh2 = 0.10f,

bonCsh3 = 0.15f,

bonCsh4 = 0.25f,

bonCshP = 0.01f,

minVal = 0;

float budget = 0,

add,

bet,

loss = 0,

win = 0,

bonWin = 0,

minCsh = 0,

cshIns = 0,

totCash,

winPer,

lossPer,

data[ROWS][COLS] = {};

int numWin = 0,

numLose = 0,

play = 0,

g8mBon = 10,

rndOff = 100,

array[MINSIZE] = {},

found = 0;

int count = 0, plCount = 0;

char begin,

load,

choice,

report

start,

game,

reset;

unsigned short rn1, rn2, rn3,

rn4, rn5, rn6,

rn7, rn8, rn9;

bool w = true;

bool x = true;

bool y = true;

bool z = true;

bool rep = true;

do {

winPat();

cout<<"\nPlayers Listed: "<<plCount<<endl;

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

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

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

cout<<" M - MINI GAME"<<endl<<endl;

cout<<"Note: Casino Staff access S for settings."<<endl<<endl;

cout<<"CHOICE: ";

cin>>begin;

getline(cin, dummy);

rep=true;

x=true;

z=true;

do {

if (plCount>=SIZE) {

cout<<"\nMaximum number of players reached. \nAdministrator must login!"<<endl;

begin = 'S';

}

switch (begin) {

case 'y':

case 'Y': {

cout<<endl;

cout<<"Enter Player Name: ";

getline(cin, name);

ifstream in;

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

in>>play;

in.close();

if (play>0) {

cout<<"\nWelcome back, "<<name<<"!"<<endl<<endl;

cout<<"Would you like to see your previous game report?"<<endl;

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

cout<<"CHOICE: ";

cin>>report;

if (report == 'Y' || report == 'y') gameRep(name,play,numWin,numLose,cshIns,win,bonWin,totCash,loss,winPer,lossPer);

resetA(play, numWin, numLose, cshIns, win, bonWin, totCash, loss, winPer, lossPer, budget);

names[count] = name;

}

else if (play<=0){

names[count] = name;

cout<<"\nYou are a new player. Welcome "<<name<<"!"<<endl;

}

budget = insert(budget, dummy);

cshIns += (budget-minCsh);

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

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

(budget>=2000)?budget\*bonCsh1:minVal;

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

float 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;

}

budget = rndOffB(budget, rndOff);

status(budget, name);

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++;

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\*14);

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

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

numWin++;

win+=(bet\*14);

}

else {

budget += (bet\*9);

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

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

numWin++;

win+=(bet\*9);

}

}

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;

}

if (play==g8mBon) {

float added = bonCash(added,bonCsh1,bonCsh2,bonCsh3,bonCsh4,loss,g8mBon,play,minVal);

budget+=added;

cout<<"\nCongratulations!\n"

<<"You earned game bonus for reaching "<<g8mBon<<" number of spins!"<<endl;

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

g8mBon+=g8mBon;

bonWin+=added;

}

budget = rndOffB(budget, rndOff);

status(budget, name);

}

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;

budget = rndOffB(budget, rndOff);

status(budget, name);

} 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 'm':

case 'M': {

cout<<"\nWELCOME TO THE NUMBER GUESS MINI-GAME!\n"<<endl;

cout<<"Rules: You must input a number and if it exists in the array for\n"

<<" 3 times, you win $5 to be added to your current money. You will\n"

<<" be able to cash it out if you proceed to play the slot machine\n"

<<" after the mini-game. If you don't, the money will remain unless\n"

<<" you choose not to play the game."<<endl<<endl;

do {

cout<<"PLAY? P - PLAY C - CANCEL"<<endl<<endl;

cout<<"CHOICE: ";

cin>>game;

filAray(array, MINSIZE);

int findN;

if (game == 'p' || game == 'P') {

cout<<"\nInput a number from 10-99!\n\n";

cout<<"Number: ";

cin>>findN;

while (cin.fail()) {

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

cout<<"Number: ";

cin.clear();

getline(cin, dummy);

cin>>findN;

}

mrkSrt(array, MINSIZE);

find(array, MINSIZE, found, findN);

cout<<endl;

prntAry(array, MINSIZE);

cout<<"\nNumber "<<findN<<" occured "<<found<<" times.\n"<<endl

if (found>=3) {

cout<<"Congratulations! $5 is added to the current money!\n"<<endl;

budget+=5;

minCsh+=5;

bonWin+=minCsh;

}

else {

cout<<"Sorry, you didn't win. Try again!\n"<<endl;

budget=budget;

}

found = 0;

}

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

cout<<"\nExiting mini game..."<<endl<<endl;

}

else cout<<"\n\nWarning: You can only enter 'P' or 'C'!\n"<<endl;

} while (game == 'p' || game == 'P');

y=false; //Exit mini-game back to main menu

} break;

case 'n':

case 'N': {

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

budget = 0;

y = false;

} break;

case 's':

case 'S': {

cout<<endl;

cout<<"Staff Login"<<endl<<endl;

cout<<"Employee Name: ";

cin>>empName;

cout<<"Input Password: ";

cin>>input;

cout<<"\nAccessing database..."<<endl<<endl;

if (valPass(input, empName)) {

cout<<"\n\tPlayer List \t Total Spins \t Total Cash \t Cash Bonus \t Cash Won \t Cash Loss"<<endl;

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

for (int cHere = 0; cHere < ROWS; cHere++) {

cout<<setw(19);

cout<<names[cHere]<<setw(19)<<spins[cHere]<<setw(21);

for (int iHere = 0; iHere < COLS; iHere++) {

cout<<data[cHere][iHere]<<setw(20);

}

cout<<endl;

}

cout<<"\nRESET? Y - YES N - NO"<<endl<<endl;

cout<<"CHOICE: ";

cin>>reset;

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

cout<<"\nResetting everything!"<<endl;

y=false;

w=false;

x=true;

}

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

cout<<"\nExiting staff settings..."<<endl<<endl;

y=false;

x=true;

}

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

}

else {

cout<<"Log in failed!\n\n";

x=true; y=false; }

} break;

default: {

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

} break;

}

} while (y);

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

if (x == false) {

int num=0;

totCash = cshIns + bonWin + win;

spins[count] = play;

data[count][num] = totCash; num++;

data[count][num] = bonWin; num++;

data[count][num] = win; num++;

data[count][num] = loss; num++;

count++;

num-=4;

plCount++;

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

if (budget>minVal) {

voucher(name, budget);

do {

askRep();

cin>>report;

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

gameRep(name, report, play, numWin, numLose, cshIns, win, bonWin, totCash,

loss, winPer, lossPer);

rep = false;

x = true;

resetA(play, numWin, numLose, cshIns, win, bonWin, totCash, loss, winPer, lossPer, budget);

}

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

gameRep(name, report, play, numWin, numLose, cshIns, win, bonWin, totCash,

loss, winPer, lossPer);

rep = false;

x=true;

resetA(play, numWin, numLose, cshIns, win, bonWin, totCash, loss, winPer, lossPer, budget);

}

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 {

askRep();

cin>>report;

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

gameRep(name, report, play, numWin, numLose, cshIns, win, bonWin, totCash,

loss, winPer, lossPer);

rep = false;

x=true;

resetA(play, numWin, numLose, cshIns, win, bonWin, totCash, loss, winPer, lossPer, budget);

}

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

gameRep(name, report, play, numWin, numLose, cshIns, win, bonWin, totCash,

loss, winPer, lossPer);

rep = false;

x = true;

resetA(play, numWin, numLose, cshIns, win, bonWin, totCash, loss, winPer, lossPer, budget);

}

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

} while (rep);

}

}

}

else if (begin == 'n' || begin == 'N' || begin == 'y' || begin == 'Y'&&play<=0) {

cout<<"\nPlay the game next time!\n"<<endl;

}

} while (w);

return 0;

}

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;

}