```
Last login: Mon Oct 22 13:03:32 on ttys000
Caijuns-MacBook-Air:~ fennec2000$ ssh qin@linprog5.cs.fsu.edu
                      -=-= AUTHORIZED USERS ONLY =-=-
You are attempting to log into a FSU Computer Science Department machine.
Please be advised by continuing that you agree to the terms of the
Computer Access and Usage Policy of the Department of Computer Science.
                      -=-= AUTHORIZED USERS ONLY =-=-

qin@linprog5.cs.fsu.edu's password:
Last login: Mon Oct 22 13:03:28 2018 from 10.135.140.248
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %                                        %
             %  DEPARTMENT OF COMPUTER SCIENCE  %
              %    Florida State University    %
             %                                      %
            %                                        %
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

          No mudding, IRC, or other games from here.
   Please get a private sector account for non-CS activities.

     See http://www.cs.fsu.edu for departmental information.
      See http://system.cs.fsu.edu for Systems information.
             Send email to help@cs.fsu.edu for help.


                           Attention!!

If you forward your CS email to your FSU emailbox, make sure your FSU
email address is up to date or you may miss important emails.

If you are over your disk or file quota, please check your CS email for
instruction on how to locate and remove files

To download Tectia, go to link https://system.cs.fsu.edu/ssh-tectia/.
If you are off FSU campus, use following ID to access the page:
User Name: sshcs
Password: letmedownloadit

For students: check your CS email at http://webmail2.cs.fsu.edu

qin@linprog5.cs.fsu.edu:~>~vastola/usub/submit1 monty.cpp
********************* File submitted **********************
   Here are the contents of your submitted file:
************************************************************
/* Name: Caijun Qin
Date: 10/17/2018
Section: 6
Assignment: 4
Due Date: 10/22/2018
About this project:
This is a simplified simulation of the game show "Let's Make
```

```
a Deal" from the 1960s-70s. The user has to guess the correct door out of three
doors which contains the winning prize. Any of the other two doors chosen
results in a loss. A winning door is pre-assigned randomly before the user even
begins playing. The user can also check his/her game statistics and reset
them if desired.

Assumptions: The user only inputs real numbers.
All work below was performed by Caijun Qin */

#include <iostream>
#include <iomanip>
#include <ctime>
#include <cstdlib>
#include <cmath>
using namespace std;

/*FUNCTION DECLARATIONS*/
int randomInteger();
void menu(int wins, int losses);
void printDoors();
bool playGame();
void stats(int wins, int losses);
int resetStats();



/*PROGRAM BODY*////////////////////////////////////////////////////////////////
int main(){                                                                  //
    //GAME STATS                                                             //
    int wins = 0;                                                            //
    int losses = 0;                                                          //
                                                                             //
    cout << "Welcome to the Monty Hall Let's Make a Deal Simulator!" << "\n"; //
    menu(wins, losses);                                                      //
                                                                             //
    return 0;                                                                //
}                                                                            //
////////////////////////////////////////////////////////////////////////////////



/*FUNCTION DEFINITIONS*/
//random number generator with results 1, 2, or 3 for the winning door
int randomInteger(){
    srand(time(0));
    int t = rand() % 3 + 1;
    return t;
}

//THE IMPORTANT FUNCTION W/ THE MOST ABSTRACTIONS
void menu(int wins, int losses){
```

```
//variables
double menuChoice = 0;
bool validNum = false;
int menu_choices[4] = {1,2,3,4};

cout << "Choose one of the following:" << "\n";
cout << "1 - Play Game" << "\n";
cout << "2 - View Stats" << "\n";
cout << "3 - Reset Stats" << "\n";
cout << "4 - Quit Game" << "\n";
cout << "> ";
cin >> menuChoice;

//check for valid input off the menu
while(validNum == false){
    //first of all, positive integer value is checked
    if(menuChoice == abs(trunc(menuChoice))){
        //assesses whether the menuChoice input is valid or not
        for(int counter = 0; counter < 4; counter++){
            if(menuChoice == menu_choices[counter]){
                validNum = true;
            }
        }
        //error message
        if(validNum == false){
            cout << "Invalid option, please retry > ";
            cin >> menuChoice;
        }
    }
    //error message
    else{
        cout << "Invalid option, please retry > ";
        cin >> menuChoice;
    }
}

//passes specific function for each menu choice
if(menuChoice == 1){
    //outcome must equal gameOutcome from playGame()
    bool outcome = playGame();
    if(outcome == true){
        wins++;
    } else if(outcome == false){
        losses++;
    }
    menu(wins, losses);
} else if(menuChoice == 2){
    stats(wins, losses);
    menu(wins, losses);
} else if(menuChoice == 3){
    wins = resetStats();
    losses = resetStats();
```

```
        menu(wins, losses);
    } else if(menuChoice == 4){
        cout << "Goodbye! \n";
    }

    return;
}

//functions pertaining to each menu choice

//MENU CHOICE 1
//ASCII Art doors (to be used when Play Game is selected)
void printDoors(){
    cout << "_________    _________    __________ \n";
    cout << "|       |    |  ____ |    |  _____ | \n";
    cout << "|   |   |    |     | |    |      | | \n";
    cout << "|   |   |    |  ___| |    |   ___| | \n";
    cout << "|   |   |    | |     |    |      | | \n";
    cout << "|   |   |    | |____ |    |  ____| | \n";
    cout << "_________    _________    __________ \n";
    return;
}

//WHEN THE USER CHOOSES TO PLAY THE GAME
bool playGame(){
    //variables
    int winningDoor = randomInteger();
    double firstPick = 0;
    double secondPick = 0;
    int losingDoor1 = winningDoor;
    int losingDoor2 = winningDoor;
    bool validDoor = false;
    bool gameOutcome = false;
    int door_choices[3] = {1,2,3};

    //body of function
    cout << "Pick a Door, 1, 2, or 3!" << "\n\n";
    printDoors();
    cout << "\n";
    cout << "Choice >  ";
    cin >> firstPick;

    //check for valid door number for the first door
    while(validDoor == false){
        //first of all, positive integer value is checked
        if(firstPick == abs(trunc(firstPick))){
            //asssesses whether the first door input is valid or not
            for(int counter = 0; counter < 3; counter++){
                if(firstPick == door_choices[counter]){
                    validDoor = true;
                }
            }
```

```
            //error message
            if(validDoor == false){
                cout << "Invalid option, please retry > ";
                cin >> firstPick;
            }
        }
        //error message
        else{
            cout << "Invalid option, please retry > ";
            cin >> firstPick;
        }
    }
    //the truncate built-in function ensures that only a positive natural number
    //prints out for a door number
    cout << "You chose door #" << trunc(firstPick) << "! \n";
    cout << "I'll now open a door for you randomly that you didn't choose! \n";

    //generating an unchosen losing door
    while(losingDoor1 == winningDoor || losingDoor1 == firstPick){
        losingDoor1 = randomInteger();
    }

    cout << "Opening door #" << losingDoor1 << " and it's a GOAT! \n";
    cout << "Now, I'll be a kind host and give you the chance to switch your doo
r! \n";

    //generating the remaining losing door
    while(losingDoor2 == winningDoor || losingDoor2 == losingDoor1){
        losingDoor2 = randomInteger();
    }
    cout << "Would you like door #" << winningDoor << " or #" << losingDoor2 <<
"? > ";
    cin >> secondPick;

    //check for valid door number picked on the second try
    while(secondPick != abs(trunc(secondPick)) || (secondPick != winningDoor &&
secondPick != losingDoor2)){
        cout << "Invalid option, please retry > ";
        cin >> secondPick;
    }

    //the truncate built-in function ensures that only a positive natural number
    //prints out for a door number
    cout << "Opening door #" << trunc(secondPick) << "..... \n";

    //determines if the player won or lost
    //the gameOutcome is true for win and false for lose
    if(secondPick == winningDoor){
        cout << "it's a BRAND NEW CAR!!!! YOU WIN!!!! \n\n\n\n";
        gameOutcome = true;
    } else if(secondPick == losingDoor2){
        cout << "baaaaaaaaaaaa.... it's a GOAT!!!! You LOSE! \n\n\n\n";
```

```
        gameOutcome = false;
    }
    return gameOutcome;

}

//MENU CHOICE 2
void stats(int wins, int losses){
    //calculations for the win and lose percentages
    double numGames = wins + losses;
    double winRate = wins / numGames * 100;
    double loseRate = losses / numGames * 100;

    //displays the game stats
    if(numGames == 0){
        cout << "No stats to display, 0 games played. \n";
    } else {
        //printing out the game stats
        cout << "Results after " << numGames << " games:  \n";
        cout << "Wins: " << wins << "\t\t" << "Losses: " << losses << "\n";
        cout << fixed << showpoint << setprecision(1) << "Win %:  " << winRate <
< "\t\t" << "Loss %:  " << loseRate << "\n";
    }
    return;
}

//MENU CHOICE 3
int resetStats(){
    int resetValue = 0;
    return resetValue;
}
```

```
********************* END FEEDBACK ************************
*    PLEASE VIEW ABOVE TO VERIFY THE CONTENTS OF         *
*              YOUR SUBMITTED FILE                       *
***********************************************************
qin@linprog5.cs.fsu.edu:~>
```