Numerical Hangman

Version 2

Jason Emerson

CSC5

Dr. Mark Lehr

Fall 2016

48102

**Introduction**

Hangman has been a game played for many, many years. The game can be enjoyed by a single person or by many people trying to complete a word before the other players can. The game itself doesn’t have too many variations, minus increasing the difficulty by allowing less lives (body parts) or by making it a timed game.

**Game Variation**

My version of Hangman has quite the variation to it. The most significant change to the playstyle of the game is the removal of words. It a generic Hangman game, the player guesses letters until he is able to solve the word or loses the game due to no more available lives. In this version of Hangman, your goal is to guess the randomly generated number in the set amount of lives. There are no letters to guess nor any words to guess. Now you may ask, how does the game become more difficult or how does one win the game? This second version of Hangman has an infinite amount of levels. Thus, in turn, you can continue to play and reach a higher score until you have ran out of lives. In turn of having infinite levels, there is no win condition to this game. You continue to play until you lose. To make it fair to the random condition of the game, once you have used up all your available lives, you will have a chance to get one more attempt. You will be given the ability to guess a random number from a list of random numbers and if you guess one of the numbers in that list, you will be given another chance to guess the number in the game, thus continuing the game. If you fail to guess one number in the list of numbers, you will lose the game.

**Code Information**

This is my own little chart of what I have in my game. The ProjectCrossListingFinal.xls spreadsheet we made in class and was uploaded to Github will be in my Project folder. I have everything that is on that list. I have not added everything in here, I have added majority of the major things. **DISCLAIMER: Code has been slightly altered after write-up has finished. Lines may vary slightly as listed compared to code. Lines should be within’ a maximum range of +or- 20 line of listed line(s).**

|  |  |
| --- | --- |
| *Code Line Information* | |
| *Total Lines of Code (including comments)* | *525* |
| *Total Lines of Comments* | *147* |
| *Total White Spaces (blank lines)* | *55* |
| *Total cout lines (all lines beginning with cout)* | *52* |
| *Total lines of pure code (total lines – comment lines – white spaces – cout lines = lines of pure code)* | *271* |

|  |  |
| --- | --- |
| **C++ Libraries** | |
| **<iostream>** | **All code** |
| **<cstdlib>** | **Lines 163, 207** |
| **<ctime>** | **Line 142** |
| **<fstream>** | **readHighScore, fillAry, fillAry, highScore Functions** |
| **<string>** | **readHighScore, fillAry, fillAry, highScore Functions** |
| **<iomanip>** | **Lines 123 – 127, 425** |
| **<cmath>** | **Line 196** |
| **<vector>** | **fillAry & prntAry Functions** |
| **Constructs** | |
| **If Statements (individual)** | **Lines 94, 190, 298, 332, 425, 460, 474, 501, 520** |
| **If/Else Statements** | **Lines 175, 182, 210, 215, 220** |
| **Nested If Statements** | **Line 198 & 210 (nested together)** |
| **Switch Case** | **Lines 53-69, 261-283** |
| **While Loop** | **Lines 160, 374, 401** |
| **Do-While Loop (first line do, second line while – will not include code inside do-while loop)** | **Lines 52/70, 88/97, 171/225, 329/343, 457/471** |
| **For Loop** | **Lines 237, 238, 295, 296, 423, 459, 482, 490, 519** |
| **Nested For Loop** | **Lines 237/238, 295/296** |
| **Primitive Data Types** | |
| **Char** | **Lines 112** |
| **Short** | **Lines 48, 440** |
| **Int** | **Lines 49, 85, 146, 147, 148, 149, 150, 152, 153, 155, 156, 254, 317, 318, 328, 336, 420, 441, 442, 456, 464** |
| **String** | **Lines 320, 321, 333, 367, 443, 447, 448, 461** |
| **Float** | **Lines 152** |
| **Boolean** | **Lines 151, 157, 327, 455** |
| **Arrays, Vectors & Constants** | |
| **Arrays** | **Lines 154, 155, 315, 445** |
| **Vectors** | **Lines 316 & 446** |
| **Constants** | **Lines 21, 145, 314, 444** |
| **File I/O** | |
| **ifstream** | **Lines 366, 394** |
| **ofstream** | **Line 439** |
| **in.open** | **Lines 371 & 398** |
| **out.open** | **Lines 481 & 489** |
| **.close()** | **Lines 379, 405, 485, 493** |
| **.clear()** | **Lines 380, 406, 486, 494** |
| **in** | **Lines 374 & 401** |
| **out** | **Lines 483 & 491** |
| **Random Number Functions & Time** | |
| **rand()** | **Lines 163 & 207** |
| **srand()** | **Line 142** |
| **time()** | **Line 142** |
| **Formatting** | |
| **setw()** | **Lines 123-127, 425** |
| **Function Prototypes** | |
| **void** | **Lines 25 -35** |
| **int** | **Line 24** |
| **bool** | **Line 36** |
| **Pass By Function & Overloading** | |
| **Pass By Reference** | **Line 404** |
| **Pass By Value** | **Line 314** |
| **Overloaded Function** | **Line 33 & 34** |
| **Function Return & Default Parameters** | |
| **Default Parameters** | **Line 38** |
| **Multiple Returns** | **Line 521 & 524** |
| **Array Types** | |
| **Parallel Arrays** | **Line 315 & 316** |
| **2 Dimensional Array** | **Line 154** |
| **Pass-In 2 Dimensional Array** | **Lines 236, 252, 294** |
| **Searching & Sorting** | |
| **Bubble Sort (array & vector)(swap)** | **Lines 506-525** |
| **Linear Search (array)** | **Lines 487-495** |
| **Math** | |
| **Power** | **Line 196** |

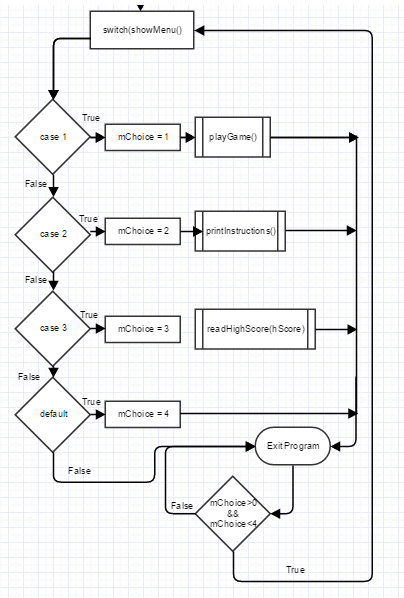
**Game Logic**

***FLOWCHARTS MAY LOOK DIFFERENT IN WRITE-UP COMPARED TO ACTUAL FLOWCHART. PLEASE FOLLOW LINK TO VIEW FINAL FLOWCHART***

*Beginning of Game & showMenu Function:*

Pseudo-Code for the beginning menu is as follows;

Player inputs choice between 1 – 4

if choice is <1 and >4 then error message is displayed asking the player to please enter a number between 1 and 4 and the player is requested to renter his/her choice. 

Main Flowchart

switch(showMenu){

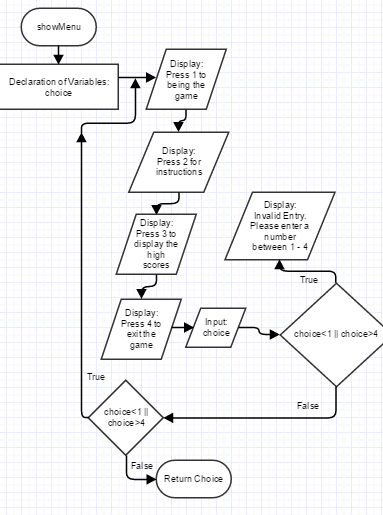
As 1 is entered, case 1 is triggered. The main is ended and we are now in the playGame function.

As 2 is entered, case 2 is triggered. The main is ended and we are now in the printInstructions functions.

As 3 is entered, case 3 is triggered. The main is ended and we are now in the readHighScore function.

As 4 is entered, case 4 is triggered. The main is ended and the program is ended.

When the game ends, this do while loop is active again allowing the player to enter his choice once again.

**

showMenu Function Flowchart

**Game Logic**

*printInstructions Function:*

Pseudo-Code for the instructions is as follows;

Case 2 is triggered on the beginning menu

Enter printInstructions function

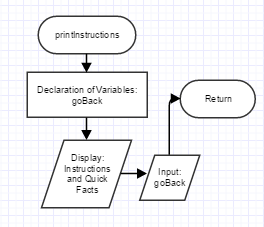
Display (cout) all of the instructions I have typed out

Display (cout) Quick Facts:

Display (cout) the quick facts I have typed out at a setw(10)

Display (cout) telling the user to press any key to return to the Main Menu

Player enters any key triggering the goBack variable to return to the menu



printInstructions Function Flowchart

**Game Logic**

*playGame Function:*

Pseudo-Code for the game is as follows;

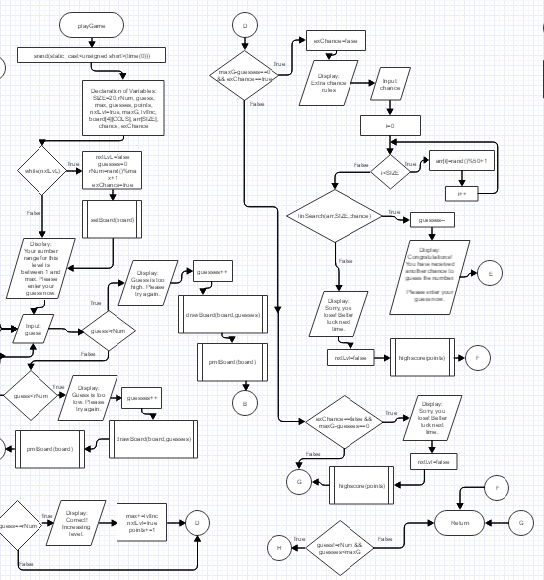
While beginning of each level

Set nxtLvl to false

Set guesses = 0

Generate the new random number rNum=rand()%max+1

Call the setBoard function to set the board to a 4x3 of –

 Set the flag to allow an extra chance to true exChance=true

Display to the player that his range is from 1 to the current max and ask to enter his guess

Do the following while guess!=rNum&&guesses<maxG

Have player input guess

If players guess is greater than rNum

Display that the guess is too high.

Increment guesses by 1

Call the drawBoard function adding a body part

Call the prntBoard function displaying the current board

If player guess is lower then rNum

playGame Flowchart (missing other parts)

Display that the guess is too low.

Increment guesses by 1

Call the drawBoard function adding a body part

Call the prntBoard function displaying the current board

If player guess is equal to rNum

Display that the guess is correct

Increase the max by 5

Set nxtLvl flag to true triggering the beginning of level loop

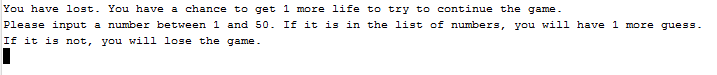
Add 1 point to the players points

If player runs out of guesses and exChance is set to true

Set exChance to false triggering the ability to gain one more chance

Display that the player has a chance to gain one more life and must correctly guess one number from an array of 20 random numbers ranging from 1 to 50.

Allow player to input his chance guess



In Game image of Extra Chance instructions

After player inputs his chance begin a for loop with i=0 and i<SIZE of 20

Randomize arr[i]=rand()%50+1

Call the linSearch function and if the players chance was correctly in the array, display that they have received another chance and ask them to enter their guess now.

If the number was not in the array using the linSearch function

Display that the number was not in the array of numbers and they have lost

Set nxtLvl flag to false and send their points to the highScore function determining if they have reached a new high score or not.

Else if they had gained an extra chance, guessed the correct number and once again used up all of their lives on the next level

Display Sorry, you lose! Better luck next time.

Set nxtLvl flag to false and send their points to the highScore function determining if they have reached a new high score or not.

**Game Logic**

*setBoard, drawBoard & prntBoard:*

Pseudo-Code for the usage of the Hangman board is as follows;

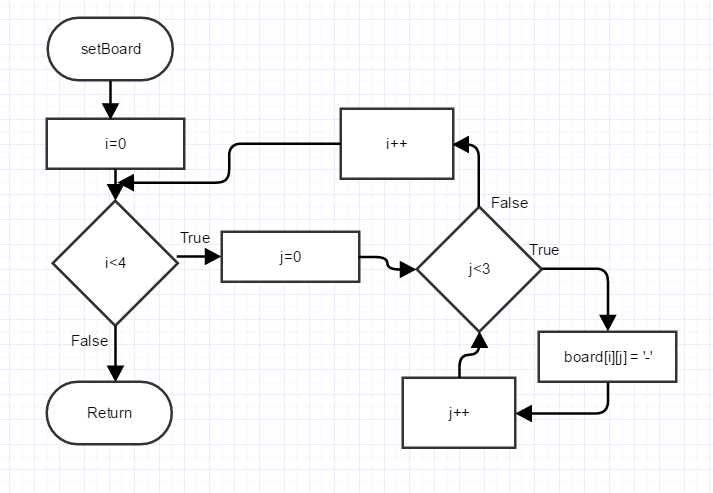
Use a for loop to set i=0 and i<4 to set 4 active rows

Use another for loop to set j=0 and set j<3 to set 3 active columns

Set the entire board[i][j] = ‘’-‘

Exit the 2nd for loop incrementing until it is at 3

Exit the 1st for loop incrementing until it is at 4



setBoard Function Flowchart

Now that the board is drawn we need to draw the body parts

Set maxG = 7 and display how many guesses the player is at and how many guesses are remaining

switch(guesses)

guesses 7 = board[3][2] = \\

guesses 6 = board[3][0] = /

guesses 5 = board[2][1] = ^

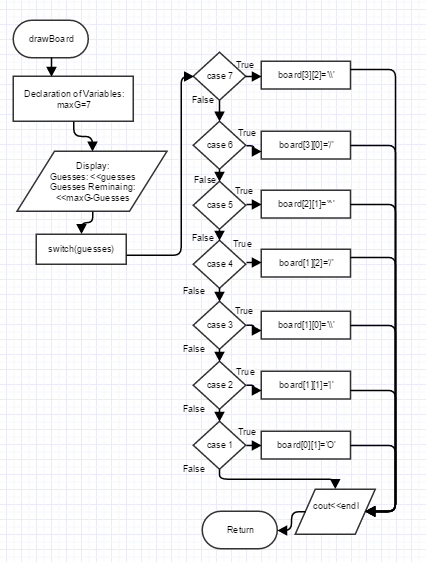
guesses 4 = board[1][2] = /

guesses 3 = board[1][0] = \\

guesses 2 = board[1][1] = |

guesses 1 = board[0][1] = O

cout a new line after each part has been drawn



drawBoard Function Flowchart

The final part to the Hangman board is displaying the board.

Initialize the first for loop with i=o and set i<4.

Initialize a second for loop with j=0 and set j<3

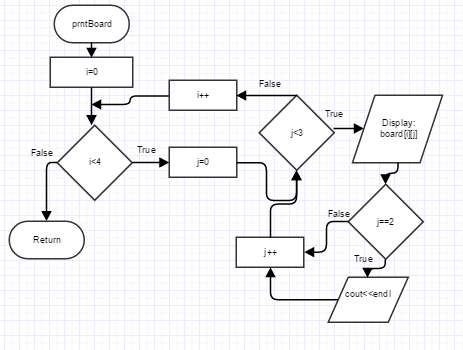
Display the array board[i][j] with i representing the rows and j representing the columns

If j==2

cout a new line

End the columns loop by incrementing j if it is not at the set size

End the rows loop by incrementing I if it is not at the set size



prntBoard Function Flowchart

**Game Logic**

*readHighScore, fillAry, fillAry, prntAry:*

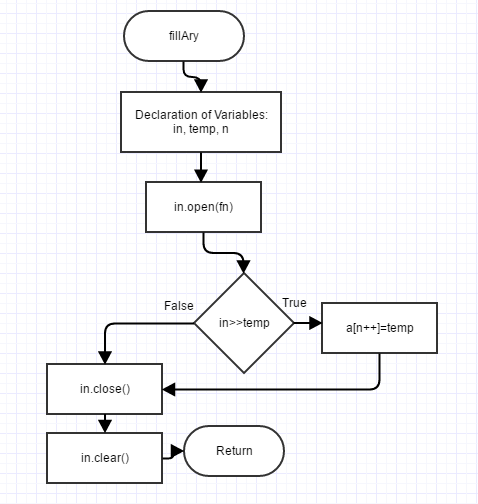
Pseudo-Code for the process and output of the high score is as follows;

Declare the filestream in and open the stream

While (in>>temp)

Increase the array a with strings from the file to fill all the names

Then close the file and clear it once the array is full



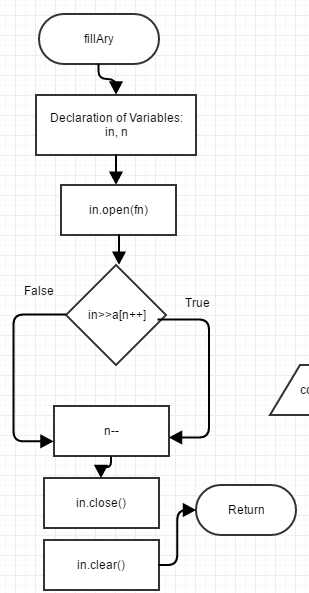
fillAry Function Flowchart – Names Array

Declare the filestream in and open the stream

While (in>>a[n++])

Decrement from the array

Close and Clear the scores file



fillAry Function Flowchart – Scores Array

To display the array we use a function named prntAry

Initialize the count = 0

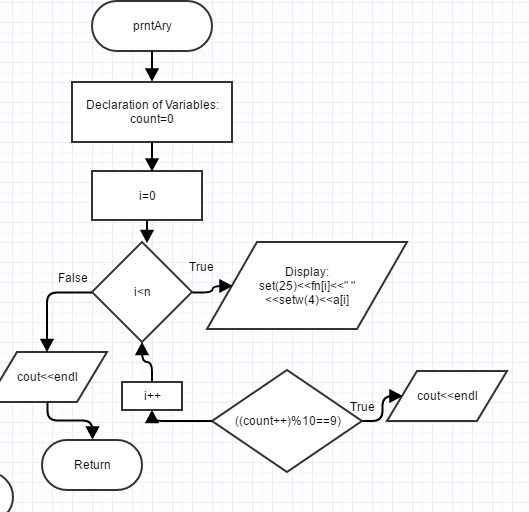
Initialize a for loop with i=0 and set the size of i<n (called in the parameters)

cout the names array setw(25) and then cout the scores array set(4)

if ((count++)%10==9)

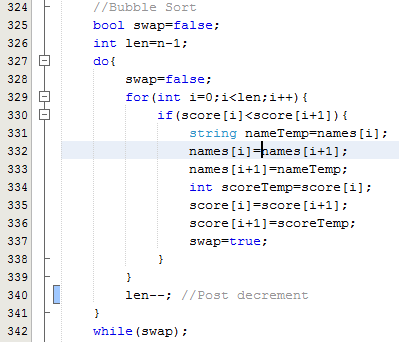
cout<<endl

End the for loop and cout<<endl



prntAry Function Flowchart

To display the high scores we use the previous functions plus we sort it

Declare variables in this function

Ex Bubble Sort Code

Call fillAry(fnNames, names, n) function

Call fillAry(fnScores, score, m) function

Use a bubble sort to sort the arrays (ex code provided to the right)

Display a high score title and call the prntAry function

**Game Logic**

Above is majority of the functions given with pseudo code and brief explanations of how they work as well example parts of my flowchart. Disclaimer: Some of the flowcharts may be missing parts of their flow after being cropped. They are not to be taken as the final flowchart image. If you would like to view my entire final flowchart, please click [here](https://www.gliffy.com/go/publish/11589509) to visit the webpage (Gliffy) with my Hangman V2 flowchart. If for some reason the here is not hyperlinked you can also follow the link below.

**Flowchart:** <https://www.gliffy.com/go/publish/11589509>

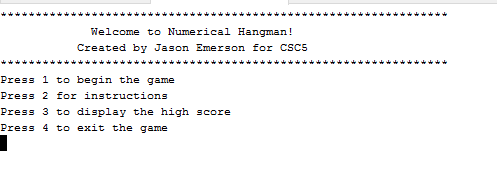
**Github Information**

**Github:**

https://github.com/jemerson93/EmersonJason\_48102

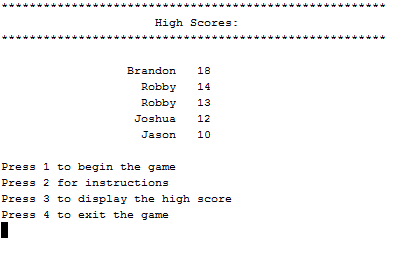
**Versioning:** https://github.com/jemerson93/EmersonJason\_48102/tree/master/Proj/Project%202/Hangman\_V2

**In GAME/Running**

Below will be some images of my game running as well as brief explanation of what is going on in the image.

Main Menu

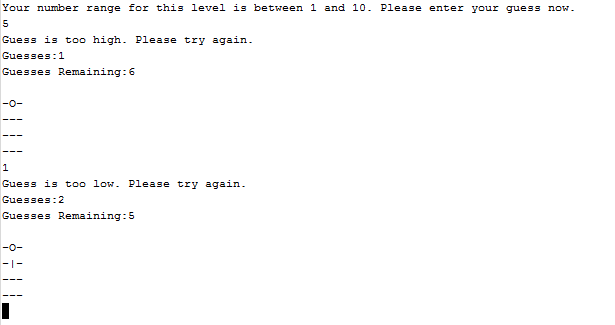
When the game starts, this is what you will see. You will make a selection here.

Let’s say you want to view the high scores? So we enter 3 and press enter. We now see this being displayed…

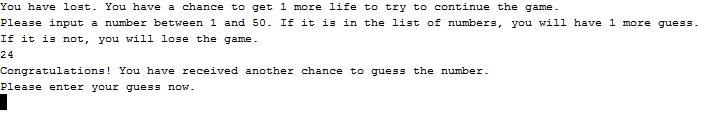
High Scores

These high scores currently are accurate scores saved from friends that tested my game functionality for me to try and break it. None were experienced with programming. (disregarding Jason)

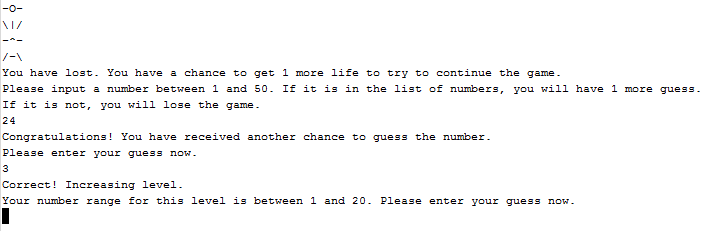
So you want to play, right? Press 1! So let’s say you have taken two guesses now. One guess was too high and one guess was too low. You would see the image below.



In Game Guessing

Once you have lost the game, you will trigger the ability to possibly gain one more attempt at guessing the number. Below are some examples of that running in game. 

Correctly Guessing a Number in the Array



Also Correctly Guessing the Random Number in Game Moving to the Next Level

**Final Comments**

I just wanted to add some concluding comments. After this section is ended, I will have all of my code on the final few pages shown clearly. Making this game was a lot of fun for me. My initial Hangman variance (Project 1) was not as challenging or time consuming as I would have preferred. My goal for Project 2 was to completely revamp my Hangman game not only with the new concepts we learned in the last few months of class (i.e. arrays, sorting, searching, vectors) but to also challenge myself to try to continue adding to the game until the final few days. My biggest struggle in the game was forming the Two-Dimensional Array to out the Hangman Board. I think that specifically took me a couple of days to finally nail down. The other struggle I had was using a Bubble sort to sort the names/scores arrays while keeping them together as I had them outputted as a parallel array (i.e cout<<fn[i]<<” “<<a[i]<<endl. That took me about an entire day. I very much enjoyed making this game though and as my Major I am pursuing is Computer Science, I cannot wait to see what future games I will be able to make for future classes and what future challenges I can overcome and how far I can push myself.

**C++ Code**

Here is my program code. Keep in mind if you would like to see my official code, please view the .cpp file inside the submitted folder. This code has had comments removed to be viewable clearly in this document.

/\*

File: main

Author: Jason Emerson

Created on November 30th, 2016, 4:30 PM

Purpose: Version 2 of Numerical Hangman

\*/

//System Libraries

#include <iostream> //Input/Output Objects

#include <cstdlib> //Random Number

#include <ctime> //Random Number Seed

#include <fstream> //File I/O

#include <string> //Strings

#include <iomanip> //Formatting

#include <vector> //Vector

#include <cmath> //Math

using namespace std; //Name-space used in the System Library

//User Libraries

//Global Constants

const int COLS = 100;

//Function Prototypes

int showMenu();

void printInstructions();

void playGame();

void drawBoard(int);

void readHighScore(float);

void highScore(float);

void setBoard(char [][COLS]);

void drawBoard(char [][COLS],int &);

void prntBoard(char [][COLS]);

void fillAry(string,string [],int&);

void fillAry(string,vector<float> &,int&);

void prntAry(string [],vector<float> &,int);

bool linSearch(int [],int,int);

void bubbSort(string [],vector<float> &,int=5);

//Execution Begins Here!

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

//Welcome

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<" Welcome to Numerical Hangman!"<<endl;

cout<<" Created by Jason Emerson for CSC5"<<endl;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

//Declaration of Variables

short mChoice=0;

int hScore;

//Display the Main Menu and Get User's Choice

do{

switch(showMenu()){

case 1:

mChoice=1;

playGame();

break;

case 2:

mChoice=2;

printInstructions();

break;

case 3:

mChoice=3;

readHighScore(hScore);

break;

default:

mChoice=4;

break;

}

}while(mChoice>0&&mChoice<4);

//Exit Program

return 0;

}

//Function Comment Removed in Word Document

int showMenu(){

//Declaration of Variables

int choice=0; //Player's main menu choice

//Input Choice and Return To Main

do{

cout<<"Press 1 to begin the game"<<endl;

cout<<"Press 2 for instructions"<<endl;

cout<<"Press 3 to display the high score"<<endl;

cout<<"Press 4 to exit the game"<<endl;

cin>>choice;

if(choice<1||choice>4){ //Error checking

cout<<"Invalid entry. Please enter a number between 1 - 4"<<endl;

}

}while(choice<1||choice>4);

//Exit Function and Return Choice

return choice;

}

//Function Comment Removed in Word Document

void printInstructions(){

//Declaration of Variables

char goBack; //Return to Main Menu

//Instructions

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<" Instructions"<<endl;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<endl;

cout<<"This version of Hangman is quite different from the Hangman we all know and love. Instead of guessing letters or words, your objective is to guess the number the game has generated."<<endl;

cout<<"If you guess the correct number, the game level will increase. If you do not guess the correct number within' the amount of applicable guesses, you will lose the game."<<endl;

cout<<"Below are some quick facts on the game. Good luck and remember to have fun!"<<endl;

cout<<endl;

cout<<setw(50)<<"Quick Facts:"<<endl;

cout<<setw(10)<<"â€¢ There is an unlimited amount of levels in this game."<<endl;

cout<<setw(10)<<"â€¢ In each level, you have a total amount of 7 lives."<<endl;

cout<<setw(10)<<"â€¢ When you complete each level, you gain 1 point."<<endl;

cout<<setw(10)<<"â€¢ Once you have used all 7 lives, you will have a chance to gain an additional life. If you fail, you lose."<<endl;

cout<<endl;

cout<<"Press any key and enter to return to the Main Menu."<<endl;

cin>>goBack;

}

//Function Comment Removed in Word Document

void playGame(){

//Set the Random Number Seed

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

//Declaration of Variables

const int SIZE=20; //Size of Linear Search array

int rNum=0; //Random Number Generated

int guess=0; //Players guess

int max=10; //Max Number for the Random Number Generator

int guesses=0; //Number of GUesses Used

float points=0.0; //Players Score

bool nxtLvl=true; //If true, enter next level. If false, do not

int maxG=7; //Max number of guesses allowed

int lvlInc=5; //Amount to increase the random number range by

char board[4][COLS]; //Character hangman board - initial board

int arr[SIZE]; //Linear Search Array

int chance; //Chance for extra life

bool exChance; //Flag for extra chance in each level

//Beginning of Level

while(nxtLvl){

nxtLvl=false; //Set to false so loop doesn't activate if player loses

guesses=0; //Start guesses over for new level

rNum=rand()%max+1; //Generate a random number between 1 and the max available

setBoard(board); //Initiate the hangman board as a 4x3 of '-'

exChance=true; //Set extra chance to true

//Players Guess

cout<<"Your number range for this level is between 1 and "<<max<<". Please enter your guess now."<<endl;

//Incorrect Guess

do{

cin>>guess; //Players input guess

//High Guess

if(guess>rNum){

cout<<"Guess is too high. Please try again."<<endl;

guesses++; //Increment guesses

drawBoard(board,guesses); //Draw the hangman body part pending on the number of guesses user has

prntBoard(board); //Display the Hangman board and update after each guess

//Low Guess

}else if(guess<rNum){

cout<<"Guess is too low. Please try again."<<endl;

guesses++; //Increment guesses

drawBoard(board,guesses); //Draw the hangman body part pending on the number of guesses user has

prntBoard(board); //Display the Hangman board and update after each guess

}

//Correct Guess

if(guess==rNum){

cout<<"Correct! Increasing level."<<endl;

max+=lvlInc; //Increase the max for the range by the initialization of lvlInc

nxtLvl=true; //Set next level to true to initiate the loop

points+=pow(2,2); //Square the current points squared and add to the current points

}

//Players Extra Chance

if(maxG-guesses==0&&exChance==true){

exChance=false;

cout<<"You have lost. You have a chance to get 1 more life to try to continue the game."<<endl;

cout<<"Please input a number between 1 and 50. If it is in the list of numbers, you will have 1 more guess."<<endl;

cout<<"If it is not, you will lose the game."<<endl;

cin>>chance;

//Randomize Array

for(int i=0;i<SIZE;i++){

arr[i]=rand()%50+1;

}

if(linSearch(arr,SIZE,chance)){

guesses--;

cout<<"Congratulations! You have received another chance to guess the number."<<endl;

cout<<"Please enter your guess now."<<endl;

}

else{

cout<<"Sorry,the number you have entered was not in the array of numbers. You lose! Better luck next time."<<endl;

nxtLvl=false; //Set to false to break the loop

highScore(points); //Run highScore function and send points to function

}

}else if(exChance==false&&maxG-guesses==0){

cout<<"Sorry, you lose! Better luck next time."<<endl;

nxtLvl=false; //Set to false to break the loop

highScore(points); //Run highScore function and send points to function

}

}while(guess!=rNum&&guesses<maxG);

}

}

//Function Comment Removed in Word Document

void setBoard(char board[][COLS]){

for(int i=0;i<4;i++){

for(int j=0;j<3;j++){

board[i][j]='-';

}

}

}

//Function Comment Removed in Word Document

void drawBoard(char board[][COLS],int &guesses){

//Declaration of Variables

int maxG=7; //The max number of guesses to dignify guesses remaining

//Guesses and Guesses Remaining

cout<<"Guesses:"<<guesses<<endl;

cout<<"Guesses Remaining:"<<maxG-guesses<<endl;

//Hangman Board Per Guess

switch(guesses){

case 7:

board[3][2]='\\';

break;

case 6:

board[3][0]='/';

break;

case 5:

board[2][1]='^';

break;

case 4:

board[1][2]='/';

break;

case 3:

board[1][0]='\\';

break;

case 2:

board[1][1]='|';

break;

case 1:

board[0][1]='O';

break;

}

cout<<endl;

}

//Function Comment Removed in Word Document

void prntBoard(char board[][COLS]){

for(int i=0;i<4;i++){ //Rows

for(int j=0;j<3;j++){ //Columns

cout<<board[i][j];

if(j==2){

cout<<endl;

}

}

}

}

//Function Comment Removed in Word Document

void readHighScore(float hScore){

//Declaration of Variables

const int SIZE=6; //Size =of array

string names[SIZE]; //Name array

vector<float> score(SIZE); //Score Vector

int n; //Size names

int m; //Size scores

string fnNames="names.dat"; //Names

string fnScores="scores.dat"; //Scores

fillAry(fnNames,names,n); //Fill the names array from the file

fillAry(fnScores,score,m); //Fill the scores array from the file

//Bubble Sort

bubbSort(names,score,n);

//Display the High Scores

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<" High Scores:"<<endl;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<endl;

//Output the scores from both arrays sorted

prntAry(names,score,n);

}

//Function Comment Removed in Word Document

void fillAry(string fn,string a[],int &n){

//Declaration of Variables

ifstream in;

string temp;

n=0;

//Open the stream

in.open(fn);

//Read in data

while(in>>temp){

a[n++]=temp;

}

//Close the file

in.close(); //Close the file

in.clear(); //Clear the file

}

//Function Comment Removed in Word Document

void fillAry(string fn,vector<float> &a,int &n){

//Declaration of Variables

ifstream in;

n=0;

//Open the file

in.open(fn);

//Read in the data

while(in>>a[n++]);

n--;

//Close the file

in.close(); //Close the file

in.clear(); //Clear the file

}

//Function Comment Removed in Word Document

void prntAry(string fn[],vector<float> &a,int n){

//Declaration of Variables

int count=0; //counter

//Output and Display the Array

for(int i=0;i<n;i++){

cout<<setw(25)<<fn[i]<<" "<<setw(8)<<a[i]<<endl;

if((count++)%10==9)cout<<endl;

}

cout<<endl;

}

//Function Comment Removed in Word Document

void highScore(float points){

//Declaration of Variables

ofstream out; //Output File

short hScore; //High Score

int n;

int m;

string name; //Input name

const int SIZE=100; //Size of the arrays

string names[SIZE]; //Name Array

vector<float> score(SIZE); //Score Vector

string fnNames="names.dat";

string fnScores="scores.dat";

//Fill arrays

fillAry(fnNames,names,n); //Fill names array

fillAry(fnScores,score,m); //Fill scores array

//Bubble Sort

bubbSort(names,score,n);

//If New High Score

if(points>score[4]){

cout<<"Congratulations! You reached a new high score. Please enter your name below."<<endl;

cin>>name;

names[4]=name;

score[4]=points;

//Out new name replacing lowest score name

out.open("names.dat");

for(int i=0;i<5;i++){

out<<names[i]<<endl;

}

out.close(); //Close the file

out.clear(); //Clear the file

//Output new score replacing lowest score

out.open("scores.dat");

for(int j=0;j<5;j++){

out<<score[j]<<endl;

}

out.close(); //Close the file

out.clear(); //Clear the file

cout<<"Saved to file!"<<endl; //Error check to make sure it saves to file

cout<<endl;

}

//If Not a new High Score

if(points<score[4]){

//Display High Score

cout<<"Sorry, not a new high score."<<endl;

cout<<endl;

}

}

//Function Comment Removed in Word Document

bool linSearch(int range[],int n,int key){

//Linear Search

for(int i=0;i<n;i++){

if(key==range[i]){

return true; //If number is found return true

}

}

return false; //If number is not found return false

}

//Function Comment Removed in Word Document

void bubbSort(string names[],vector<float> &score,int n){

//Bubble Sort

bool swap=false;

int len=n-1;

do{

swap=false;

for(int i=0;i<len;i++){

if(score[i]<score[i+1]){

string nameTemp=names[i];

names[i]=names[i+1];

names[i+1]=nameTemp;

int scoreTemp=score[i];

score[i]=score[i+1];

score[i+1]=scoreTemp;

swap=true;

}

}

len--; //Post decrement

}while(swap);

}