

Mastermind.

Project1

CSC- 17a – 48983 C++

Jonathan Balisky

25 – October – 2015

Content

1.Introduction………………………………………….3-4  
Rule and Gameplay  
Thoughts after Program

2. Development………………………..………………4-5  
Approach Strategy

3. Research…………………………………………..…..5

Arrays

Parallel array

4. Variables list…………………………….………….…6

5. Topic Covered (Checklist)……………………………7

6. Libraries included……………………………………..7

7. Flowchart

8. Code

1. Introduction

**Rules and Gameplay**

This version of mastermind allows for two modes of gameplay easy or normal. In the easy mode the player has to guess a 3 digit pin, in normal a 4 digit pin. Regardless of which mode is selected the program will make each digit, of the pin, some number between 0-9. Each time after the player guesses, the program will tell the player how many digits are correct and are in the right place, and how many digits are the right digits, but not in the right place. The program will then display how many guess the player has left. If the player fails to guess the right pin in 9 tries, the program will display all the guesses that the player submitted, then sort and display them with and numerical sort and ask the player if they would like to try again.

**Thoughts after Program**

Next version of this game should start to rewrite the program in a modular fashion. Roping in related variables into constructs with constructors and de-structers . Then tying into that functions rather then in the program itself. I believe this will greatly “clean” up the code and make it easier for future editors to understand. Finally I could place a log in a construct which would contain the players name, age, amount of games played, and the average of how many number of guesses it took to guess the answer. The log would then be updated each and every time the player plays and the user can then check his ranking against other players before or after he/she plays.

2. Development

Approach Strategy

In updating Mastermind my main purpose was to implement concepts I had recently learned in class. The goal was to take a already working piece of code and implement more advance functions and tools to remove clutter and reduce the lines of code

Because most of the logic was already there from the previous version game I went through the code again looking for places to implement several changes. These items included more the use of dat files, advance read write functions, character array, structures, and pointers to structures.

3. Research

1. Structures   
   Using structures in this program as a more simplified way to multiple variable in a single variable. Also allowed the use of pointers to structurs
2. Structure pointers  
   By applying the use of pointers passing multiple variables or referencing multiple variables in a function became much easier. Instead of having to pass in both xs and oxs I could simply pass in the pointer to the structure.
3. Batch writing to Binary files  
   Wanted a way to shorten up the lines of code needed for writing data to a file. By changing the format of my files from .txt to .dat and writing in binary format I was able to write a whole array of answers from the user in one line. No for loop needed Nice!

4. Variables list

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Variable Name** | **Description** |  | **Line** |
| int | Level | How many digits the pin is |  | 33 |
|  | counter | How many times the player guessed |  | 44 |
|  | answer[SIZE] | Array for the answer |  | 35 |
|  | Indx | Location for the table |  | 265 |
| char | \*\* table | Dynamic array for the players guesses |  | 43 |
| string | temp | Temporary place holder for the file |  | 36 |
|  | usrG | For the player to input guess |  | 39 |
| bool | match[SIZE] | check whether user digits matches answer |  | 34 |
|  | Swap | Check to see if swap was made |  | 264 |
| const int | SIZE | Size of arrays |  | 27 |
| ofstream | output |  |  | 37 |

5. Topic Covered (Checklist)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter** | **Type** | **Code** | **File** | **Line** |
| Constructor | Game |  | Game.cpp |  |
| Destructor | ~Game |  | Game.cpp |  |
| Template |  |  | Game.h |  |
|  |  |  |  | 55 |
|  |  |  |  | 38 |
|  |  |  |  | 264 |
| Memory Allocation |  |  |  | 39 |
|  |  |  |  | 33 |
|  |  |  |  | 97 |
|  |  |  |  | 124 |
| 2 style | comment | //variables |  | 31 |
| 3 boolean expression | != | if(usrG.length() != level){ |  | 189 |
|  | <, >, || | if (usrG[i] <48 || usrG[i] > 57){ |  | 195 |
| 3 multiway branches | if | if(usrG.length() != level){ |  | 189 |
|  | else | else{ |  | 193 |
|  | nested | do{ |  | 65 |
|  |  | if(input.is\_open()){ |  | 45 |
|  |  | for(int j = 0; j<level;j++){ |  | 173 |
| 3.3 type of loop | for | for(int i = 0; i<9; i++){ |  | 172 |
|  | do-while | while(temp[0]!=49 && temp[0]!=50); |  | 70,71 |
| 4 predefined function | srand, time | srand(time(0)); |  | 41 |
|  | rand | answer[i] = rand()%10; |  | 159 |
| 4function prototypes | Int | int compare(int [], short&, string, … |  | 21 |
|  | Bool | bool isvalid(string, int); |  | 20 |
| 5 void function | void | void gssHst(string, int, char \*\*, int); |  | 22 |
| 5 call-by-reference | & | int compare(int [], short&, string, bool … |  | 21 |
| 6 streams and basic | ofsream declare | ofstream output; |  | 37 |
|  | Ifstream declare | Ifstream input |  | 38 |
|  | Input | input.open("instructions.txt"); |  | 52 |
|  | Input.close | input.close(); |  | 58 |
|  | output | output.open("Answer.txt"); |  | 83 |
|  | close | output.close(); |  | 86 |
| 7 array | int array | Int answer[SIZE] |  | 34 |
|  | bool aray | Bool mathc[SIZE] |  | 35 |
|  | char\*\* | table = new char \*[guess]; ... |  | 195 |
| 10 String Objects | string | string temp; |  | 44 |
| 10 Conversion | toupper | toupper(usrG[0]) == 'Y') |  | 157 |
| 10 Char array | Char array[] | char sAnswer[SIZE]; |  | 42 |
| 11 Structured data | struct Numbers | struct Numbers{ |  | 21 |
| 11 Combing data types | Numbers | struct Numbers{  int xs; //How many x's  short os; //How many o'x  }; |  | 21-24 |
| 11 Dot operator | nmbr.xs | cout << "X(s)=" << nmbr.xs << endl; |  | 130 |
| 11 Structure initialization | Struct Numbers | struct Numbers{  int xs; //How many x's  short os; //How many o'x  }; |  | 21-24 |
| 11 Structures in Arguments | nPtr->xs == level | if (nPtr->xs == level) { |  | 137 |
| 12 Binary files | "Answer.dat" | inOut.open("Answer.dat", ios::out | ios::binary | ios::app); |  | 103 |
| 12 Writing to binary | inOut.write | inOut.write(sAnswer, sizeof (sAnswer)); |  | 107 |
| 12 Batch Writing to binary with array | inOut.write | inOut.write(sAnswer, sizeof (sAnswer)); |  | 107 |
| 12 Reading binary files | "instructions.dat", ios::in | inOut.open("instructions.dat", ios::in); |  | 61 |
|  |  |  |  |  |

6. Libraries included

* <cstdlib>
* <iostream>
* <ctime>
* <fstream>

7. Pseudo code

Set time seed

Output instruction from file

Do

Ask user for easy or normal

Do

Call prepare function

Random answer and initialize other variables

Create table

Output the answer to file

Game start

do

Input guess number

Call isvalid to check validation

Call gssHst

Call compare function

Display Xs and Ox (result)

Guess -1

While guess>0 and guess answer is not correct

Sort answers

Delete table

If x==level output win

Else output lose

Ask for another new game

While(Yes)

7. Flowchart

Main



Main



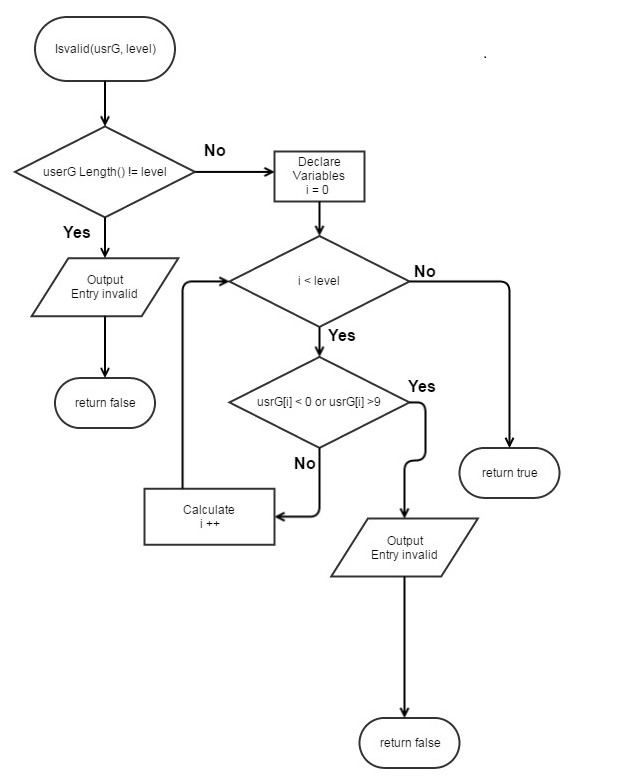
Main



Prepare



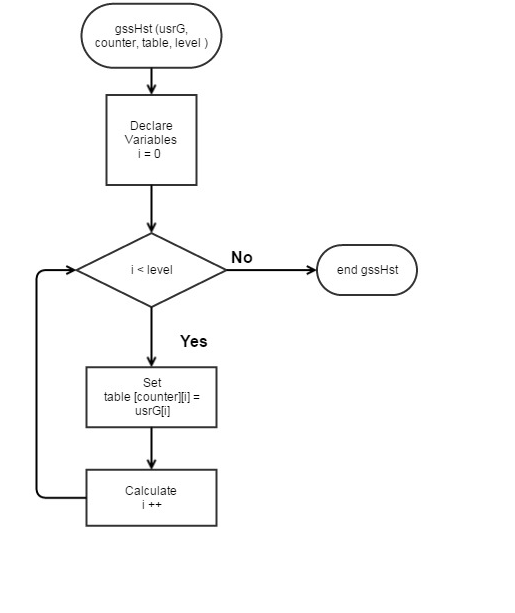
isValid



Compare



gssHst



Sort



8. Input output (raw)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* \*\*Mastermind jr.\*\* \*\*

\*\* You have to guess numbers that are between 0-9 within 9 guesses! \*\*

\*\* First select either easy or normal. easy = 3 numbers normal = 4 \*\*

\*\* First type in three or four numbers as your guess \*\*

\*\* Then the game will show you the amount of Os and Xs you have for that guess. \*\*

\*\* An O means that one of your numbers is correct but it is in the wrong position \*\*

\*\* An X means that you have a number correct and in the right possition! \*\*

\*\* So guess away! But remember you only have 9 guess. Run out of guesses and you lose. \*\*

\*\* Do you have what it takes? Can you decode the sequence? \*\*

\*\* We will soon find out...;) \*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*What difficulty would you like to play? 1 for Easy or 2 for Hard: 2

Input your guess:

12314

Please enter 4 numbers.

Input your guess:

1231215

Please enter 4 numbers.

Input your guess:

dfe

Please enter 4 numbers.

Input your guess:

1234

x=0

o=3

Guesses left: 8

Input your guess:

9999

x=1

o=0

Guesses left: 7

Input your guess:

5689

x=0

o=1

Guesses left: 6

Input your guess:

342

Please enter 4 numbers.

Input your guess:

2315

x=2

o=1

Guesses left: 5

Input your guess:

2912

x=4

o=0

Guesses left: 4

Congrats you won!

Unsorted answers from you!!

1234

9999

5689

2315

2912

Sorted answers via bubble sorts !!

1234

2315

2912

5689

9999

Play again Y/N?

1

Play again Y/N?

4

Play again Y/N?

s

Play again Y/N?

Y

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* \*\*Mastermind jr.\*\* \*\*

\*\* You have to guess numbers that are between 0-9 within 9 guesses! \*\*

\*\* First select either easy or normal. easy = 3 numbers normal = 4 \*\*

\*\* First type in three or four numbers as your guess \*\*

\*\* Then the game will show you the amount of Os and Xs you have for that guess. \*\*

\*\* An O means that one of your numbers is correct but it is in the wrong position \*\*

\*\* An X means that you have a number correct and in the right possition! \*\*

\*\* So guess away! But remember you only have 9 guess. Run out of guesses and you lose. \*\*

\*\* Do you have what it takes? Can you decode the sequence? \*\*

\*\* We will soon find out...;) \*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*What difficulty would you like to play? 1 for Easy or 2 for Hard: 1

Input your guess:

123

x=0

o=0

Guesses left: 8

Input your guess:

123

x=0

o=0

Guesses left: 7

Input your guess:

123

x=0

o=0

Guesses left: 6

Input your guess:

1234

Please enter 3 numbers.

Input your guess:

543

x=0

o=0

Guesses left: 5

Input your guess:

674

x=1

o=1

Guesses left: 4

Input your guess:

865

x=1

o=0

Guesses left: 3

Input your guess:

456

x=0

o=0

Guesses left: 2

Input your guess:

345

x=0

o=0

Guesses left: 1

Input your guess:

86

Please enter 3 numbers.

Input your guess:

567

x=1

o=1

Guesses left: 0

Sorry but you ran out of guesses. :(

Unsorted answers from you!!

123

123

123

543

674

865

456

345

567

Sorted answers via bubble sorts !!

123

123

123

345

456

543

567

674

865

Play again Y/N?

N

See you again next time!

RUN SUCCESSFUL (total time: 1m 21s)

/\*

File: Game.cpp

Author: Jonathan Balisky

Created on July 25, 2015, 9:18 pM

Purpose: Mastermind jr.

\*/

//Libraries

#include <iostream>

#include <string>

#include <cstdlib>

#include <fstream>

#include <ctime>

//User libraries

using namespace std;

//structured data

struct Numbers{

short xs; //How many x's

short os; //How many o'x

};

//Functions Prototypes

char \*\* prepare(int [], int&, bool[], int, char[], Numbers \*);

bool isvalid(string, int );

void compare(int [], string, bool[], int, Numbers \*) ;

void gssHst(string, int, char \*\*, int);

void sort(char \*\*, int, int );

//Global

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

const int SIZE = 4;

//Variables

int level = 3; //Difficulty of game

int answer[SIZE]; //number of pin

char sAnswer[SIZE]; //Answer

bool match[SIZE]; //Which numbers are matched

string temp; //For the file output

fstream inOut; // for file stream

string usrG; //The users guess or input

int guess; //How many guess the user had guessed

char \*\*table; //Table of the user guesses

int counter = 0; //Row counter for table

Numbers nmbr; //Variable for structure

Numbers \*nPtr = &nmbr;//Pointer to numbers

srand(time(0)); //setting time seed

do {

counter = 0;

// cout<<"Call prepare."<<endl;//For diagonostics

inOut.open("instructions.dat", ios::in);

if (inOut.is\_open()) {

while (getline(inOut, temp)) {

cout << temp;

}

inOut.close();

} else {

cout << "Instructions failed to open" << endl;

}

do {

do {

cout << "What difficulty would you like to play? 1 for Easy or 2 for normal: ";

getline(cin, temp);

} while (temp.length() != 1); // User did not enter 1 digit

} while (temp[0] != 49 && temp[0] != 50); // User did not enter 1 or 2

if (temp[0] == 49) { //Level is easy

level = 3;

} else { //User selected hard level

level = 4;

}

table = prepare(answer, guess, match, level, sAnswer, nPtr); //Initialize

// cout<<"Call prepare."<<endl;//For diagonostics

// //Out Put answer to a file

// inOut.open("Answer.dat", ios::out);

//

// if(inOut.is\_open()){

//

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

// inOut<<answer[i];

// }

// inOut.close();

// }

// else{

// cout<<"Failed to write answer to file";

// }

//

//Output answer as Binary

inOut.open("Answer.dat", ios::out | ios::binary | ios::app);

if (inOut.is\_open()) {

inOut<<endl;

inOut.write(sAnswer, sizeof (sAnswer));

inOut.close();

}

else {

cout << "Failed to write answer to file" << endl;

}

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

//cout<<answer[i]; //For diagonostics

// }

do {

do {

cout << "Input your guess: " << endl; //User enter guess

getline(cin, usrG);

//cin.ignore();

} while (isvalid(usrG, level) == false); //Loop until user enters valid answer

gssHst(usrG, counter, table, level);

counter++; //gssHst Ran

compare(answer, usrG, match, level, nPtr);

cout << "X(s)=" << nPtr->xs << endl; //Right numbers in right space

cout << "O(s)=" << nPtr->os << endl; //How many Correct number but in the incorrect space

guess--;

cout << "Guesses left: " << guess << endl;

} while (guess > 0 && nPtr->xs != level); // User out of guess or has guess correctly

if (nPtr->xs == level) { //user won

cout << "Congrats you won!" << endl;

} else { //user lost

cout << "Sorry but you ran out of guesses. :( " << endl;

}

sort(table, counter, level);

for (int i = 0; i < 9; i++) {

delete table[i];

}

delete[] table;

do { //Checking for an input of Y or N

do { //Checking for input over 1 char

cout << "Play again Y/N?" << endl;

getline(cin, usrG);

} while (usrG.length() != 1);

} while (toupper(usrG[0]) != 89 && toupper(usrG[0]) != 78);

//}while(usrG[0] > 'Y' || usrG < 'N' || (usrG[0]>'N' && usrG[0]<'Y'));

} while (toupper(usrG[0]) == 'Y');

cout << "See you again next time!" << endl;

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Prepare\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: Initializing values for the game.

\* Input: answer, guess, match, level, nPtr

\* Output:

\* table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

char \*\* prepare(int answer[], int &guess, bool match[], int level, char sAnswer [],Numbers \*nPtr) {

char \*\*table;

guess = 9;

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

answer[i] = rand() % 10; //creating answer from 0-9

sAnswer [i] = answer[i] + 48; //ascii equivalent to numbers

match[i] = false; //Set all to false

//cout<<"answer = "<<answer[i]; //For diagonostics

}

// answer[0]=3;

// answer[1]=3;

// answer[2]=4;

//

table = new char \*[guess]; //Creating 2 d dynamic array

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

table[i] = new char[level];

}

// cout<<endl;

nPtr->xs = 0;

nPtr->os = 0;

for (int i = 0; i < guess; i++) {//Filling the array with empty spaces

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

table[i][j] = ' ';

}

}

return table;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*isvalid\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: To check whether or not the user entered 3 numbers

\* Input: usrG, level

\* Output: True or false

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

bool isvalid(string usrG, int level) {

// cout<<"Call isvalid."<<endl; //For diagonostics

if (usrG.length() != level) {

cout << "Please enter " << level << " numbers." << endl;

return false;

} else {

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

if (usrG[i] < 48 || usrG[i] > 57) {

cout << "Number not entered" << endl;

return false;

}

}

// cout<<"number valid"<<endl;//For diagonostics

return true;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Compare\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: To compare the user's guess with the answer and return how many

\* were correct or incorrect.

\* Input: answer, usrG, match, level, nPtr

\* Output:none

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void compare(int answer[], string usrG, bool match[], int level, Numbers \*nPtr) {

nPtr->xs = 0;

nPtr->os = 0;

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

match[i] = false; //int all values to zero again

}

//Checking for correct numbers in the right position

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

if (answer[i] == (usrG[i] - 48)) { //subtracting 48 makes it an integer

nPtr->xs++;

//cout<<"match["<<i<<"] = true"<<endl;

match[i] = true;

}

}

//Checking for os

for (int i = 0; i < level; i++) { //i is position of answer

for (int j = 0; j < level; j++) { //j is position of usrG(user guess)

if (j != i && match[i] == false && answer[i] == usrG[j] - 48) {

// // cout<<"Match["<<i<<"] = "<<match[i]<<" usrG["<<j<<"] = "<<usrG[j]<<endl

// // <<"Answer["<<i<<"] = "<<answer[i]<<endl;

nPtr->os++;

match[i] = true;

}

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*gssHst\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: To keep a record of the the user's guesses

\* Input: usrG, counter, table

\* Output:none

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void gssHst(string usrG, int counter, char \*\*table, int level) {

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

table[counter][i] = usrG[i];

}

// cout<<"\n"; //For Diagnositics

// for(int i=0;i<9;i++){

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

// cout<<table[i][j];

// }

// cout<<"\n";

// }

// cout<<"\n\n\n\n";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Sort\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Purpose: To Sort all the user guesses to show I can do it...so HA! Using a bubble sort

\* Input: table, counter, level

\* Output:none

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void sort(char \*\*table, int counter, int level) {

bool swap = false; //For the bubble swap function

int indx = 0; //Second index location for table

cout << "Unsorted answers from you!!" << endl;

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

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

cout << table[i][j];

}

cout << endl;

}

do {

swap = false;

for (int i = 0; i < counter - 1; i++) { //counter -1 because bubble swap is always columns - 1

indx = 0;

while (table[i][indx] == table[i + 1][indx] && indx < level)indx++; //Checking to see if current

//index is equal then going to next location if they

//are equal.

if (table[i][indx] > table[i + 1][indx]) { //If that row and col. not equal

//then check if first is larger if it is then swap all numbers

//cout << endl << "i = " << i << endl;

for (int j = 0; j < level; j++) { //Swapping each 2 rows and their respective columns until all the rows are swapped

// cout << "Table[i] = " << table[i][j] << "table[i+1] = " << table[i + 1][j] << endl; //For diagnostics

table [i][j] = table [i][j]^table[i + 1][j]; //in place swap. to hopefully make Dr lehr happy so he give me extra credit

table [i + 1][j] = table [i][j]^table[i + 1][j];

table [i][j] = table [i][j]^table[i + 1][j];

}

swap = true;

}

}

} while (swap == true);

cout << "Sorted answers via bubble sorts !!" << endl;

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

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

cout << table[i][j];

}

cout << endl;

}

}