Game Project Final

<Mastermind Game>

CSC 17A- 42824

Heidy Tamayo

Date: 06/08/16

**Introduction**

Title: Mastermind Game

Mastermind is played online where there is a table showing four empty slots allowing the user to pick a round marble color. The colors to choose from are red, blue, black, green, white, orange, brown and yellow. The computer generates its own colors and the user must guess which of the colors the computer has picked. The user would normally pick the colors allowed but in this program the user types in the color in order from left to right (ex: blue red green black). The user’s input is then compared to that generated by the computer and is told if they have made a wrong guess. This program asks for the name they would associated with the game and is allowed as many turns as they like to play the game. However, they will see a statement after the tenth try that they have lost the game. The program gives a hint indicating if there is a color they have inputted in the correct spot and correct color by outputting “X” and if they just have the correct color an “0” is outputted. Otherwise, it is left blank. The user can keep guessing the colors the computer has generated up to the end of the number of tries the user inputted or if the user guesses all the colors correctly. If the user cannot guess the colors, the message displays they have lost the game. If the user has won, the percent accuracy is generated along with the number of tries it took the user to complete the game. At the end of the game whether win or lose, the computer asks the user if they would like to play again or end the program.

**Summary**

Project size: 832 lines

Concepts Utilized from Chapters 9-16:

* Structures
* Pointers with arrays
* Arrays of structures
* Structures passed through functions
* String Objects and class
* Reading a binary file
* Opening a binary file
* Allocated memory
* Function that returns structure objects
* Inputting into a structure variable and outputting
* Creating classes
* Deriving from one class to another class
* Class operators
* Templates

Concepts Utilized:

* Character data types
* Integer data types
* Float data types
* Boolean data types
* Ternary operators
* Single if-else statements
* Expanded if-else statements (if, else if, else)
* Formatting (setting the precision of a decimal)
* Single if
* Switch decision
* While loops
* Do-while loops
* For loops
* File (opening a file containing the outputted results)
* Random number seed
* pass by value calling a value to a function and then outputting it
* pass by reference where the & sign comes in the function
* defaulted parameters
* returning primitive data types
* outputted to file
* arrays
* parallel arrays
* searching required
* 2D arrays
* passing 2D arrays into/out of function

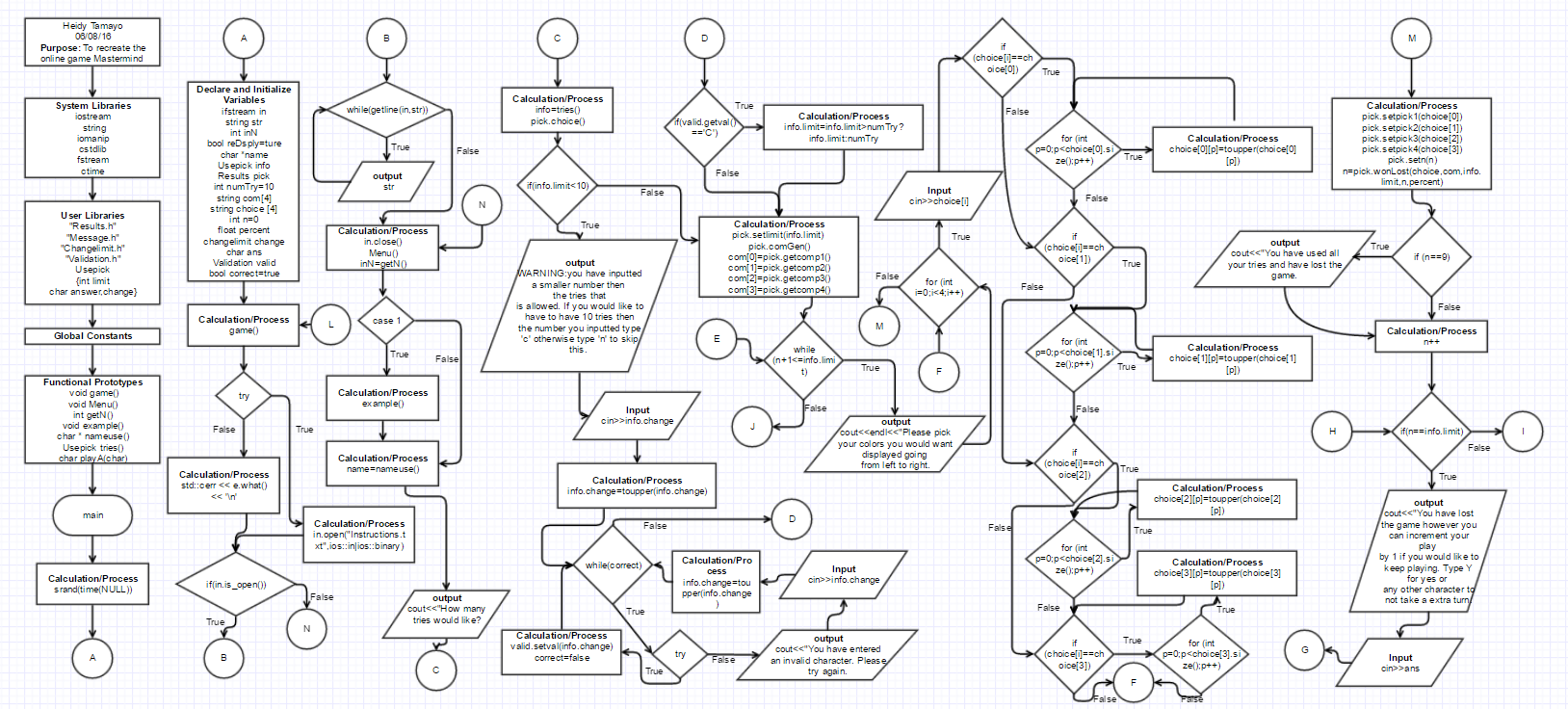
I improved the game by not making it so obvious when the user has inputted the correct color and in the correct spot. This is by the output of the hint says an “X” but it doesn’t indicate which one is in the correct spot just that one is correct. Another improvement is indicating the color that is correct but not in the right spot. I was able to drastically shorten the hints by implementing a for loop instead of various if else statements. I would like to improve the game by making a chart for the user to make it easier to use the process of elimination. Also if possible be allowed to save the user’s information in a separate memory so that if the user would like to see their information in the future they can.

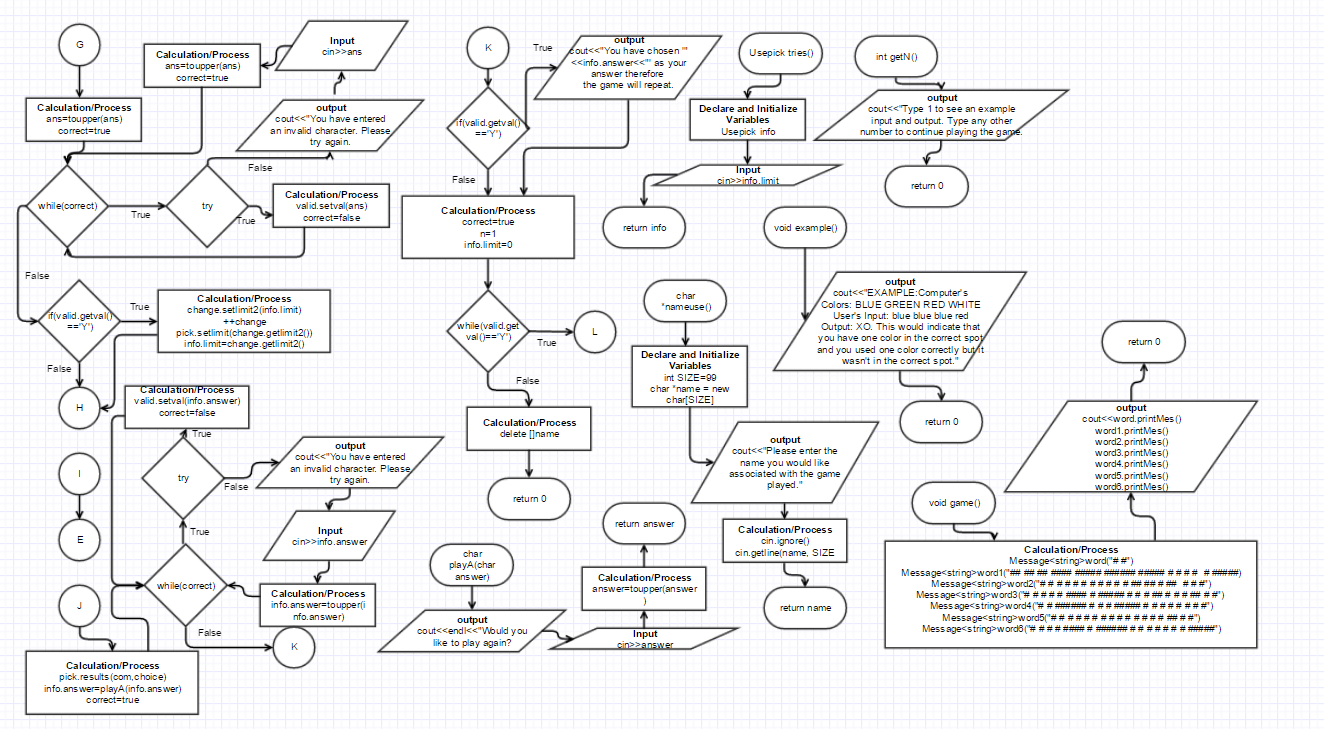
**Description**

The main point of the program is to have the user think through the game based on the hints the computer generates and allow the user to guess the colors the computer has generated randomly through the process of elimination without the consequence that they would run out of turns. This is mainly since they can request more turns.

**Flowchart**

The flowchart was too big to fit into one screen shot and have is legible so I have put the whole flowchart and sections to show each portion.





**Pseudo Code**

Declaring the variables

Shows the title mastermind

Tests to open the file and to read the directions

Closes the file

Asks the user if they would like to see an example

Asks user for the name and number of tries

Tells the user how many colors to pick from

If the number of tries is less than ten conforms with the user if that is what they want

Tests if what the user has inputted is a correct character

Sets the limit

Generates the computer’s colors and sets them in main

Begins the tries the user has inputted and gets the user’s colors

Compares the user’s colors to the computer

Determines if the user is correct or incorrect

After ten turns it is considered they have lost

They are allowed to play one extra turn if they would like after all the user’s turns are up

Shows the results of the user

Asks the user if they would like to play again the game

**Major Variables**

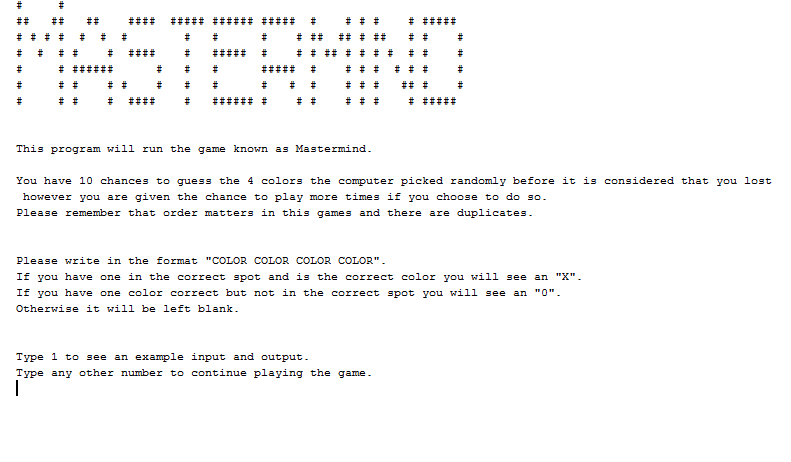
|  |  |  |
| --- | --- | --- |
| **Type** | **Variable Name** | **Description** |
| int | limit | The limit the user inputs |
| string | Com[4] | An array of 4 of the computer pick |
| string | Choice[4] | An array of 4 of the choice of the student |
| ifstream | in | File that contains all the directions |
| Int | NumTry=10 | The number of tries the user has before they lose the game |
| Changelimit | Change | Allows the user to change the limit if the user wishes to |
| char | answer | The answer the user inputs |
| Validation | Valid | Determines if various characters enter are valid if not it will ask the user again to input |
| int | n=0 | Increments the while loop goes through |
| Float | Percent | The percentage the user gets if they win the game |
| Bool | Hint2 | Statement to determine if true and allow the output |
| Bool | Correct | Makes sure so that the try catch statements validate the input |
| Usepick | Info | Allows the structure variables to be brought into main |
| String | Str | Brings the direction to main by transferring from the file |
| Char | \*name | Pointer that allows the user to input the name they wanted associated with the game |

**Constructs**

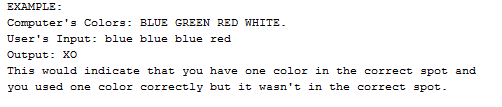
|  |  |
| --- | --- |
| **Chapter** | **Type** |
| 2 | Char |
|  | Int |
|  | Float |
|  | Bool |
|  | Ternary operator |
| 4 | If |
|  | switch |
| 5 | while |
|  | Do-while |
|  | File |
|  | random |
| 6 | Pass by value |
|  | Defaulted parameter |
|  | Returning primitive data types |
|  | Functions |
| 7 | Parallel Arrays |
|  | Arrays |
|  | 2D Arrays |
| 9 | Pointers with arrays (line 39,54,358) |
|  | Allocated memory (line 361) |
| 10 | String objects/classes (lines 50,59,60,297-309,147-150,160,87,89) |
| 11 | Array of structures (lines 59,60,147-150,160,192-195) |
|  | Structures types passed into functions (lines 377,283,269,253,212,153,113,119) |
|  | Structures (line 24) |
|  | Function that return structure object variables (line 377,378,108) |
|  | Inputting/Outputting a structure variable (lines 108,118,119,126,132,133,139,153,212,241,244,253,275) |
| 12 | Reading a binary file (line 87,89) |
|  | Opening a binary file (line 77) |
| 13 | Classes (line 17-21,56,63,65,110,126,129,138,143,144,147-150,192-195,197,200,228,231,238,241,243,250,297-317) |
|  | Passing arguments (line 110,143,144,147-150,192-195,197,200,243,250) |
| 14 | Static members (line 10-12:ComputerPick.cpp) |
|  | Operator overloading (line 242, 23-28:changelimit.cpp) |
| 15 | Inheritance (line 15:UserPick.h, line 17:Results.h) |
| 16 | Templates (line 14-24:Message.h, 297-309) |
|  | Exceptions (line 129, 231,266, 14:Validation.h) |

**Example of input and out**

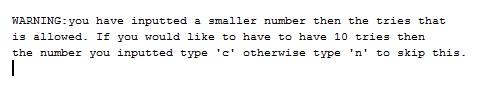
* Start off of the screen and asking the user if they would like an example



* If yes



* If no the program continues, either way the user is then asked their name, number of tries, and displays the colors that the user can choose from
  + If less than 10



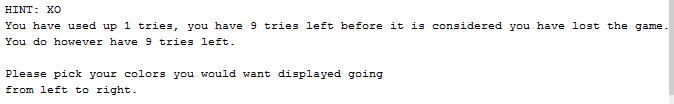
* If over this text is skipped
* Note: If anything invalid is entered then this would show



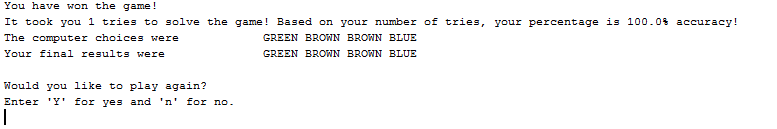
* Once the number of tries are set the user is now asked for the colors they would like to write



* In this example the computer’s colors were RED, GREEN, GREEN, AND ORANGE
* When I enter RED BLUE ORANGE RED as my input this is what came out.



* This indicates that I have one in the correct spot and one color correct but not in the correct spot. It then tells me the number tries I have used up and how many tries I have left. It asks again for the color the user would like to input.
* When all the colors have been guessed correctly



* If yes the game repeats if no the program ends.

**Program**

The program is only from main and includes the class names that were used. To see the actually use of class would be in the actual program.

/\*

\* File: main.cpp

\* Author: Heidy Tamayo

\* Created on May 2, 2016, 3:17 PM

\* Purpose: Playing the game Mastermind

\*/

//System Libraries

#include <iostream> //I/O

#include <string> //string

#include <iomanip> //Formatting

#include <cstdlib> //srand and rand function

#include <fstream> //File I/O

#include <ctime> //Time

//User Libraries

#include "Results.h"//Class that does the comparison of the choices and shows results

#include "Message.h"//Class that shows the mastermind

#include "changelimit.h"//Class that changes the limit if the number of turns has

//ended

#include "Validation.h"//Class that validates the entres from the user

using namespace std;

struct Usepick

{

int limit;//The limit the user inputs

char answer,change;//The response of whether the user would like to

//play again,take a hint, or change their number of

//tries

};

//Global Constants

//Functional Prototypes

void game();//Function that displays mastermind

void Menu();//Function that displays if the user would like see an example

int getN();//Function that takes in the users input of the menu

void example();//Function that displays an example of an input and output

char \* nameuse();//Function to ask the user for the name

Usepick tries();//The number of tries the user would want

char playA(char);//Function to ask the user if they would like play again

//Execution Begins Here

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

//Set the random number seed

srand(time(NULL));

//Declare and initialize variable

ifstream in;//File output

string str;//String that is being used to bring the file to the program

int inN;//The number that is inputted to determine if the user would like to

//see an example

bool reDsply=true;//Used for the switch function to default

char \*name;//Ask for the users name

Usepick info;//Brings the information to the structure

Results pick;//Object to set and bring back from the class

int numTry=10;//The number of tries the user gets before it is considered

//they have lost.

string com[4];//Sets the four colors for the computer

string choice[4];//The 4 colors the user chooses

int n=0;//The increments to indicate the turns allowed before the user loses

float percent;//The intervals of the turns, the percentage of the accuracy

changelimit change;//Allows the user to change the limit of their tries

char ans;//Answer the user inputs to determine if they would like to play again

Validation valid;//Validates the input of the user

bool correct=true;//Used to validate the input of the user

//Do while loop to see if the user would like to play again

do

{

//Function that displays the message

game();

cout<<endl<<endl;

//Try catch function in order to display the instructions from the text

try

{

in.open("Instructions.txt",ios::in|ios::binary);//Opens the file

}

catch (std::ios\_base::failure& e)

{

std::cerr << e.what() << '\n';

}

//Reads the file into the program

if(in.is\_open())

{

while(getline(in,str))

{

cout<<str << endl;

}

}

in.close();//Closes the file

cout<<endl<<endl;

Menu();//Menu asking the user if they would like to see an example

inN=getN();//Input that user puts to know if they would like an example

//Switch statement to determine if the program should show the example

switch(inN)

{

case 1: {example();break; }//Shows the example

default: {reDsply=false;}

}

cout<<endl;

name=nameuse();//Function for user to write their name

cout<<"How many tries would like?"<<endl;

info=tries();//Asking the user how many tries they would like

pick.choice();//Declares the color the user can use

//If statement to warn the user the number of tries they have implemented

if(info.limit<10){

cout<<endl<<endl<<"WARNING:you have inputted a smaller number then ";

cout<<"the tries that"<<endl;

cout<<"is allowed. If you would like to have to have 10 tries then"<<endl;

cout<<"the number you inputted type 'c' otherwise type 'n' to skip this."<<endl;

cin>>info.change;

info.change=toupper(info.change);//Changes the letter choice to uppercase

//Tests the user input if they have inputted a valid response to the question

while(correct)

{

//Try catch statement testing the character inputted

try{

valid.setval(info.change);

correct=false;

}

catch(Validation::Except)

{

cout<<"You have entered an invalid character. Please try again."<<endl;

cin>>info.change;

info.change=toupper(info.change);//Changes the letter choice to uppercase

}

}

//If statement to change the number of tries if the user wishes to

if(valid.getval()=='C'){

info.limit=info.limit>numTry?info.limit:numTry;//Ternary operator

}

}

pick.setlimit(info.limit);//Sets the limit to class

pick.comGen();//Generates the random colors for the computer

//Sets the colors the computer has generated to main

com[0]=pick.getcomp1();

com[1]=pick.getcomp2();

com[2]=pick.getcomp3();

com[3]=pick.getcomp4();

//while loop to generate the number tries for the user

while (n+1<=info.limit)

{

cout<<endl<<"Please pick your colors you would want displayed going"<<endl;

cout<<"from left to right."<<endl;

//Output request the user to enter the 4 colors

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

{

cin>>choice[i];

//If statement to uppercase the users choices

if (choice[i]==choice[0])

{

for (int p=0;p<choice[0].size();p++)

{

choice[0][p]=toupper(choice[0][p]);

}

}

if (choice[i]==choice[1])

{

for (int p=0;p<choice[1].size();p++)

{

choice[1][p]=toupper(choice[1][p]);}

}

if (choice[i]==choice[2])

{

for (int p=0;p<choice[2].size();p++)

{

choice[2][p]=toupper(choice[2][p]);}

}

if (choice[i]==choice[3])

{

for (int p=0;p<choice[3].size();p++)

{

choice[3][p]=toupper(choice[3][p]);}

}

}

//Sets the color choices to the class

pick.setpick1(choice[0]);

pick.setpick2(choice[1]);

pick.setpick3(choice[2]);

pick.setpick4(choice[3]);

pick.setn(n);//Sets the increment in the classes

//Function to determine if the user has won or lost the game

n=pick.wonLost(choice,com,info.limit,n,percent);

//if statement to indicate the user has used up the ten turns they

//to solve the same is over and is now considered they lost

if (n==9)

{

//Output of results after all tries have been used up

cout<<"You have used all your tries and have lost the game."<<endl<<endl;

}

n++;//increments the turns

//If statement to allow the user to have an extra turn if they run out

if(n==info.limit)

{

cout<<"You have lost the game however you can increment your play"<<endl;

cout<<"by 1 if you would like to keep playing. Type Y for yes or"<<endl;

cout<<"any other character to not take a extra turn."<<endl;

cin>>ans;

ans=toupper(ans);

//Resets the variable in order to test again

correct=true;

//Tests the user input if they have inputted a valid response to the question

while(correct)

{

try

{

valid.setval(ans);

correct=false;

}

catch(Validation::Except)

{

cout<<"You have an invalid character. Please try again."<<endl;

cin>>ans;

ans=toupper(ans);//Changes the letter choice to uppercase

}

}

if(valid.getval()=='Y')

{

//To determine if the limit should be changed

change.setlimit2(info.limit);

++change;

pick.setlimit(change.getlimit2());

info.limit=change.getlimit2();

}

}

}

//Function that determine the results of the user

pick.results(com,choice);

//Function that determines if the user would like to play again

info.answer=playA(info.answer);

//Resets the variable in order to test again

correct=true;

//Tests the user input if they have inputted a valid response to the question

while(correct)

{

try

{

valid.setval(info.answer);

correct=false;

}

catch(Validation::Except)

{

cout<<"You have an invalid character. Please try again."<<endl;

cin>>info.answer;

info.answer=toupper(info.answer);//Changes the letter choice to uppercase

}

}

//If statement if the user decides to play again.

if(valid.getval()=='Y'){

cout<<"You have chosen '"<<info.answer<<"' as your answer therefore ";

cout<<"the game will repeat."<<endl;

}

//Resets the variable in order to test again

correct=true;

n=1;//Resets the increment of tries

info.limit=0;//Resets the limit

}while(valid.getval()=='Y');

delete []name;//Deletes the allocate memory of the user name

//Exit stage right

return 0;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Gives the name of the game using a template

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

void game()

{

Message<string>word("# #");

Message<string>word1("## ## ## #### ##### ###### ##### # # # # "

" # #####");

Message<string>word2("# # # # # # # # # # # ## ## # ## "

" # # #");

Message<string>word3("# # # # # #### # ##### # # # ## # # # #"

" # # #");

Message<string>word4("# # ###### # # # ##### # # # # "

"# # # #");

Message<string>word5("# # # # # # # # # # # # # # "

" ## # #");

Message<string>word6("# # # # #### # ###### # # # # # # "

" # #####");

cout<<word.printMes()<<endl;

cout<<word1.printMes()<<endl;

cout<<word2.printMes()<<endl;

cout<<word3.printMes()<<endl;

cout<<word4.printMes()<<endl;

cout<<word5.printMes()<<endl;

cout<<word6.printMes()<<endl;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Menu Function to determine if the user would like to see an example

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

void Menu()

{

cout<<"Type 1 to see an example input and output."<<endl;

cout<<"Type any other number to continue playing the game."<<endl;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Integer response to the Menu

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

int getN()

{

int inN;

cin>>inN;

return inN;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Gives an example to user of an input and an output

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

void example()

{

cout<<"EXAMPLE:"<<endl;

cout<<"Computer's Colors: BLUE GREEN RED WHITE."<<endl;

cout<<"User's Input: blue blue blue red "<<endl;

cout<<"Output: XO"<<endl;

cout<<"This would indicate that you have one color in the correct spot and"<<endl;

cout<<"you used one color correctly but it wasn't in the correct spot."<<endl;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Allows the user to enter the name to associate with the game

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

char \*nameuse()

{

int SIZE=99;//Creates the size the user can enter

char \*name = new char[SIZE];//allocates memory for what the user inputs

//Asks for the users name

cout<<"Please enter the name you would like associated with the game played."<<endl;

cin.ignore();

cin.getline(name, SIZE);

return name;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Allows the user to enter the number of tries

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

Usepick tries()

{

Usepick info;

cin>>info.limit;

return info;

}

//000000011111111112222222222333333333344444444445555555555666666666677777777778

//345678901234567890123456789012345678901234567890123456789012345678901234567890

// Asks the user if they would like to play again

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

char playA(char answer)

{

cout<<endl<<"Would you like to play again?"<<endl;

cin>>answer;

cout<<endl;

answer=toupper(answer);

return answer;

}