**Project 1**

**Game Title:**

Mastermind Game

**Project Includes:**

Versions 1 and 2 of Development

**Course:**

CSC 7 Spring 2018

**Section:**

42486

**Date Due:**

April 25, 2018

**By:**

Javier Ventura

**Introduction:**

For Project 1 we were assigned to make a replica of the fun logic Mastermind. This game as played on an application or website consists of certain amount of spots in a row where each can contain typically a different color. If a color is in the right, some sort of indicator tells you that a spot is correct, not saying which spot of course. If a color is right but not in the right spot then an indicator for that is given, also not giving away which color it is. You must use these clues to figure out what combination is the correct one. This is a really good game to analyze logic and test our understanding of concepts from class. Logic gates and inference of previous statements is what is always underlying in any proof or problem. In other words being a good problem solver is working with what one knows and does not know. Separating those and working towards having unknowns equal to none is really the general idea. Therefore, this game serves as great practice for such and sharpens your logic and cognitive skills. Therefore, it was a good challenge and fun playing as there are variations where some allow duplicates, allow blanks and only allow a certain amount of tries of a certain color per round, as well as having more spots in a row.

**Description:**

My game uses 4 spots which is done with an array. That is filled with integers 1-8 each representing where the 8 colors, typically used in websites, would be.Then the player is given 4 variables to be compared with the contents of the array. If statements first check if the variable matching with that spot is correct, if it is then a message is given that a try is right, if not then it checks if that variable matches any of the other contents in the array, if that is true then a message tells player that a guess is in the wrong spot but is right. The same is done with all four guesses since no blanks are allowed but duplicates are. I give 10 tries to solve the combination if not then the player is told they lost and the menu screen is shown on screen again. If they do solve all four then the corresponding booleans must all be true and they will be told that they have won. The menu is done with a switch statement and the other options include seeing the rules and quitting the game. Another option is seeing a bisection method beat the game which is not required but is a start to building an AI for the next project. Version 1 is just getting the game running but version 2 includes the bisection method solving the game which can be expanded to bigger ranges very easily.

**PseudoCode:**

Include libraries: iostream, cstdlib,ctime

Use namespace standard

Use integer main function

Set random number seed

Declare variables used throughout menu like menu choice and boolean quit

Use do while loop, while quit is false

Output menu and options

User enters choice

Use while loop to make sure choice is a valid option

Use switch statement to match choice with appropriate option

Case 1 is player trying to beat game

Declare array with integers 1-8 and players tries, booleans,tries

Use do while, while tries is greater than 0

Player enters four guesses

Check if first try is correct,

if so output message

And set boolean one to true

If not check if try matches any other one,

if so output message

Set one=false

Check if second try is correct,

if so output message

And set boolean two to true

If not check if try matches any other,

if so output message

Set two to false

Check if third try is correct,

if so output message

And set boolean three to true

If not check if try matches any other,

if so output message

Set three to false

Check if last try is correct

if so output message

And set boolean four to true

If not check if try matches any other,

If so output message

Set four equal to false

Check if all booleans are true

If so output win message

And break out do while

If not all booleans are true

Subtract 1 from tries

Output remaining tries

Check if number of tries is less than or equal to 0

` if not then output lose message

Show answer

And break out do while

Whether player wins or loses initial break will only exit out do while but not case 1, so break again here to not go to case 2

Case 2 is bisection method

Declare array with integers 1-8 and ai tries,

Booleans,guesses,4 minimums and 4 maximums

Minimums start at 1 and maximums at 8

Set guesses to minimum plus maximum divided by 2

Use do while, while tries is greater than 0

Ai enters four guesses

Output guess

Check if first try is correct,

If so output guess is right

If not check if guess is less than answer,

Minimum for this guess range is the guess

If not check if guess is greater than answer,

Maximum for this guess range is the guess

Check if second try is correct,

If so output guess is right

If not check if guess is less than answer,

Minimum for this guess range is the guess

If not check if guess is greater than answer,

Maximum for this guess range is the guess

Check if third try is correct,

If so output guess is right

If not check if guess is less than answer,

Minimum for this guess range is the guess

If not check if guess is greater than answer,

Maximum for this guess range is the guess

Check if fourth try is correct,

If so output guess is right

If not check if guess is less than answer,

Minimum for this guess range is the guess

If not check if guess is greater than answer,

Maximum for this guess range is the guess

Check if all booleans are true

If so output win mesage

And break do while

If not

Tries goes down by 1

Check if tries is greater than 0

If not output lose message

Break out of do while

Whether player wins or loses initial break will only exit out do while but not case 2, so break again here to not go to case 3

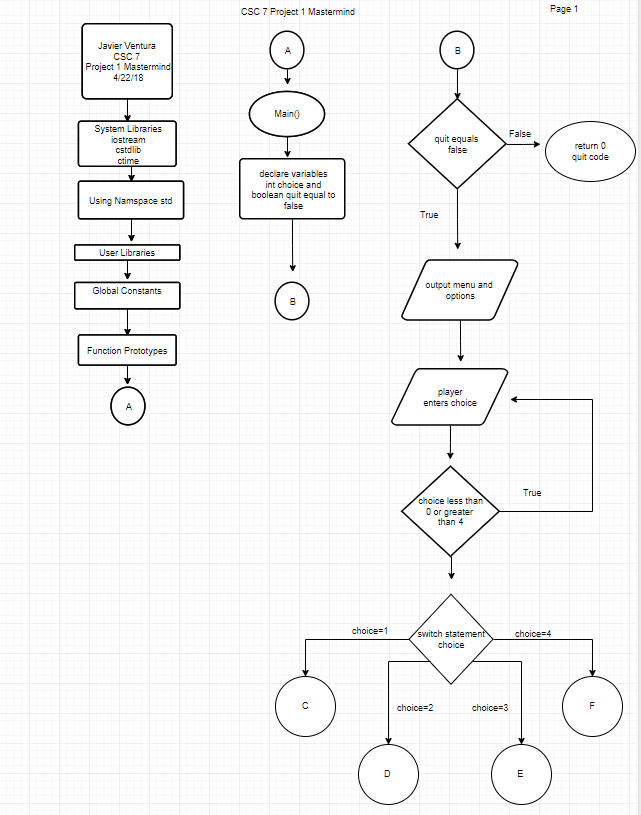
Case 3 is rules

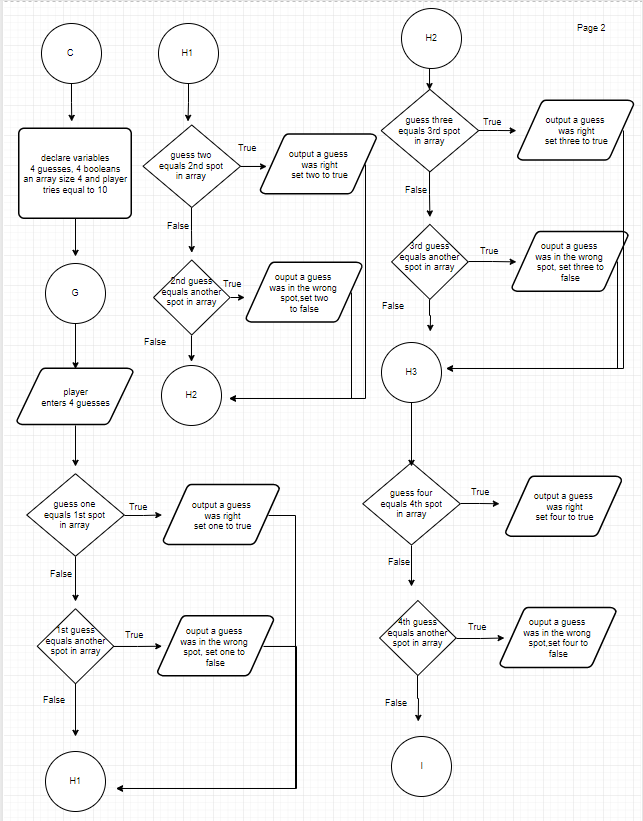
Simply output rules and break after message back to menu

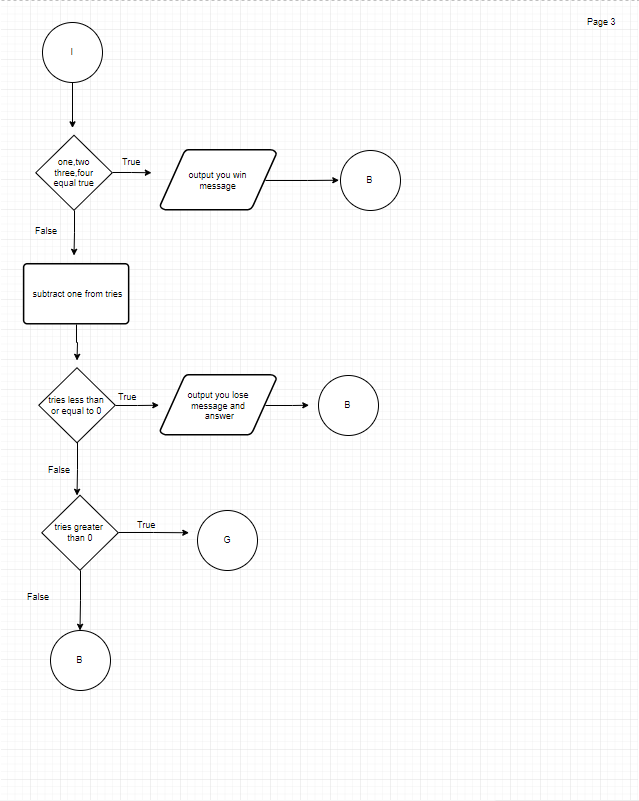
Case 4 is quitting game

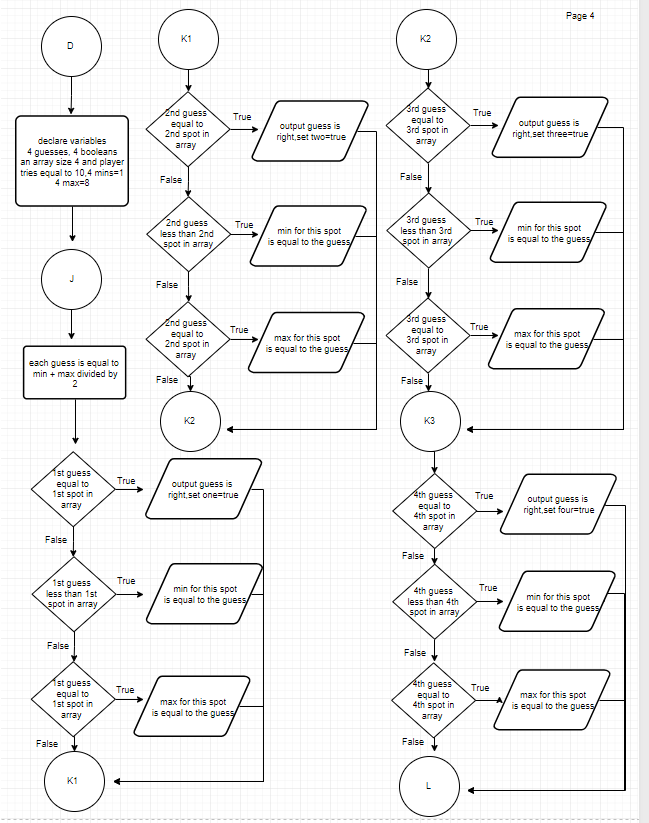
Output quitting game set bool quit equal to true and return 0 to quit code

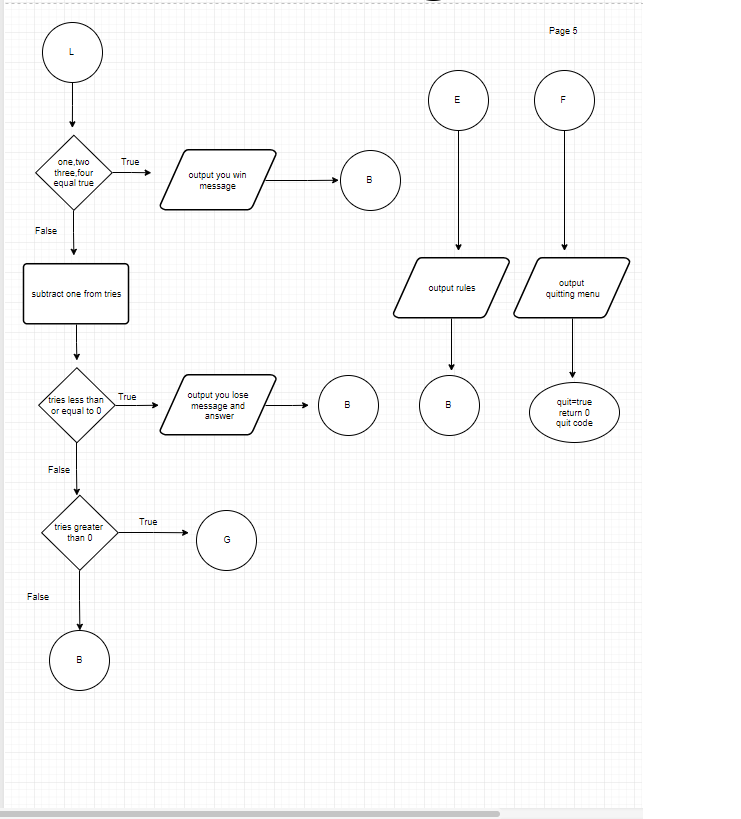
**Flowcharts:**











**Source Code:**

/\*

\* File: main.cpp

\* Author: Javier Ventura

\* Purpose: CSC 7 Project 1 MasterMind Game

\* Created on April 6, 2018, 12:42 PM

\*/

#include <cstdlib>

#include <iostream>

#include <ctime>

using namespace std;

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

//Set Random Number Seed

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

//Declare Variables

int choice; //MEnu choice

bool quit=false; //Quit is false so menu keeps running

do{

cout<<"MasterMind"<<endl;

cout<<"----------"<<endl;

cout<<"(1)Play Game"<<endl;

cout<<"(2)Watch AI Play"<<endl;

cout<<"(3)Quick Rules"<<endl;

cout<<"(4)Quit Game"<<endl;

cout<<"Enter Choice: ";

cin>>choice;

//Use while loop to make sure choice is a valid option

while(choice<1 || choice>4){

cout<<"Not a Valid Menu Option, Try Again: ";

cin>>choice;

}

switch(choice){

case 1:{

//Declare Local Variable

int a[4]; //Array Holding Key, Each element will be a spot

int x1; //1st Spot, Player Enters

int x2; //2nd Spot, Player Enters

int x3; //3rd Spot, Player Enters

int x4; //4th Spot, Player Enters

bool one=false,two=false,three=false,four=false; //Marks when Player Guesses All 4

int tries=10; //Number of Tries, Player gets

//Fill Array with the Contents 1-8

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

a[i]=rand()%8+1; //1-8 is equivalent to 8 colors

}

do{

cout<<endl;

cout<<"4 Spots, 1-8 is Possible in Each!"<<endl;

cout<<"Enter Your 4 Tries: "<<endl; //Player will enter tries

cin>>x1;

cin>>x2;

cin>>x3;

cin>>x4;

cout<<endl;

if(x1==a[0]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

one=true; //Set flag to true

}

else if(x1==a[1] || x1==a[2] || x1==a[3]){

cout<<"One of Your Guesses is in the Wrong Spot"<<endl;

one=false; //Set flag to false so if previously right it wont stay right

}

if(x2==a[1]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

two=true; //Set flag to true

}

else if(x2==a[0] || x2==a[2] || x2==a[3]){

cout<<"One of Your Guesses is in the Wrong Spot"<<endl;

two=false; //Set flag false so if previously true wont stay right

}

if(x3==a[2]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

three=true; //Set Flag to true

}

else if(x3==a[0] || x3==a[1] || x4==a[3]){

cout<<"One of Your Guesses is in the Wrong Spot"<<endl;

three=false; //Set flag to false to if previously right wont stay right

}

if(x4==a[3]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

four=true; //Set flag to true

}

else if(x4==a[0] || x4==a[1] || x4==a[2]){

cout<<"One of Your Guesses is in the Wrong Spot"<<endl;

four=false; //Set to false so if previously right it wont stay right

}

cout<<endl;

//If all flags are set to true then player wins

if(one==true && two==true && three==true && four==true){

cout<<"You Won!"<<endl;

cout<<endl;

break;

}

tries--; //Tries goes down

cout<<"Tries Left: "<<tries<<endl; //Output tries left

//Break out of do while if tries is less than 0

if(tries<=0){

cout<<endl;

cout<<"You Lost Sorry! Try Again"<<endl;

cout<<"Answer: "<<endl;

cout<<a[0]<<" "<<a[1]<<" "<<a[2]<<" "<<a[3]<<endl;

cout<<endl;

break;

}

}while(tries>0);

break; //First break breaks to case 1 this breaks back to menu

}

case 2:{

cout<<endl;

cout<<"AI Coming Soon.... :]"<<endl;

cout<<endl;

cout<<"For Now,Watch the Bisection Method: "<<endl;

//Declare Local Variables

int a[4]; //Array Holding Key, Each element will be a spot

int x1; //1st Spot, Player Enters

int x2; //2nd Spot, Player Enters

int x3; //3rd Spot, Player Enters

int x4; //4th Spot, Player Enters

int tries=10; //Number of Tries

int min1=1; //Initial Minimum Guess for first spot

int max1=8; //Initial Max Guess for 1st spot

int min2=1; //Needed since range for guess this guess is different than others

int max2=8; //Needed since range for guess this guess is different than others

int min3=1; //Needed since range for guess this guess is different than others

int max3=8; //Needed since range for guess this guess is different than 1

int min4=1; //Needed since range for guess this guess is different than 1

int max4=8; //Needed since range for guess this guess is different than 1

bool one=false,two=false,three=false,four=false; //Flags to set when guesses are right

//Fill Array with its contents integers 1-8 each

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

a[i]=rand()%8+1;

}

while(tries>0){

cout<<endl;

cout<<"4 Spots, 1-8 is Possible in Each!"<<endl;

cout<<"Enter Your 4 Tries: "<<endl;

//Make Guesses between the current max and min;

x1=(max1+min1)/2;

x2=(max2+min2)/2;

x3=(max3+min3)/2;

x4=(max4+min4)/2;

cout<<x1<<" "<<x2<<" "<<x3<<" "<<x4<<endl;

cout<<endl;

if(x1==a[0]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

one=true;

}

else if(x1<a[0]){

min1=x1;

}

else if(x1>a[0]){

max1=x1;

}

if(x2==a[1]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

two=true;

}

else if(x2<a[1]){

min2=x2;

}

else if(x2>a[1]){

max2=x2;

}

if(x3==a[2]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

three=true;

}

else if(x3<a[2]){

min3=x3;

}

else if(x3>a[2]){

max3=x3;

}

if(x4==a[3]){

cout<<"One of Your Guesses is in the Right Spot"<<endl;

four=true;

}

else if(x4<a[3]){

min4=x4;

}

else if(x4>a[3]){

max4=x4;

}

cout<<endl;

if(one==true && two==true && three==true && four==true){

cout<<"You Won!"<<endl;

cout<<endl;

break;

}

else{

tries--; //Didnt Win so Tries goes Down

cout<<"Tries Left: "<<tries<<endl;

}

if(tries<=0){

cout<<endl;

cout<<"You Lost Sorry! Try Again"<<endl;

cout<<endl;

break;

}

};

break;

}

case 3:{

cout<<endl;

cout<<"Mastermind Game Rules:"<<endl;

cout<<"There is 4 Spots to fill"<<endl;

cout<<"Each spot can be an integer 1-8"<<endl;

cout<<"Duplicates are allowed"<<endl;

cout<<"Each round is given 10 Tries"<<endl;

cout<<"Blanks are not allowed"<<endl;

cout<<"Wrong Spot means a number you guessed is right but in the wrong spot"<<endl;

cout<<"Right Spot means a guess is right and in the right spot"<<endl;

cout<<"Good Luck!"<<endl;

}

case 4:{

cout<<endl;

cout<<"Exiting Game... :("<<endl;

cout<<endl;

return 0;

}

default:{

cout<<endl;

cout<<"Not a Valid Option! Sorry, Try Again."<<endl;

cout<<endl;

}

}

}while(quit==false);

return 0;

}