Project 2

Tittle
Master Mind (AI Ver)

Course
CSC-7

Section
42645

Due Date
June 9, 2017

Author
Sili Guo

I. Introduction

Master Mind is a tradition American game which has a high requirement of logic thinking and basic knowledge for algorithm. During the game, player are asked to guess the color and position of a set of balls that listed in random order. There will be hints for the correct color numbers and correct color and position numbers each turn; and players can use this hint to make further guess.

During each turn, players should use logic and the previous result to eliminate the wrong color and approach the range of correct colors. Sometimes, it is a good strategy to first get all colors correct, then consider about the positions.

Also, when guessing each time, trying to put balls in different position can help you to shorten the steps you need to find out the correct position of each ball.

Players who guess all balls in correct color and correct positions will win the game. Some Master Mind game has a restriction of turn numbers to get correct, others just record the turns players have spent.

II. Game Rules

A. Basic Rules

1.Start a new game.

2.Choose game mode.

3.Make guesses and look at hints.

4.Check turn numbers you have used to solve the game.

B. Game Mode

1.The number of balls.

- 4 balls
- 6 balls
- 8 balls

2.Duplicated

- Allow duplicated
- Don’t allow duplicated

C. Win

1. Who has all positions correct win the game.

2.)If you want to compete with friends, compare the turn numbers both used to win the game!

D. AI Version

1. In the newest update of AI Version, player can pick a random number from 0000~9999, and ask computer to break the code.

2.The AI based on finding the best condition to break the code.

III. Summary

Total Line of Code	250
Comment Line	-
Variable	12
Function	3

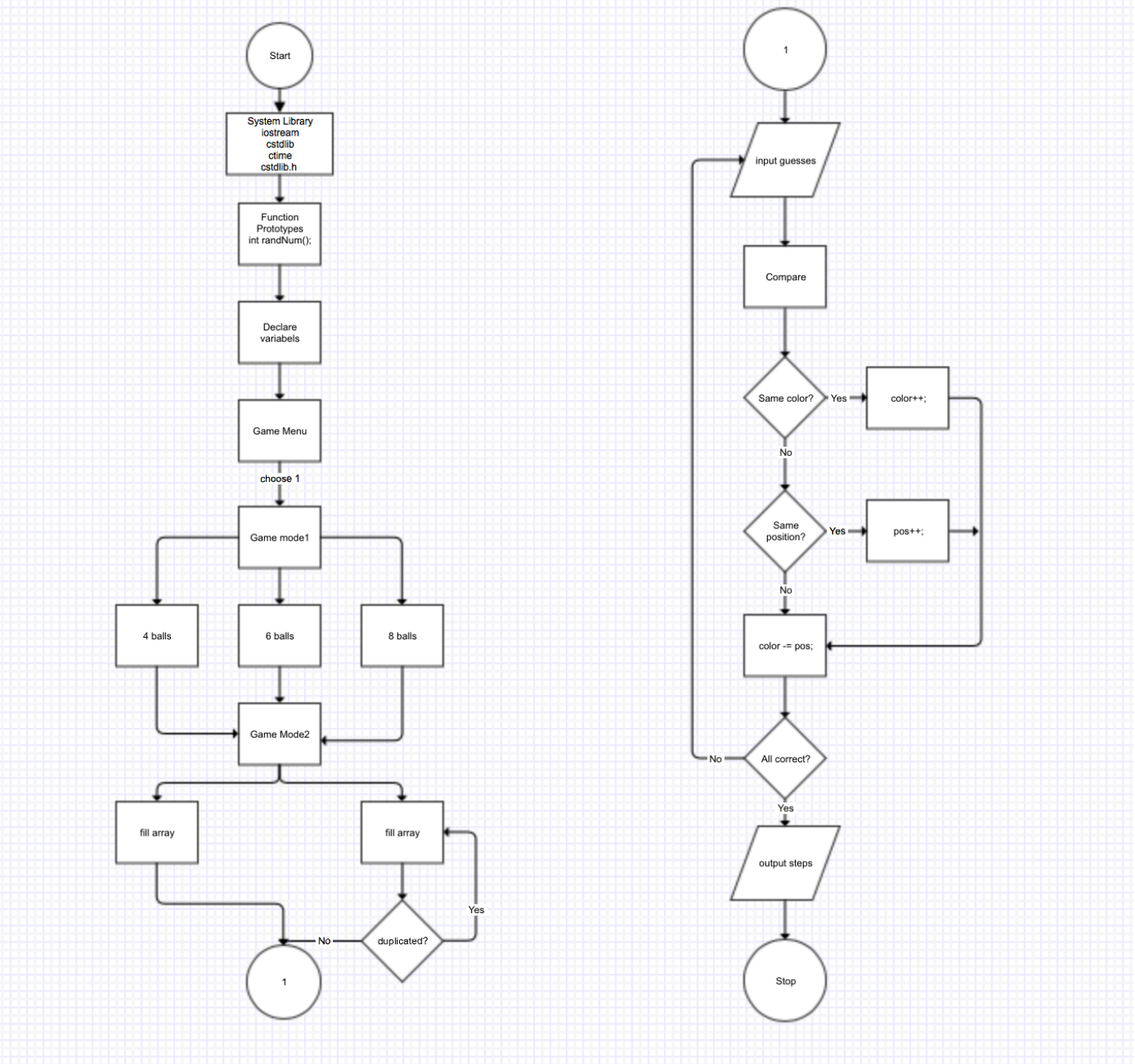
Basically, this Master Mind game allows player to have choices of how many balls for guessing and if duplicated is allowed. The program

is build on the combination of switch case and loop. I used dynamic array to utilize the change between the number of balls. To make the process fluent, I use the windows clear command to refresh the new pages.

In the newest AI Version, I create an AI to break the code that player gives. It is kind of hard to generalize all condition in the game, and is easily to miss some condition. So I searched online and see a basic way to generalize an AI is to find the best solution in each turn. So I initialized a system called score. All the judgement inside the code is based on this system. If the condition is considered as better by computer, it will keep the charges; otherwise, it will come back to original condition, and this helps me a lot to solve this problem.

IV. Description

A. Flowcharts



Main:

B. Process

First, choose game menu

```
   Mastermind Game (AI-Version)

   Game Menu:
   1. Start game
   2. Read rules
   0. Quit game
Please enter your choice (0~2):
```

Suggest read game rules before playing game.

```
 ************ Mastermind (AI-Version) Rules ***********************************
• There are 10 balls (0~9) in total, and 4 of them are chosen.
• Player randomly choose 4 numbers from 0~9, allowing duplicates.
• In each turn, there will be a hint for current condition:
   # for correct balls in wrong positions
   # for correct balls in correct positions
• Computer will based on these two conditions to guess the balls.

 ************ Mastermind (AI-Version) Rules ***********************************
```

Start game.

Ask User to input 4 numbers (For example, 8 3 8 9).

```
Please enter 4 numbers from 0 to 9! (with spaces)
For example: 0 1 2 3
(Warning: Don't press space after last digit, or it will jump back to game menu!!!)
8 3 8 9
```

Computer will break the code for you.

```
Guess # 1:  0 0 1 1 ----- [ 0 ; 0 ]
Guess # 2:  2 2 2 2 ----- [ 0 ; 0 ]
Guess # 3:  3 3 3 3 ----- [ 0 ; 1 ]
Guess # 4:  3 4 4 4 ----- [ 1 ; 0 ]
Guess # 5:  5 3 5 5 ----- [ 0 ; 1 ]
Guess # 6:  6 3 6 6 ----- [ 0 ; 1 ]
Guess # 7:  7 3 7 7 ----- [ 0 ; 1 ]
Guess # 8:  8 3 8 8 ----- [ 0 ; 3 ]
Guess # 9:  8 3 8 9 ----- [ 0 ; 4 ]

I won the game, in 9 turns!
Your code was:
[8 3 8 9]
```

press 0 to quit the game

```
You quit the game.
```

VI. Code

```
/*
 * File:   main.cpp
 * Author: Sili Guo
 * Created on June 4, 2017
 * Purpose: Mastermind Game AI
 * Comment: The AI judge based on "better condition",
 *          which means that it changes when find a better condition,
 *          keep original condition if the new condition is not better.
 *          The judgment of condition based on the system of "score" I declared.
 *          There are 13 levels of scores that are used for judgment.
 */

//System Libraries
#include <iostream>
#include <stdlib.h>
using namespace std;

//User Libraries

//Global Constants
```

```
//Function Prototypes
void check(int [], int [], int []);
int score(int, int);
void base10counter(int []);

//Execution Begins here!

int main(int argc, char** argv) {
    //Random number seed
    srand(static_cast<unsigned int> (time(0)));
    //Constants
    const int TOTAL = 10;
    const int SELCT = 4;
    //Declare Variables
    int input[4]; //Four balls input by user
    int guess[4]; //Computer's output guess
    int AI_guess[4]; //Computer's guess inside code
    int result[2] = {0, 0}; //{correct balls, correct positions}
    int previous_answers[10][4]; //Record previous answers
    int previous_scores[10]; //Record previous scores
    int gameScore; //The score of computer's final guess
```

```
int testScore; //The score used to compare with previous score, deciding for better condition
int start; //Game menu choice
int turn; //Game turns
char enter; //Clear screen signal
bool same; //Check duplicates
do {
    system("clear");
    //Initialize variables each turn
    guess[0] = 0;
    guess[1] = 0;
    guess[2] = 1;
    guess[3] = 1;
    AI_guess[0] = 0;
    AI_guess[1] = 0;
    AI_guess[2] = 1;
    AI_guess[3] = 1;
    turn = 0;
    //Output game menu
    cout << endl << "   Mastermind Game (AI-Version)" << endl << endl;
    cout << "   Game Menu:" << endl;
    cout << "   1. Start game" << endl;
    cout << "   2. Read rules" << endl;
```

```
        cout << "   0. Quit game" << endl;
        cout << "Please enter your choice (0~2): ";
        cin >> start;
        //If choose quitting game
        if (start != 1 && start != 2) {
            system("clear");
            cout << "You quit the game." << endl;
            break;
        }
        switch (start) {
            case 1://Start game
                system("clear");
                //Promote 4 digits from user
                cout << "\n Please enter 4 numbers from 0 to 9! (with spaces)" << endl;
                cout << " For example: 0 1 2 3" << endl;
                cout << " (Warning: Don't press space after last digit, or it will jump back to game
menu!!!)" << endl;
                do {
                    cout << " ";
                    cin >> input[0] >> input[1] >> input[2] >> input[3];
                    //Check for validation
                    if (input[0] < 0 || input[0] > 9 || input[1] < 0 || input[1] > 9
```

```
            || input[2] < 0 || input[2] > 9 || input[3] < 0 || input[3] > 9) {
        cout << " Invalid input! Please enter in correct form! (eg. 0 1 2 3): " << endl;
    }
} while (input[0] < 0 || input[0] > 9 || input[1] < 0 || input[1] > 9
        || input[2] < 0 || input[2] > 9 || input[3] < 0 || input[3] > 9);
//Initialize turns to 15
while (turn < 15) {
    turn++;
    //Output computer's guess
    cout << " Guess #";
    if (turn < 10) cout << " ";
    cout << turn << ": ";
    cout << " " << guess[0] << " " << guess[1] << " " << guess[2] << " " << guess[3];
    //Check if the guess was correct
    check(guess, result, input);
    //Calculate game score
    gameScore = score(result[0], result[1]);
    //Record previous answers and scores
    previous_answers[turn - 1][0] = guess[0];
    previous_answers[turn - 1][1] = guess[1];
    previous_answers[turn - 1][2] = guess[2];
    previous_answers[turn - 1][3] = guess[3];
```

```
previous_scores[turn - 1] = gameScore;
//If answers is correct, break
if (result[1] == 4)
    break;
//Deciding for best condition
bool anySolutions = true;
while (anySolutions) {
    //Marking for same condition
    bool consistent = true;
    //Find a better condition than last time
    for (int i = 0; i < turn; i++) {
        check(AI_guess, result, previous_answers[i]);
        testScore = score(result[0], result[1]);
        if (testScore != previous_scores[i]) {
            consistent = false;
            break;
        }
    }
    //If no better condition, break
    if (consistent) break;
    base10counter(AI_guess);
    //If not win, test if no solution
```

```
            if (testScore != 13) {
                if (AI_guess[0] == 0 && AI_guess[1] == 0 && AI_guess[2] == 1 &&
AI_guess[3] == 1) {
                    anySolutions = false;
                    printf(" ERROR: No solution possible!\n");
                }
            }
        }
        //Use the best condition as computer's next guess
        guess[0] = AI_guess[0];
        guess[1] = AI_guess[1];
        guess[2] = AI_guess[2];
        guess[3] = AI_guess[3];
        //Output result of computer's guess
        cout << " ----- [ " << result[0] << " ; " << result[1] << " ]" << endl;
    }
    //Output last result of computer's guess
    cout << " ----- [ " << result[0] << " ; " << result[1] << " ]" << endl;
    //If win the game, output result
    if (result[1] == 4) {
        cout << "\n I won the game, in " << turn << " turns!" << endl;
        cout << " Your code was: " << endl;
```

```
                cout << " [" << guess[0] << " " << guess[1] << " " << guess[2] << " " << guess[3]
<< "]" << endl;
            } else//If did not get solution
                cout << "\n Sorry, I did not get the solution!" << endl;
            break;
        case 2://Read rules
            system("clear");
            //Output rules and menu
            cout << "\n ********** Mastermind (AI-Version) Rules
**********************************" << endl;
            cout << "• There are 10 balls (0~9) in total, and 4 of them are chosen." << endl;
            cout << "• Player randomly choose 4 numbers from 0~9, allowing duplicates." <<
endl;
            cout << "• In each turn, there will be a hint for current condition: " << endl;
            cout << "   # for correct balls in wrong positions" << endl;
            cout << "   # for correct balls in correct positions" << endl;
            cout << "• Computer will based on these two conditions to guess the balls." << endl;
            cout << "\n ********** Mastermind (AI-Version) Rules
**********************************" << endl;
            break;
        default://Quit game
            break;
```

```
        }//End of switch case
        cout << "\n Press enter to go back to Game Menu." << endl;
        cin.ignore();
        do {
            enter = getchar();
        } while (enter != '\n');
    } while (enter == '\n');

    //Exit Stage right!
    return 0;
}

void check(int guess[], int result[], int input[]) {
    //Initialize variables
    result[0] = 0;
    result[1] = 0;
    //copy guess and input for comparing and making changes
    int copy1[4] = {0, 0, 0, 0};
    int copy2[4] = {0, 0, 0, 0};
    //Copy select  and guess array to enable for edit
    for (int i = 0; i < 4; i++)
        copy1[i] = input[i];
```

```
for (int i = 0; i < 4; i++)
    copy2[i] = guess[i];

//First check correct position
for (int i = 0; i < 4; i++) {
    if (copy2[i] == copy1[i]) {
        copy1[i] = -1;
        copy2[i] = -2;
        result[1]++;
    }
}
//Then check for correct color
for (int i = 0; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        if (j != i) {
            if (copy2[i] == copy1[j]) {
                result[0]++;
                copy1[j] = -1;
                break;
            }
        }
    }
```

```
    }
}

int score(int num, int pos) {
    //Initialize scores with different condition
    if (num == 0 && pos == 0) return 0;
    if (num == 1 && pos == 0) return 1;
    if (num == 0 && pos == 1) return 2;
    if (num == 2 && pos == 0) return 3;
    if (num == 1 && pos == 1) return 4;
    if (num == 0 && pos == 2) return 5;
    if (num == 3 && pos == 0) return 6;
    if (num == 2 && pos == 1) return 7;
    if (num == 1 && pos == 2) return 8;
    if (num == 0 && pos == 3) return 9;
    if (num == 4 && pos == 0) return 10;
    if (num == 3 && pos == 1) return 11;
    if (num == 2 && pos == 2) return 12;
    if (num == 0 && pos == 4) return 13;
}

void base10counter(int num[4]) {
```

```
    //Count for bounder
    num[3]++;
    if (num[3] == 10) {
        num[2]++;
        if (num[2] == 10) {
            num[1]++;
            if (num[1] == 10) {
                num[0]++;
                if (num[0] == 10) num[0] = 0;
                num[1] = 0;
            }
            num[2] = 0;
        }
        num[3] = 0;
    }
}
```