Mastermind

Project1

CSC- 5 – 46023 Intro C++

Tsz,Kwan

10 – July – 2014

Content

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

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

3. Research…………………………………………..…..5

Arrays

Parallel array

Shuffle with array  
String.length

4. Variables list…………………………….………….…6

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

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

7. Flowchart

8. Code

1. Introduction

**Rules and Gameplay**

This program is like the original mastermind. The original mastermind is let the player guess certain time of the colors in 4slots. Each time after the player guess, it will tell the player how many colors are at the right place, and how many colors are the right colors excluded the right place. I change the colors to numbers (0-9). Also, there are two levels which are easy and hard let the player to choose. The easy level doesn’t generate 4digits with same numbers such as 1234 and 9876. However, the hard level generates 4digits which contains at least 2 same numbers such as 1314 and 9335. The player has 10 times to guess to number. If they can’t guess the right number in 10 times, the program will end and announce the player lose and the correct number.

**Thoughts after Program**

The game was simple. The main ability the player need is logical thinking because the next guess is based on the result of the previous guesses. The player needs to use the previous results and cross comparison to inference the right digits.

It really makes me happy when the program runs successfully. However, there are some bugs if I use char to store the answer the player enters. If the player enters 100 instead of 0100, the program still separates the digits and runs. 100 should be invalid input because it is only three digits. I use string to store the guess and check the validation with string size and use ascii code to check whether is number or not.

2. Development

Approach Strategy

Before I chose mastermind as my project. My idea is Black Jack. I try to use rand to generate numbers between 1 and 52 four times, and figure that the number may be the same. It is easy to check the repeated number and generate again with arrays. After I fixed this error, I know that Black Jack is too difficult if there is a robot as a house, and it will be very boring fi players can see others’ cards on the screen. Therefore, I change my topic to mastermind. Because I know how to use arrays and I write the program to check repeated numbers, it is easy to write a very simple mastermind.

The way to check whether the digits are at the right place or right digits is parallel arrays. The program checks the right place first. Use real[i]==garr[i] to check (real[i] is the generate digit and garr[i] is guess digit). If they are match, the program will replace real[i] to 10 and the bool[i] to true to avoid it matches again, store that memory location, and count. After checking the place, the program will check the numbers no matter the place is. If the numbers are match and bool[i] is false, the program will replace real[i] to 10 and count. After finish the comparison, it counts the number of guesses, displays how many digits at the right place and how many digits are correct, and recover real[i] with the original generated number array realO[i].

I use int to store guess number at the beginning, but I figure out it gives me errors if I input some non-numbers. Therefore, I use string instead of int and check it with ascii code (add 48). After checking, I use array copy the value from char to int with subtract 48.

The comparison is really complicated and need to check everything step by step. Because the array do the command at that memory location first, then change to other location, I need to use a parallel arrays to avoid the error.

After I finished my program, I noticed that it is very easy to use the correct number to check the guess number than use guess number to check the correct number.

3. Research

1. Arrays  
   It is better to use array in this project because the program need to compare each digit of place and number every time the player enter the guess number. It is possible to use if and else-if to do this program, but it uses much more lines than array does. Also, it is hard to expand it to let player to decided how many digits of the generate number without arrays.
2. Parallel arrays  
   In this project, I use array to store the digits. I need a parallel array to compare the same location value (garr[0] compare to real[0]), after that compare to other location value (garr[0] compare to real[1]).
3. Shuffle with array  
   In the hard version, I want to generate a 4 digit number with at least 2 same number. I copy the number is realO[0] to realO[3]. If I don’t do the shuffle, the first and the last digit will be the same every time. Therefore, I create an array index[4] with value 0 to 3, and generate a number with a range 0 to 3 in the for loop to rearrange the location of the index[4]. Then put index[i] into realO[i] which is realO[index[i] to avoid the first and last digit will be the same everytime.
4. String.length

If I use char array as a input. I can’t check the length because the program will wait if the player enters a 3digit number. If the player enters a 5 digit number, the last digit will store as the next input. String.length can help me to check the length and let the player enter any digits.

4. Variables list

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Variable Name** | **Description** | **Line** |
| int | count |  | 31 |
|  | realO[4] | the generated number save | 32 |
|  | real[4] | the generated number compare to guess | 33 |
|  | X | right place counter | 35 |
|  | O | right number counter | 35 |
|  | garr[4] | guess array after check validation | 159 |
| char | choice | choose which level | 29 |
|  | display | choose whether display the correct number or not | 30, 145 |
| string | sg | guess number input to check validation | 157 |
| bool | repeat | check whether generated digits are the same | 34 |
|  | valid | check sg validation | 158 |
|  | same[4] | to remember the place when the digit is at the right and avoid to count with int O | 160 |
| time\_t | tstart | record start time | 230 |
|  | tend | record end time | 230 |
| ofstream | output |  | 36 |

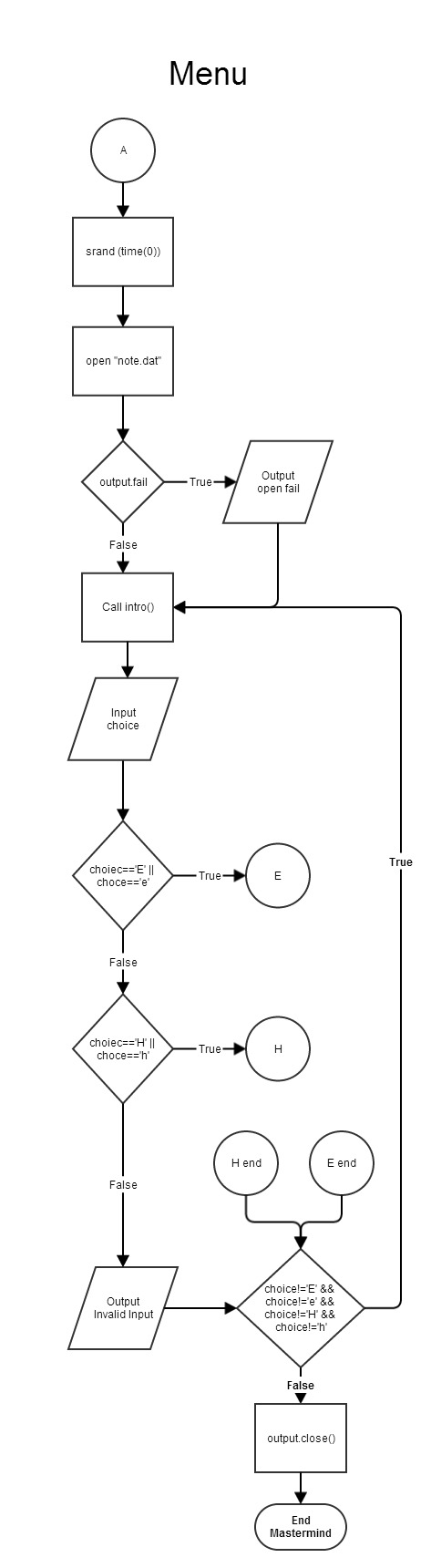
5. Topic Covered (Checklist)

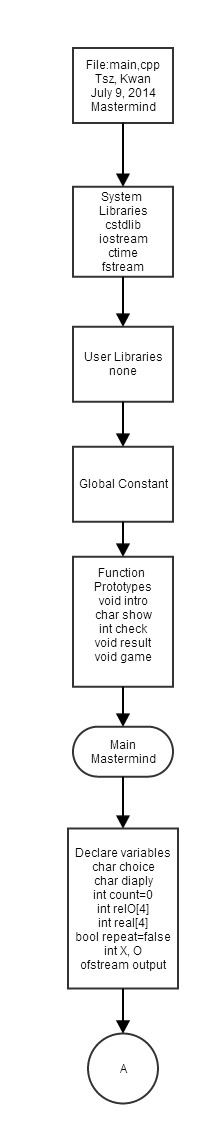
|  |  |  |  |
| --- | --- | --- | --- |
| Chapter | type | code | line |
| 2.1 Variables | int | int count=0; | 31 |
| 2.2 Input Output | cin | cin>>choice; | 46 |
|  | cout | cout<<realO[i]; | 67 |
|  | endl | cout<<endl; | 71 |
| 2.3 data types | char | char choice; | 29 |
|  | bool | bool repeat = false; | 34 |
|  | string | string sg; | 157 |
| 2.4 condition | = | repeat = false; | 58 |
|  | == | if(realO[i]==realO[j]) | 61 |
|  | ++ | X++; | 191 |
| 2.5 style | comment | //chooose level | 29 |
| 3.1 boolean expression | >=, &&, != | if(count>=10 && X!=4){ | 216 |
|  | <, >, || | if(sg[i]<'0' || sg[i]>'9'){ | 173 |
| 3.2 multiway branches | switch | switch(choice){ | 50 |
|  | if | if(count>=10 && X!=4){ | 216 |
|  | else | else{ | 219 |
|  | nested | for(int i=0; i<4; i++){ | 56 |
|  |  | do{ | 57 |
|  |  | for(int j=0;j<i;j++){ | 60 |
|  | break | break; | 75 |
| 3.3 type of loop | for | for(int i=0;i<4;i++){ | 95 |
|  | do-while | do{}while(count<10 && X!=4); | 233,236 |
| 4.2 predefined function | srand, time | srand (time(0)); | 38 |
|  | rand | realO[i]=rand()%10; | 59 |
| 4.3function prototypes | char | char show(); | 21 |
| 5.1 void function | void | void intro(); | 20 |
| 5.2 call-by-reference | & | int check(int [],int [], int& X, int& O, int count); | 22 |
| 6.1 streams and basic | ofsream declare | ofstream output; | 36 |
|  | output | output.open("note.dat"); | 40 |
|  | close | output.close(); | 125 |
| 7.1 array | int array | int realO[4]; //save | 32 |
|  | bool aray | bool same[4]; | 160 |

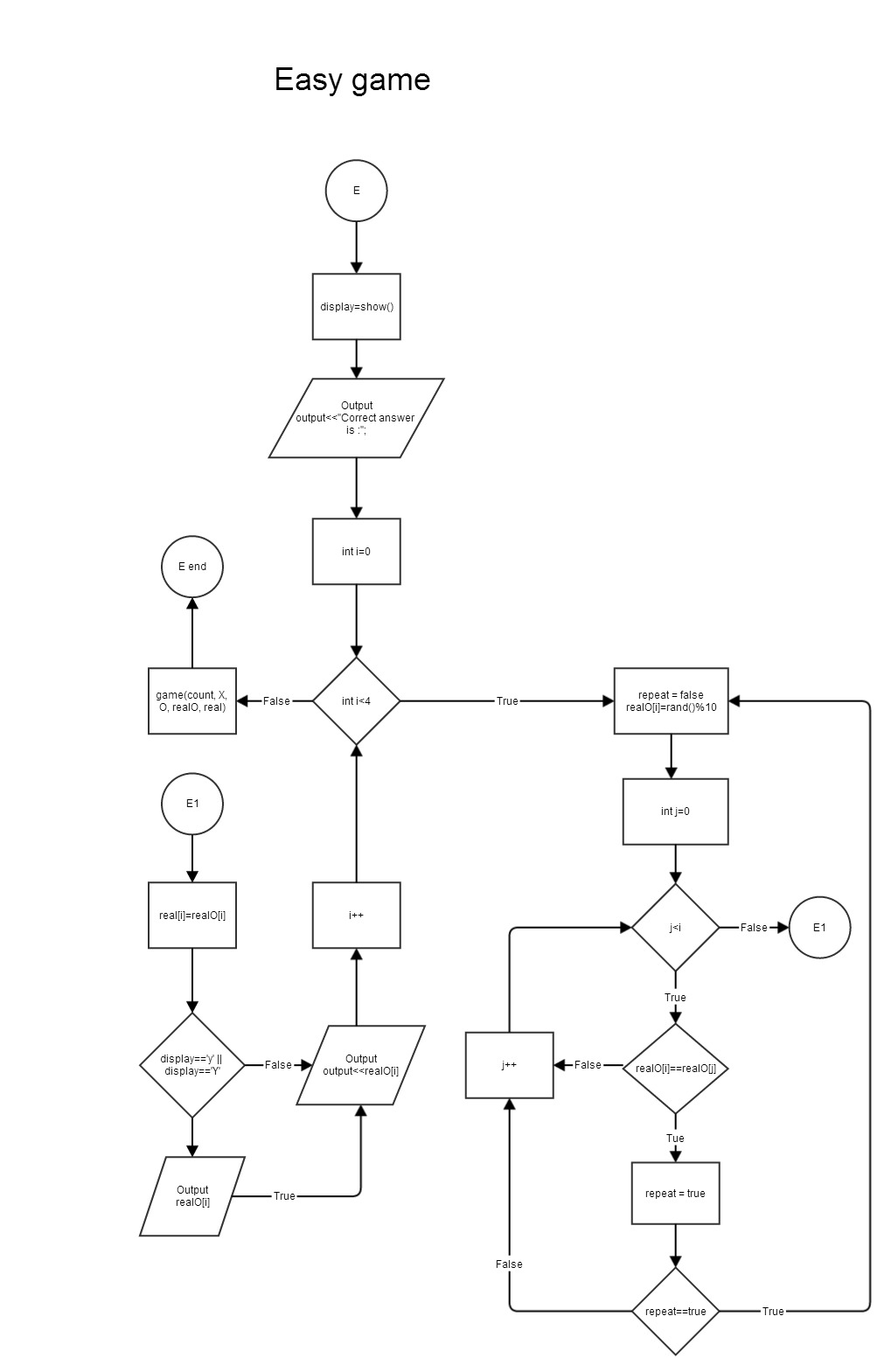
6. Libraries included

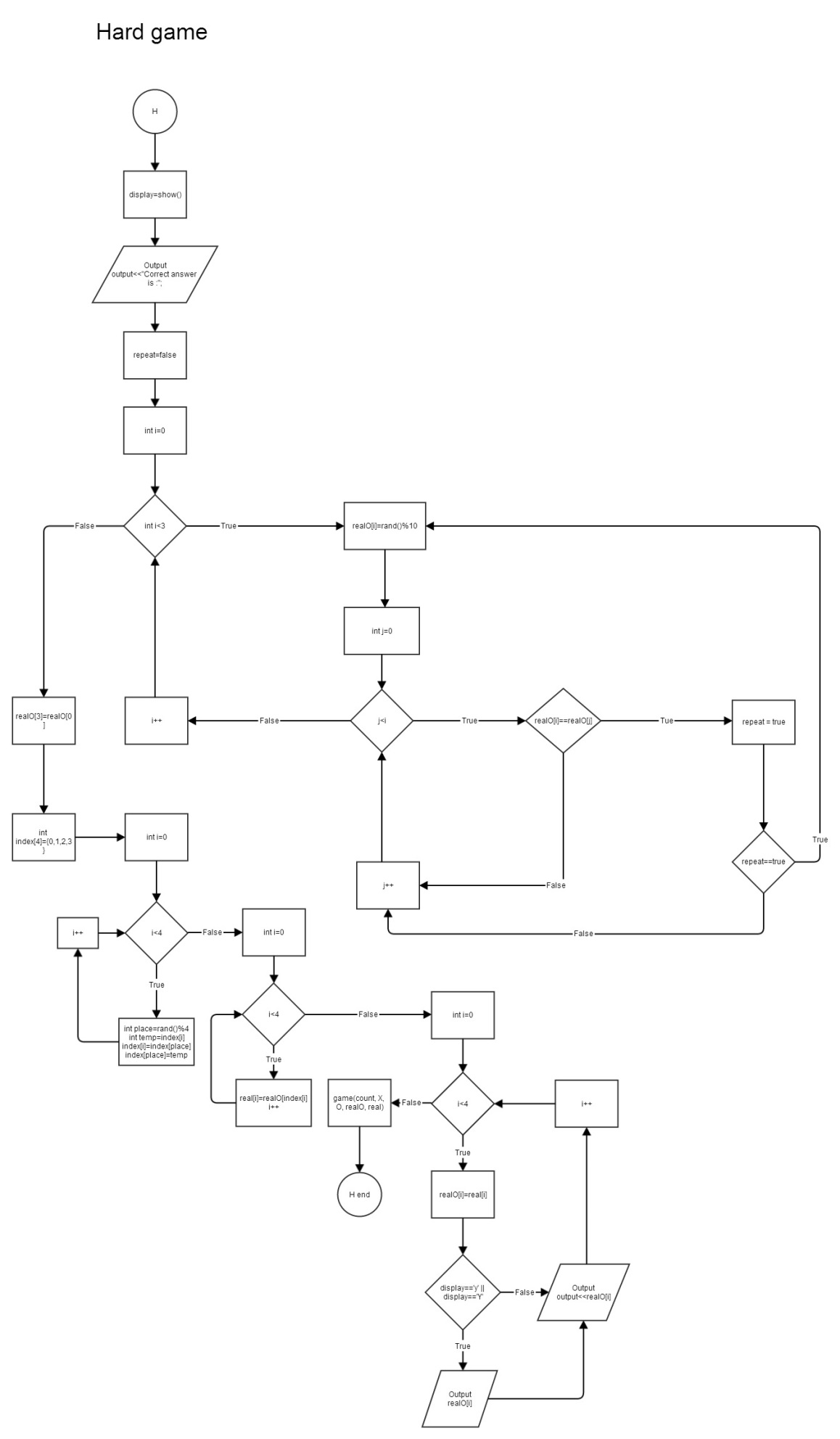
* <cstdlib>
* <iostream>
* <ctime>
* <fstream>

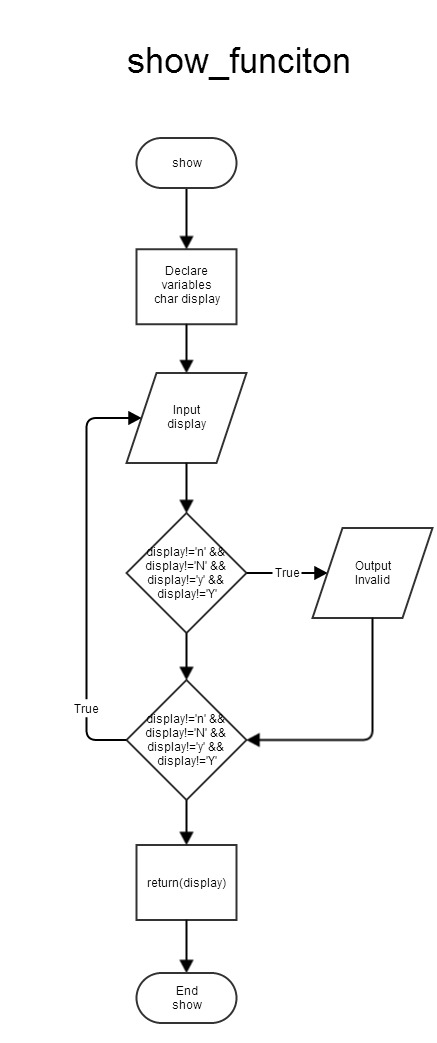
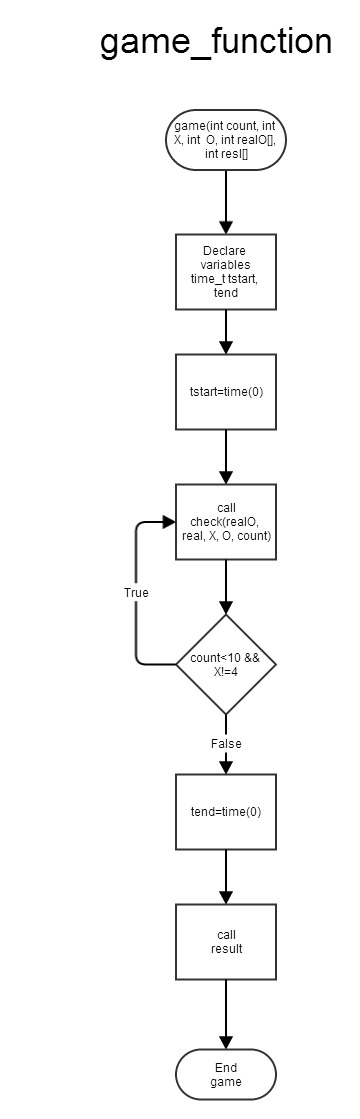
7. Flowchart

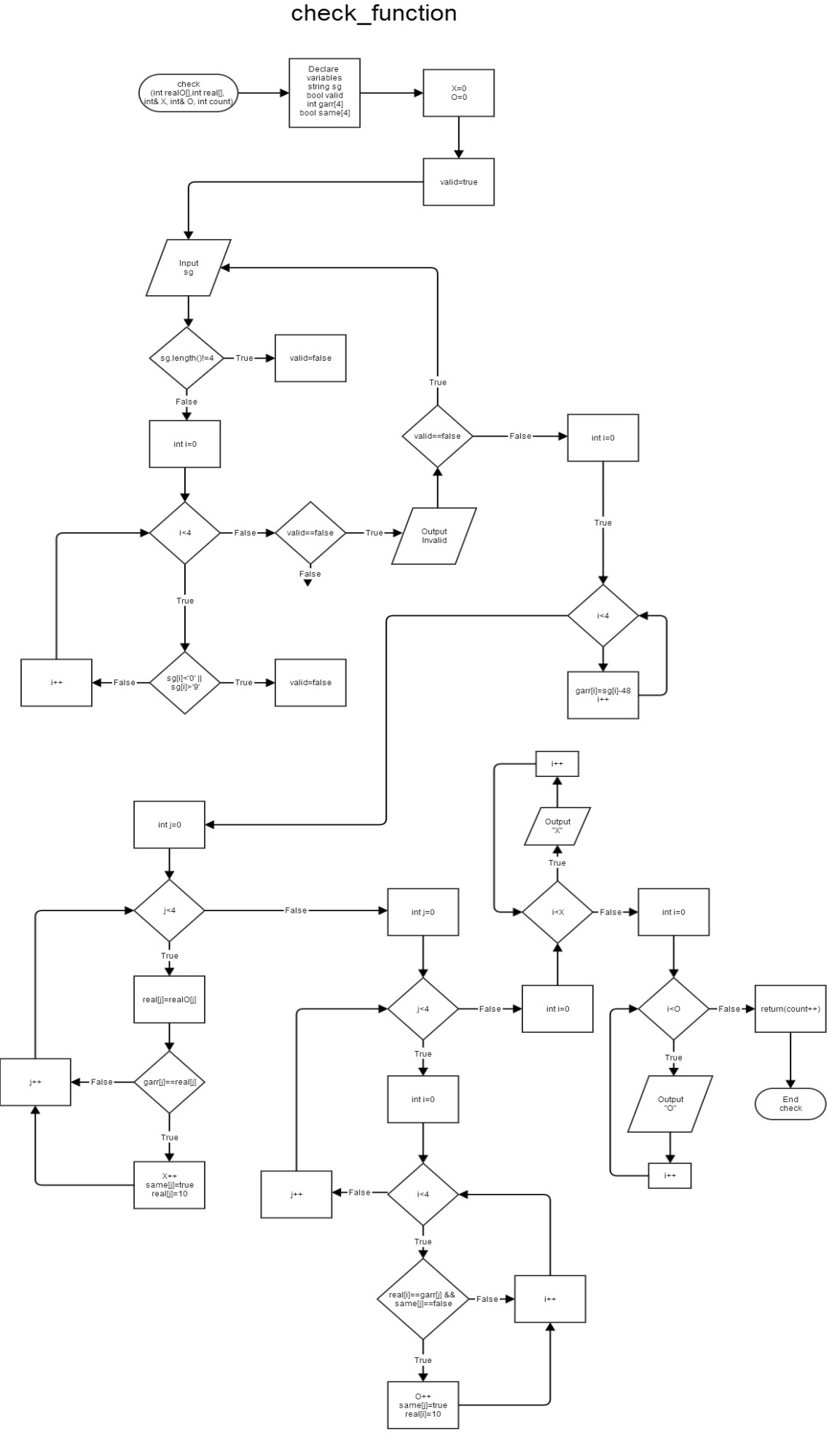


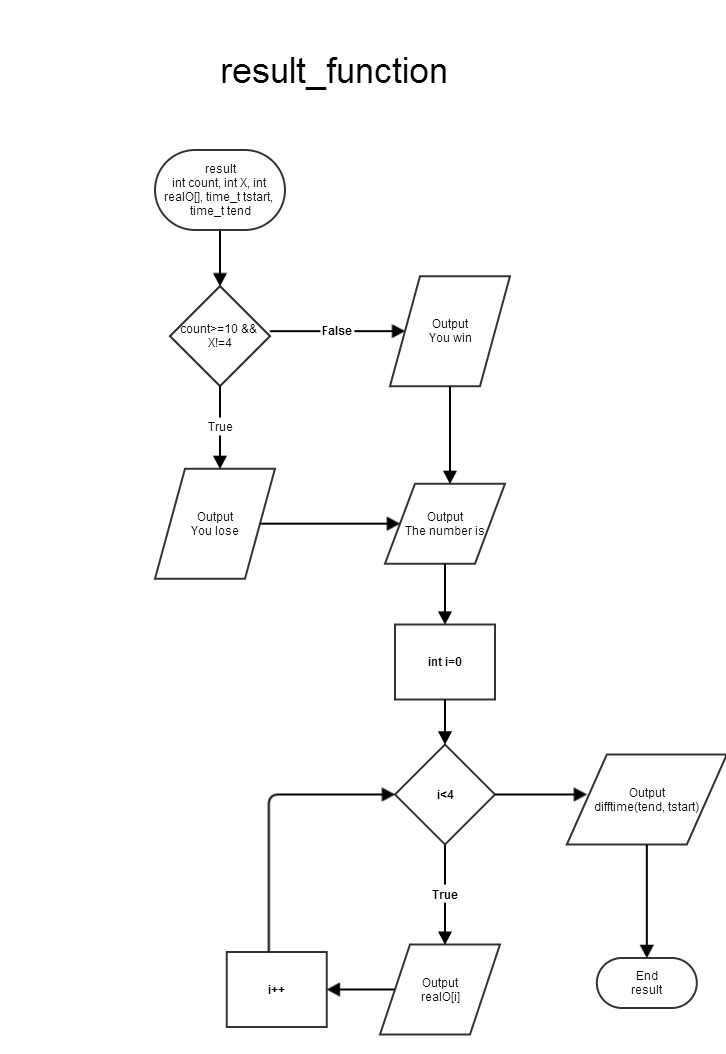












8. Code

/\*

\* File: main.cpp

\* Author: Tsz, Kwan

\* Created on July 9, 2014, 12:39 PM

\* Purpose: Mastermind

\*/

//System Libraries

#include <cstdlib>

#include <iostream>

#include <ctime>

#include <fstream>

using namespace std;

//User LIbraries

//Global Constant

//Function Prototypes

void intro();

char show();

int check(int [],int [], int& X, int& O, int count);

void result(int, int, int [], time\_t, time\_t);

void game(int, int, int, int[], int[]);

//Execution Begins Here

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

//Declare Variables

char choice; //chooose level

char display; //question of display

int count=0; //10guesses

int realO[4]; //save

int real[4]; //four digit rand

bool repeat = false; //check rand same

int X,O; //X right place, O right num

ofstream output;

srand (time(0)); //seed

output.open("note.dat");

if(output.fail()){

cout<<"Output file opening failed.\n";

}

do{

intro();

cin>>choice; //choose level

cin.ignore();

//Input

switch(choice){

case'E':case'e':

//ask display

display=show();

output<<"Correct answer is :";

//rand num (no repeat)

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

do{

repeat = false;

realO[i]=rand()%10;

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

if(realO[i]==realO[j])

repeat = true;

}

}while(repeat);

real[i]=realO[i];

if(display=='y' || display=='Y'){

cout<<realO[i];

}

output<<realO[i];

}

cout<<endl;

//game begins

game(count, X, O, realO, real);

break;

case 'H': case'h':{

//ask display

display=show();

output<<"Correct answer is :";

//rand num (2same num)

do{

repeat = false;

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

realO[i]=rand()%10;

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

if(realO[i]==realO[j])

repeat = true;

}

}

}while(repeat==true);

//copy 1st digit to last digit

realO[3]=realO[0];

//shuffle

int index[4]={0,1,2,3};

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

int place=rand()%4; //rand place

int temp=index[i]; //save memory location value

index[i]=index[place]; //shuffle location

index[place]=temp; //replace the value

}

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

real[i]=realO[index[i]];

}

for(int i=0;i<4;i++){ //replace realO[i]instead of realO[index[i] easy to use function

realO[i]=real[i];

output<<realO[i];

//display

if(display=='Y' || display=='y'){

cout<<realO[i];

}

}

cout<<endl;

//game begins

game(count, X, O, realO, real);

break;

}

default:

cout<<"Not a valid choice.\n";

cout<<"Choose again.\n";

cout<<"Press ENTER to continue...\n";

cin.ignore();

}

}while(choice!='E' && choice!='e' && choice!='H' && choice!='h');

output.close();

return 0;

}

void intro(){

//intro

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout<<"\*\* Mastermind Game! \*\*\n";

cout<<"\*\* The number is between 0000 to 9999 \*\*\n";

cout<<"\*\* After you type in your answer, it will show Xs and Os \*\*\n";

cout<<"\*\* X means one of the number is at the right position \*\*\n";

cout<<"\*\* O means one of the number matches \*\*\n";

cout<<"\*\* 2 LEVEL Easy and Hard \*\*\n";

cout<<"\*\* In easy level, the number doesn't contain same digit \*\*\n";

cout<<"\*\* In hard level, the number contains 2 same number \*\*\n";

cout<<"\*\* You have 10 chances to guess the number \*\*\n";

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout<<"Choose the level E/H :";

}

char show(){

//ask display

char display;

do{

cout<<"Do you want to display the number? (Y/N) :";

cin>>display;

cin.ignore();

if(display!='n' && display!='N' && display!='y' && display!='Y'){

cout<<"Invalid input!\n";

}

}while(display!='n' && display!='N' && display!='y' && display!='Y');

return(display);

}

int check(int realO[],int real[], int& X, int& O, int count){

string sg;

bool valid; //bool sg valid

int garr[4]; //guess in int type

bool same[4]; //remember

//reset

X=0;

O=0;

do{

valid=true; //reset

//input

cout<<"Please enter your answer : ";

cin>>sg;

if(sg.length()!=4){

valid=false;

}

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

if(sg[i]<'0' || sg[i]>'9'){

valid=false;

}

}

//if not all true = some digits not in range 0 - 9

if(valid==false){

cout<<"Invalid input!\n";

}

}while(valid==false);

for(int i=0;i<4;i++){ //transfer cg(char)to garr(int)

garr[i]=sg[i]-48;

cout<<garr[i];

}

cout<<" ";

//check matches

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

real[j]=realO[j]; //recover loaction memory

if(garr[j]==real[j]){ //same place

X++; //Count X

same[j]=true; //remember!!!! location

real[j]=10; //erase that location memory

}

}

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

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

if(real[i]==garr[j] && same[j]==false){

O++;

same[j]=true; //matches!!!! remember location

real[i]=10; //erase that location memory

}

}

}

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

cout<<"X";

}

cout<<" ";

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

cout<<"O";

}

cout<<endl;

return(count++);

}

void result(int count, int X, int realO[], time\_t tstart, time\_t tend){

if(count>=10 && X!=4){

cout<<"You lose!\n";

}

else{

cout<<"You win!\n";

}

cout<<"The number is ";

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

cout<<realO[i];

}

cout<<endl;

cout<<"You use "<<difftime(tend, tstart)<<" second(s).\n";

}

void game(int count, int X, int O, int realO[], int real[]){

time\_t tstart, tend;

tstart=time(0);

do{

//Input & check matches

check(realO, real, X, O, count);

}while(count<10 && X!=4);

//Result

tend=time(0);

result(count, X, realO, tstart, tend);

}