**Thomas Saldana**

**Dr. Lehr**

**CSC-5, Section 42450**

**25 April 2014**

**V2**

**CSC-5 Final PROJECT**

**Bulls and Cows**

**Table of Contents**

Bulls and Cows Game                                                                               Page

Project Summary                                                                                             3

Project Overview                                                                                             6

Instructions                                                                                                       7

Pseudo Code                                                                                                    8

Source Code                                                                                                    10

**Project Summary**

The following is a brief explanation of the content that is found in this project:

-Project Size: 350+ Lines

-Part one of program:Lines 25-121

-Part two of program:Lines 124-298

- Libraries Used: 6

              •cstdlib

              •iomanip

              •iostream

              •ctime

              •string

              •fstream

-Number of Functions Used: 3

-Number of 1-D Arrays Used:0

-Number of 2-D Arrays Used: 1 (line151)

-Values Passed by Reference:1(line 59)

            Passed by Value:10

-Selection Structures Used: 3

               • if

               • if/else

               • switch

-Repetition Structures Used: 3

               • while

               •do/while

               •for

The first thing that appears in this program is the system libraries (lines 9-14). After that, lines 18-21 hold the function prototypes, which allow the compiler to check if the right functions are being called. There are three functions in this program: **bool check (int, int, int, int, int)**, **int srtngfnct(int, int, int, int, int)**, and **int** **srtngfnct2(int, int, int, int, int)**. First, the bool check() function examines the users guesses for validity.  It also adds 1 to the variable “tries,” which counts how many times the user takes to guess the computers number.  The variable “tries” is passed by reference to the function bool check() so that its value does not return to zero (or its original value) after every call. Next, the function srtngfnct() sorts a four digit number in  an escalator-like manner. By way of example, srtngfnct() takes a number like, let’s say.., 4321 and outputs 3214, then 2143, and so on and so on. The last function, srtngfnct2(), also functions as a sorting function. It takes a four digit number, and it outputs every possible- different combination of that number. For example, if the number 4321 is passed to srtngfnct2, srtngfnction2() would return 4321 as 4312,  2413, 3214 or any other of the 24 possible combinations for this number.  After the compiler finishes reading lines 0-23, it reaches the function main(). The function main() starts at line 23 and it ends at line 301. It contains most of the information that makes this program do its designated task.

The first lines within main()-- lines 26-30--is where the declaration of variables takes place. Within these lines, 13 variables of data type int and one variable of data type bool are declared. After the declaration of variables takes place on lines 26-30, lines 34-42 follow. These lines contain the code that allows the program to read and display the content of a file that has the instructions of the game Bulls and Cows. These instructions are displayed immediately to the user, right after the program is ran. After lines 34-42, the function srand() follows. This function allows this program to generate random numbers. Specifically, this function is used to generate four different, random numbers that lie in the range [0, 9]. These four numbers are generated on lines 47-50. After these numbers are generated, they are joined together by line 52. This is how the computer makes a random, four-digit number. It first makes four numbers, and then it adds them by using multiplication and addition to make one four digit number with distinct digits.

After line 52 generates a  four-digit number, lines 54-59 tell the user to try to guess the number in line 52, and it also adds 1 to the count “tries”.  Lines 64-68 output a message if and only if the user is able to guess the computers number on the first try. Of course however, the odds of this happening are too great, but it can still happen--which is why these lines of code are included in the program.

The main part, of part 1 of this program starts on line 71 and it ends on line 112. This part of the program lies within a while structure that executes over and over until the user of the program guesses the computers number. Inside this while structure, many selection and repetition structures are embedded. All the structures embedded in this while structure work to determine if any of the users’ guesses are equal to the computer’s, random four-digit number. They do this with the help of assignment and logical operators. This while structure also displays to the user the number of bulls and the number of cows he or she gets after every guess. Whenever the user’s guess  matches the computers number, this while structure ends and the if structure that comes right after on line 114 executes, displaying a message congratulating the user for guessing the computers number and finishing part one of the program.

Part 2 of this program starts on line 123. Part two is the opposite of part one of this program. In part one of this program the user tries to guess the computer’s random, four-digit number; in part two, the computers tries to guess the user’s random, four-digit number. In other words, part one of this program is the users turn, while part two is the computers turn. Part two of this program starts by instructing the user (lines 124-130). After lines 124-130, more declaration of variables takes place again (lines 132-137), and after the declaration of variables takes place, a while structure (which starts on line 139 and ends on line 296) that holds most of the code for part two of the program follows. This while structure that follows generates a different four digit guess every time it is compiled. This four digit guess is displayed to the user after every loop, and the user informs (also after every loop) the computer if it is the right number or not. If it is the right number, the program determines if the computer guessed the users number in fewer tries than the user or if the user guessed the computers number in fewer tries. If it is not the right number, the while loop repeats until the user’s number is found or until the computer is not able to find the computers number. After this while structure, the computer calculates who is the winner, informs the user who won, and ends the program.

**Project Overview:**

In this project, I will be creating a computer game using the C++ language. The name of the game is called “Bulls and Cows.” It is a simple code-breaking, mind game for two or more players. This game is fairly simple to learn, and it is not as complex as Mastermind--which is the complex version of this game. The instructions for this game will be explained on the following page. Moreover, this game can be played using only a pen or pencil and paper.  However, because it will be played on the computer, no pen or pencil is necessarily required, but it is recommended. All that is needed to play this game is a computer, a NetBeans IDE, and the source code.

Although I mentioned that this game could be played by two or more players, only two players will be playing in this game. Namely, the computer user, whoever that might be, and the computer. When the program starts the user will go first, and he or she will try to guess the computers number in as few tries as possible. Then, the computers turn to guess the user’s number will start (in part two), right after the user guesses the computers number. The computer will try to guess the users number in as few tries as possible as well. The winner will be the one who guesses the others four-digit number in less tries.

**Instructions**

**Objective:** The whole object of this game is to try to guess the four-digit number the computer randomly thinks of.

**How to Play:** First, when the computer prompts you to do it, type a four-digit number whose digits do not repeat into the keyboard. Then wait for the computer to tell you if you guessed the right-four digit number. If you do not guess the right number on the first try, the computer will tell you hints so that you can guess the right number on future tries. Those hints are called BULLS and COWS. The computer will display the number of BULLS and the number of COWS in each one of your guesses. A BULL is a correct digit, in the correct spot. A COW is a right digit, in the incorrect spot. For example, if your guess is 4567, and the computers number is 4659, the computer will tell you that you got 1 BULL and 2 COW’s in your guess. The BULL  will be the digit 4 in your guess because it is in the exact location as the four that is in the computers number, and  the COW will be the digit  5 and 6 in your guess because they are the same, but in a different location,  as two of the digits in the computer’s number . To win, keep entering 4-digit guesses, whose digits do not repeat, into the computer and keep in mind the bulls and cows (which are hints from the computer). You will win the game if and only if you guess the computers number before the computer guesses yours.

**Pseudo Code**

-write necessary block comments (This will be done by reading a file)

-identify preprocessor directives

-function prototypes

-global constants

-main

-declaration of variables

-tell user about the game

- set random number seed

-make computer think of a number

-make the computer think of an number

-check if the guess of the user is correct the first time he inputs a guess

-check if the range of the user input( range is composed of 4 numbers)

-check the first digit of the user’s four digit number to see if it is a cow or a bull

- mod guess with 1000

-check the second digit of the user’s four digit number to see if it is a cow or a bull

- mod new guess with 100

-check the third digit of the user’s four digit number to see if it is a cow or a bull

-mod  guess with 10

-check the fourth digit of the user’s four digit number to see if it is a cow or a bull

-output result of the program

- repeat this code until user guesses the computers number

-Tell the user what the instructuion are

-declare variables

-set a structure that will loop until the computer has guessed the users number

-set the random number seed

-guess a four digit number(Use two d array)

-ask user if the guessed number is the same as his or hers

-check for invalid inputs from the user

-if at any point the user informs the computer that the right number was found, break out of program and declare the winner

-find the users number using two four structures

- keep count of how many times it takes the computer to find the users number

-if the computer takes to long to find the users number, return 0

- tell user who won

-exit main

**Source Code**

/\*

\* File:   main.cpp

\* Author: Saldana, Thomas

\* V2: Final Project

\*

\* Created on 5 June 2014, 10:38 AM

\*/

//System Libraries

#include <cstdlib>

#include <iomanip>

#include <iostream>

#include <ctime>

#include <string>

#include <fstream>

using namespace std;

// No Global Constants

//Function Prototypes

bool check (int, int&);

int srtngfnct ( int, int , int , int);

int srtngfnct2 ( int, int , int , int, int);

//Execution Begins.

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

   //Declarition of variables

   int num1 = 0, num2 = 0, num3 = 0,

       num4 = 0, guess = 0,

       dgt1 = 0, dgt2 = 0, dgt3 = 0, dgt4= 0, randnum;

   int  bulls = 0, cows =0, tries = 0 ;

   bool Sentinel = true;

   cout << endl;

    //Prompt and instructions

    //This code reads from the file "instructions1.txt"

   string getcontent;

   ifstream openfile("instructions1.txt");

   if (openfile.is\_open()){

       while(!openfile.eof()){

           getline(openfile,getcontent);

           cout << getcontent << endl;

       }

   }

   cout << "\n\n";

   // Sets the random number  generator

   srand ( time(0));

   num1 = rand() % 2 + 3;//the shifting value and the scaling factor are different in order to generate a non repeating four-digit number.

   num2 = rand() % 2 + 1;// num2 has range [2,3]

   num3 = rand() % 3 + 5;// num3 has range [3,8]

   num4 = rand() % 2 + 8; // num4 has range[2,9]

   //this is the number the user has to try to guess

   randnum = num1 \* 1000 + num2 \* 100 + num3 \* 10 + num4;// reverses the processes in lines 35 to 40

   do {

   cout << "Guess the computers 4-digit number(Note: Digits in your in your 4-digit number should not repeat): "; // prompts user to guess for the first time

   cout << endl;

   cin >> guess;//input

   ++tries;

   }while( ! ( check(guess, tries) ) );

   // This while structure checks to see if the first guess of the user is correct.

   // According to probability, this construct should never execute.

   //Although this do/while structure can be omitted from the program, it is not. i like it

   do{

     if ( guess == randnum )

         cout << "\nYou guessed the computers number right from the start!\n";

     break;

   }while ( randnum == guess );

   //This while structure will repeat until user guesses the right number.

   while ( guess != randnum ){

   if ( guess / 1000 == num1 ) //Checks if the first digit of the users four digit number is a bull

       ++bulls;

   if ( ( guess/1000 != num1 ) && ( num2 == guess/1000 || num3 == guess/1000  || num4 == guess/1000 ))  //Check for cows

       ++cows;

   guess = guess % 1000;      // checks the 2nd digit

   if ( guess / 100 == num2)  //Checks for bulls

      ++bulls;

   if ( ( guess/100 != num2 ) && ( num1 == guess/100 || num3 == guess/100  || num4 == guess/100 ))  //Check for cows

      ++cows;

   guess = guess % 100;       // checks the third digit

   if ( guess / 10 == num3 )   //Checks for bulls

       ++bulls;

   if ( ( guess / 10 != num3 ) && ( num1 == guess/10 || num2 == guess/10  || num4 == guess/10 ))  //Check for cows

      ++cows;

   guess = guess % 10;        // checks the fourth digit

   if ( guess == num4 )      //Checks for bulls

       ++bulls;

   if ( ( guess != num4 )&& ( num1 == guess || num2 == guess  || num3 == guess ))  //Check for cows

      ++cows;

   cout << "Bulls: " << bulls << ", Cows: " << cows <<  endl;

      do{

   cout << "\nTry again(Digits in your in your 4-digit number should not repeat): ";

   cout << endl  ;

   cin >> guess;

   ++tries;

    }while( ! ( check(guess, tries) ));

   while ( guess < 1000 || guess > 9999){ // Checks for the range of the n input.

           cout << endl << "\nThe number you entered is incorrect.\n";

           cout << "Please enter a four digit, positive numbers--only.\n";

           cin >> guess;

  }

   //Checks to see if user guessed the right number on this try

   if ( guess == randnum ){

       cout << endl << "\nCongratulations! You guessed the computer's number in "<< tries << " tries!\n\n";

   }

   bulls = 0;

   cows  = 0 ;

   }

                                    //\*\*\*\*SECOND PART OF THIS PROJECT BEGINS HERE\*\*\*\*

   cout <<setw(30)<< "\n\t\t\t\t\*\*\*\*It is now the computers turn to guess your number!\*\*\*\*\n";

   cout << "\nSame rules apply: think of a four-digit number, and write it down ";

   cout <<"on a pice of paper\n so that you do not forget.(Please do not repeat digits and do not include the zero digit in your number)\n";

   cout << "The computer will try to guess your number, and";

   cout << " then it will display what it thinks your four-digit number is every time until it guesses your number.\n";

   cout << "In order for the computer to guess your number successfully however, you will need to tell the computer how many bulls and cows it has in its guess.\n";

   cout << "\nThe computer will now think of a number...If the computer guesses your number in less tries than you, the computer will win.\n";

   int x = 0; // x is going to be equal to the number of bulls, since i already used the variable "bulls" previously in the program

   int y = 0; // y is going to be equal to the number of cows, since i already used the variable "cows" previously in the program

   int a =0, b=0, c=0, d=0;

   int tries2 = 0;

   int found = 0, frstdgt,scnddgt,thrddgt,

       frthdgt, frstgss, a1, b1, c1, d1;

   //Begining of code that will find the users number

   while ( (x != 4) ){

       //set the random number seed

       srand(time(0));

       frstgss = rand() % 9000 + 1000;//this is a random number between 1000 and 9999

       frstdgt = frstgss /1000;//First digit

       frstgss = frstgss %1000;//make frstgss a three digit number

       scnddgt = frstgss /100; //Second digit

       frstgss = frstgss % 100;//make frstgss a two digit number

       thrddgt = frstgss/10;   // third digit

       frstgss = frstgss % 10; //make frstgss a two didit number

       frthdgt = frstgss;      //first digit

       const int Rows = 2, Cols = 2;// Set the constant variables for 2D array usrnum[][];

       //2 Dimensional Array

       int usrnum[Rows][Cols] = { { frstdgt, scnddgt}, {thrddgt, frthdgt}};

          cout<< "\nIs your number " << usrnum[0][0]

               << usrnum[0][1]

               << usrnum[1][0]

               << usrnum[1][1]

               << " ?\n";

          found = usrnum[0][0] \* 1000 + usrnum[0][1] \* 100 + usrnum[1][0] \* 10 + usrnum[1][1]; // combines all the subscripts in the 2D array into one number

          cout << "\nType in number of bulls:";//Prompt

          cin>> x;//Input

          cout << "Type in the number of cows:";//Prompt

          cin >> y;//Input

          tries2++;// increment the tries of the computer

           // Checks for invalid inputs

           while (x > 4 || x < 0 || y > 4 || y < 0 ){

              cout << "\nERROR!! The number of bulls and cows has to be between 0 and 4.";

                       cout << "\nType in number of bulls:";

              cin>> x;

              cout << "Type in the number of cows:";

              cin >> y;

              cout << endl;

          }

          //this if structure will cause this while structure to stop looping when the computer finds the computers number

          if ( x == 4)

                     break;

          cout << endl;

          // This while structure will loop untill the computer finds a combination of bulls and cows that add up to 4.

          while ( ! ((x==3 && y== 1) || (x==2 && y==2) || (x==1 && y==3) || (x==0 && y==4))    ){

                  int fndnum = rand()%9000 + 1000;

                  a1= fndnum/1000;

                  fndnum= fndnum%1000;

                  b1= fndnum/100;

                  fndnum = fndnum%100;

                  c1= fndnum/10;

                  fndnum= fndnum%10;

                  d1= fndnum;

                  cout << "\nIs your number "<< a1 << b1 << c1 << d1 << " ?\n";

                   found = a1 \* 1000 + b1 \* 100 + c1 \* 10 + d1;

                  cout << "\n\nType in number of bulls:";

                  cin>> x;

                  cout << "Type in the number of cows:";

                  cin >> y;

                  tries2++;

                  if (tries2 > tries){

                      cout << "\nYou guessed the computers number in " << tries << " tries. The computer could not top that!\n"

                           << "You Win!!\n";

                      return 0;

                  }

                  if (x==4) //DONT FORGET TO ADJUST!!!!

                      break;

                  // Checks range of x and y . (0,4)

                while (x > 4 || x < 0 || y > 4 || y < 0 ){

                    cout << "ERROR!! The number of bulls and cows has to be between 0 and 4.";

                    cout << "\nType in number of bulls:";

                    cin>> x;

                    cout << "Type in the number of cows:";

                    cin >> y;

                    cout << endl;

          }

          }

          if ( x == 4 )

              break;

          // once this while construct executes, it will arrange the users four digit number until it finds it

          while ( (x==3 && y== 1) || (x==2 && y==2) || (x==1 && y==3) || (x==0 && y==4) ){

              //FIRST FOR STRUCTURE

              for ( int i= 1 ; i <=20 ; i ++ ){

                       a = found / 1000;         // allows us to check first digit

                           found = found % 1000; //makes guess a three digit number

                       b = found / 100;          //allows us to check second digit

                           found  = found % 100; //makes guess a two digit number

                       c = found / 10;           // allows us to check third digit

                           found = found % 10;   // makes guess a one digit number

                       d  = found;               // allows us to check fourth digit

                       found = a\*1000+b\*100+c\*10+d;

                       cout << "\nIs your number " << srtngfnct2(a,b,c,d,i)  << " ?\n";

                       cout << "\n\nType in number of bulls:";

                       cin>> x;

                       cout << "Type in the number of cows:";

                       cin >> y;

                       tries2++;

                        if (tries2 > tries){

                      cout << "\nYou guessed the computers number in " << tries << " tries. The computer could not top that!\n"

                           << "You Win!!\n";

                      return 0;

                  }

                        //Checks range of x and y (0,4)

                 while (x > 4 || x < 0 || y > 4 || y < 0 ){

                     cout << "\nERROR!! The number of bulls and cows has to be between 0 and 4.";

                     cout << "\nType in number of bulls:";

                     cin>> x;

                     cout << "Type in the number of cows:";

                     cin >> y;

                     cout << endl;

            }

                       if (x==4)

                           break;

              }

              //SECOND FOR STRUCTURE

               if ( x == 4)

                     break;

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

                       a = found / 1000;         // allows us to check first digit

                           found = found % 1000; //makes guess a three digit number

                       b = found / 100;          //allows us to check second digit

                           found  = found % 100; //makes guess a two digit number

                       c = found / 10;           // allows us to check third digit

                           found = found % 10;   // makes guess a one digit number

                       d  = found;               // allows us to check fourth digit

                       found  = srtngfnct(a,b,c,d);

                       cout << "\nIs your number " << found << " ?\n";

                       cout << "\n\nType in number of bulls:";

                       cin>> x;

                       cout << "Type in the number of cows:";

                       cin >> y;

                       tries2++;

                        if (tries2 > tries){

                      cout << "\nYou guessed the computers number in " << tries << " tries. The computer could not top that!\n"

                           << "You Win!!\n";

                      return 0;

                  }

                        //Checks range of x and y (0,4)

                 while (x > 4 || x < 0 || y > 4 || y < 0 ){

                     cout << "\nERROR!! The number of bulls and cows has to be between 0 and 4.";

                     cout << "\nType in number of bulls:";

                     cin>> x;

                     cout << "Type in the number of cows:";

                     cin >> y;

                     cout << endl;

              }

                       if (x==4)

                           break;

              }

              x = 4; y = 0;

          }

   }

      cout << "\n\nThe computer guessed your number in " << tries2 << " tries. You guessed the computers number in " << tries << " tries.\n";

      cout << "The computer wins!\n";

//Exit Stage Right

   return 0;

   }

//Function Definitions

bool check ( int guess, int& tries){

   int dgt1 = 0,dgt2 = 0,dgt3 = 0, dgt4 = 0;

   dgt1 = guess / 1000;  // allows us to check first digit

   guess = guess % 1000; //makes guess a three digit number

   dgt2 = guess / 100;//allows us to check second digit

   guess  = guess % 100;//makes guess a two digit number

   dgt3 = guess / 10;// allows us to check third digit

   guess = guess % 10;// makes guess a one digit number

   dgt4  = guess;// allows us to check fourth digit

   guess = guess;// guess has now become a one digit number

   guess = dgt1 \* 1000 + dgt2 \* 100 + dgt3 \* 10+ dgt4; // reverses the above process. line 121 to 128.

       if( guess <1000 || guess >9999 )// Checks for the range of the first input.

       {

               cout << endl << "The number you entered is incorrect.\n";

               cout << "Please enter a four digit, positive numbers--only.\n";

               //return type

               return false;

       }

       else

       {

           if (( dgt1 == dgt2)||( dgt1 == dgt3 )||( dgt1 == dgt4 )||( dgt2 == dgt3)||(dgt2 == dgt4)||( dgt3 == dgt4))   // checks  to see if any digit in the users guess repeats

               {

                   cout << "\nError!\n" << "Please enter a four-digit number whose integers do not repeat.\n"; //tells user he/she messed up

                   // return type

                   return false;

               }

           else

               {

                   return true;

               }

       }

}

//Function "srtngfnct" sorts out the users guess

int srtngfnct(int a, int b, int c, int d){

   int value = b\*1000 + c\*100 + d\*10 + a;

   return value;

//Function "srtngfnct2" sorts out the users guess in a different pattern than the srtngfnct above

}

int srtngfnct2(int a, int b, int c, int d, int i ){

   // this switch structure will return different combinations until it finds the users numbers.

   switch (i){

           case  1:

               return a\* 1000 +  b\* 100 +  d\* 10 + c;

           case  2:

               return b\* 1000 + a\* 100 + c\* 10 + d;

           case  3:

               return b\* 1000 + a\* 100 + d\* 10+c;

           case  4:

               return c\* 1000 + d\* 100 + b\* 10+a;

           case  5:

               return d\* 1000 + c\* 100 + a\* 10+b;

           case  6:

               return d\* 1000 + c\* 100 + b\* 10+a;

           case  7:

               return b\* 1000 + c\* 100 + a\* 10+d;

           case  8:

               return c\* 1000 + b\* 100 + a\* 10+d;

           case  9:

               return c\* 1000 + b\* 100 + d\* 10+a;

           case  10:

               return a\* 1000 + d\* 100 + b\* 10+c;

           case  11:

               return a\* 1000 + d\* 100 + c\* 10+b;

           case  12:

               return d\* 1000 + a\* 100 + c\* 10+b;

           case  13:

               return b\* 1000 + d\* 100 + a\* 10+c;

           case  14:

               return b\* 1000 + d\* 100 + c\* 10+a;

           case  15:

               return d\* 1000 + b\* 100 + a\* 10+c;

           case  16:

               return d\* 1000 + b\* 100 + c\* 10+a;

           case  17:

               return a\* 1000 + c\* 100 + b\* 10+d;

           case  18:

               return a\* 1000 + c\* 100 + d\* 10+b;

           case  19:

               return c\* 1000 + a\* 100 + b\* 10+d;

           case  20:

               return c\* 1000 + a\* 100 + d\* 10+b;

       default :

           cout << "\nSomething went wrong.\n";

           break;

   }

}