Bulls & Cows

Points

Points	
185	Runs correctly
15	Code Review
200	**TOTAL**

Submission

1. **Source Code**: Submit the source code (.cpp file) to Mimir.

Objectives

- Work with compound data structures.
- Use loops to iterate through elements.
- Use nested loops.

Specifications

Design

Create a PDF document containing your design and additional components.
Try yourself, but don't be afraid to discuss with others and ask instructors and TAs for hints.

- Design. The following might help you as you create your design:
  - How do you extract individual digits from a single **integer?** Note you cannot treat it as a string.
  - How do you go about creating a list of random numbers that are guaranteed to not repeat since the rand() function sometimes will give you repeated values?
  - How do you check if a user guess has repeated digits in it?
  - How do you determine if too many digits were put into a guess?
- Additional Components
  - Identify test cases, especially ones not covered in the sample run but indicated in the description. Test cases will be specific sequences of inputs and the expected outputs they would generate.

Program

This homework will be done over two weeks in the fall and spring and one week during the summer. Bulls and Cows is a guessing game where you try to guess a 'n' digit code. (Note: Each number in the code must be distinct.) When a number is guessed and a digit is in the correct location, then that is a bull. When a digit is in the number, but in the incorrect location, then that is a cow. So if a four-digit code is 2894, and the user guesses 4698. Then that is "1 bull and 2 cows" because '9' is in the correct location and '4' and '8' are in the code but are currently in the wrong location. The guessing continues until the code is guessed in the correct order... "4 bulls"! This website will give you a feel for the game.http://www.mathsisfun.com/games/bulls-and-cows.html
To emulate our version, click on "Options" and select any 0-9 choice.

Requirements

Program Flow

- Get the number of digits for the code.
  - Only accept inputs of 0, 3, 4, or 5. (Note: 0 is a cheat code)
  - Repeat until a valid choice is an input.
- Load vector with code digits
  - If 0 is input as a number of digits for the code:
    (Note: This will allow TAs/autograding to enter a code to make testing easier. Since this is for testing purposes, you do not have to check these for correct input.)
    - Get code as a single integer (int).
    - Get the number of digits for the code. (Note: You need this so you can add a leading zero if needed, e.g. they type in 29 and you need to create 029.)
    - Extract individual digits from the single integer and put into the vector for the code. Consider using **integer division** and **modulus**.
  - Else randomly create a valid code. (i.e. the number of digits was 3, 4 or 5)
    - Load vector with the **required number** of random digits ensuring there are **no duplicates**.
- Regardless of the method, you used to create the code (entered from the keyboard or randomly generated), output the code to be guessed. (This is purely to help with auto-grading. Of course in a published game you would not output the code.)
- Get guess as a single integer.
  - Extract single digits from the single integer and put in a vector for the code.
    - If not enough digits assume the missing digits to the left are zero.

        - *For example: For a 3 digit code, 35 is the same as 0-3-5 where zero is the first digit. For a 4 digit code, 35 is the same as 0-0-3-5.*
      - Ensure the guess is a valid guess!
        - If it is invalid, report why the guess is not good and get it again.
        - Invalid
          - More digits than the number of digits in the code.
          - Digits are repeated.
        - If both problems occur, only report that the code has too many digits.
- Output results of the guess.
  - Bulls and cows on separate lines.
  - When the correct number of bulls and echo the guess back. Be sure to include a leading zero if it has one. E.g. 0-2-3 and not 2-3.
- Match the output of the sample runs.
- Do not use functions to find something that you could do by iterating through the vector yourself. E.g. Do not use something like .find() instead of practicing iterating through a compound data structure (i.e. vector) element by element and determining if there is a match during the iteration.

Note: Be sure to match the output of the sample runs.

Coding Requirements

- Use this starting code: main.cpp
    - The first line of the main function must be left at the beginning along with the #includes needed for the things on that line. If you remove or change it, you will likely lose points for 3 test cases worth 6 points.
    - If you're having problems accessing the link above, you can copy from here:

```
#include <iostream>
#include <vector>
#include <cstdlib>
#include <chrono>

using namespace std;

int main() {
   srand(std::chrono::duration_cast<std::chrono::milliseconds>
    (std::chrono::system_clock::now().time_since_epoch()).count()%2000000000);
   // needed to autograde some test cases in Mimir
}
```

- The code and guess must each be stored in a vector of ints.
- Follow output formatting as demonstrated in the "Sample Runs" below.
- You cannot use string comparisons.
- You must loop through the code and guess.
- Even though we only have you do codes of 3, 4 or 5 digits, your program should be easy to modify and scale to 2 - 9 digits.

Sample Runs

This is not complete testing!

```
Enter number of digits in code (3, 4 or 5): 0
Enter code: 2894
Enter number of digits in code: 4
Number to guess: 2-8-9-4↵
Enter Guess: 5555
Each number must be different.↵
Enter Guess: 59
Each number must be different.↵
Enter Guess: 12345
You can only enter 4 digits.↵
Enter Guess: 112233
You can only enter 4 digits.↵
Enter Guess: 4698
1 bulls↵
2 cows↵
Enter Guess: 9687
0 bulls↵
2 cows↵
Enter Guess: 2894
4 bulls... 2-8-9-4 is correct!↵
```

```
Enter number of digits in code (3, 4 or 5): 0
Enter code: 29
Enter number of digits in code: 3
Number to guess: 0-2-9↵
Enter Guess: 89
2 bulls↵
0 cows↵
Enter Guess: 29
3 bulls... 0-2-9 is correct!↵
```

```
Enter number of digits in code (3, 4 or 5): 5
Number to guess: 4-6-9-7-3↵
Enter Guess: 49763
2 bulls↵
3 cows↵
Enter Guess: 46973
5 bulls... 4-6-9-7-3 is correct!↵
```