Bulls and Cows

You are playing the following Bulls and Cows game with your friend: You write down a number and ask your friend to guess what the number is. Each time your friend makes a guess, you provide a hint that indicates how many digits in said guess match your secret number exactly in both digit and position (called "bulls") and how many digits match the secret number but locate in the wrong position (called "cows"). Your friend will use successive guesses and hints to eventually derive the secret number.

For example:

```
Secret number:  "1807"
Friend's guess: "7810"
```

Hint: `1` bull and `3` cows. (The bull is `8`, the cows are `0`, `1` and `7`.)
Write a function to return a hint according to the secret number and friend's guess, use `A` to indicate the bulls and `B` to indicate the cows. In the above example, your function should return `"1A3B"`.

Please note that both secret number and friend's guess may contain duplicate digits, for example:

```
Secret number:  "1123"
Friend's guess: "0111"
```

In this case, the 1st `1` in friend's guess is a bull, the 2nd or 3rd `1` is a cow, and your function should return `"1A1B"`.
You may assume that the secret number and your friend's guess only contain digits, and their lengths are always equal.

Credits:
Special thanks to @jeantimex for adding this problem and creating all test cases.

Solution 1

The idea is to iterate over the numbers in `secret` and in `guess` and count all bulls right away. For cows maintain an array that stores count of the number appearances in `secret` and in `guess`. Increment cows when either number from `secret` was already seen in `guest` or vice versa.

```
public String getHint(String secret, String guess) {
    int bulls = 0;
    int cows = 0;
    int[] numbers = new int[10];
    for (int i = 0; i<secret.length(); i++) {
        int s = Character.getNumericValue(secret.charAt(i));
        int g = Character.getNumericValue(guess.charAt(i));
        if (s == g) bulls++;
        else {
            if (numbers[s] < 0) cows++;
            if (numbers[g] > 0) cows++;
            numbers[s] ++;
            numbers[g] --;
        }
    }
    return bulls + "A" + cows + "B";
}
```

A slightly more concise version:

```
public String getHint(String secret, String guess) {
    int bulls = 0;
    int cows = 0;
    int[] numbers = new int[10];
    for (int i = 0; i<secret.length(); i++) {
        if (secret.charAt(i) == guess.charAt(i)) bulls++;
        else {
            if (numbers[secret.charAt(i)-'0']++ < 0) cows++;
            if (numbers[guess.charAt(i)-'0']-- > 0) cows++;
        }
    }
    return bulls + "A" + cows + "B";
}
```

written by ruben3 original link here

Solution 2

```
public class Solution {
    public String getHint(String secret, String guess) {
        int bull = 0, cow = 0;

        int[] sarr = new int[10];
        int[] garr = new int[10];

        for(int i = 0; i < secret.length(); i++){
            if(secret.charAt(i) != guess.charAt(i)){
                sarr[secret.charAt(i)-'0']++;
                garr[guess.charAt(i)-'0']++;
            }else{
                bull++;
            }
        }

        for(int i = 0; i <= 9; i++){
            cow += Math.min(sarr[i], garr[i]);
        }

        return (bull + "A" + cow + "B");
    }
}
```

written by harish-v original link here

Solution 3

The idea is simple, if two char is match, add aCnt, otherwise, record it and process bCnt in second pass.

```
class Solution {
public:
    // only contains digits
    string getHint(string secret, string guess) {
        int aCnt = 0;
        int bCnt = 0;
        vector<int> sVec(10, 0); // 0 ~ 9 for secret
        vector<int> gVec(10, 0); // 0 ~ 9 for guess
        if (secret.size() != guess.size() || secret.empty()) { return "0A0B"; }
        for (int i = 0; i < secret.size(); ++i) {
            char c1 = secret[i]; char c2 = guess[i];
            if (c1 == c2) {
                ++aCnt;
            } else {
                ++sVec[c1-'0'];
                ++gVec[c2-'0'];
            }
        }
        // count b
        for (int i = 0; i < sVec.size(); ++i) {
            bCnt += min(sVec[i], gVec[i]);
        }
        return to_string(aCnt) + 'A' + to_string(bCnt) + 'B';
    }
};
```

written by moumoutsay original link here