MICROSOFT
#3 Bulls and Cows

Medium

You are playing the **Bulls and Cows** game with your friend.

You write down a secret number and ask your friend to guess what the number is. When your friend makes a guess, you provide a hint with the following info:

- The number of "bulls", which are digits in the guess that are in the correct position.
- The number of "cows", which are digits in the guess that are in your secret number but are located in the wrong position. Specifically, the non-bull digits in the guess that could be rearranged such that they become bulls.

Given the secret number `secret` and your friend's guess `guess`, return *the hint for your friend's guess*.

The hint should be formatted as `"xAyB"`, where `x` is the number of bulls and `y` is the number of cows. Note that both `secret` and `guess` may contain duplicate digits.

Example 1:

```
Input: secret = "1807", guess = "7810"
Output: "1A3B"
Explanation: Bulls are connected with a '|' and cows are
underlined:
"1807"
  |
"7810"
```

```
string getHint(string secret, string guess) {
    unordered_map<char,int>m;
    for(auto x:secret)
       m[x]++;
    int n=secret.length();
    int bulls=0,cows=0;
    for(int i=0;i<n;i++)
    {
       if(secret[i]==guess[i])
       {
          bulls++;
          m[secret[i]]--;
       }
    }
    for(int i=0;i<n;i++)
    {
       if(secret[i]!=guess[i])
       {
          if(m.find(guess[i])!=m.end())
          {
             if(m[guess[i]]!=0){
                cows++;
                m[guess[i]]--;
                }
          }
       }
    }
    return to_string(bulls)+"A"+to_string(cows)+"B";
}
```

Microsoft:
#3 Explanation:

- First we'll store frequencies of all the characters. Then if at any index secret character equals to guess.. increment bulls by 1 and decrease the frequency in Hashmap by 1

- Now to count bulls where secret character is not equal to guess as that case is already handled for bulls, we'll check if the character is present in secret string at any other index (location),increment cows by 1 and decrease the frequency in Hashmap by 1.

Complexity

- Time complexity:O(N), N is length of the string
- Space complexity: `O(N),N is length of string (Storing occurrence of characters in Hashmat`