Jewels and Stones

You're given strings `J` representing the types of stones that are jewels, and `S` representing the stones you have. Each character in `S` is a type of stone you have. You want to know how many of the stones you have are also jewels.

The letters in `J` are guaranteed distinct, and all characters in `J` and `S` are letters. Letters are case sensitive, so `"a"` is considered a different type of stone from `"A"`.

Example 1:

```
Input: J = "aA", S = "aAAbbbb"
Output: 3
```

Example 2:

```
Input: J = "z", S = "ZZ"
Output: 0
```

Note:

- `S` and `J` will consist of letters and have length at most 50.
- The characters in `J` are distinct.

Solution 1

First, read `J` and build jewels hash set.
Second, read `S` and count jewels.

I used hash set so that the time complexity will be**O(N),** instead of O(MN)

C++

```
int numJewelsInStones(string J, string S) {
        int res = 0;
        set<char> setJ(J.begin(), J.end());
        for (char s : S) if (setJ.count(s)) res++;
        return res;
    }
```

Java:

```
public int numJewelsInStones(String J, String S) {
        int res = 0;
        Set setJ = new HashSet();
        for (char j: J.toCharArray()) setJ.add(j);
        for (char s: S.toCharArray()) if (setJ.contains(s)) res++;
        return res;
    }
```

Python:

```
def numJewelsInStones(self, J, S):
        setJ = set(J)
        return sum(s in setJ for s in S)
```

written by lee215 original link here

Solution 2

```
    public int numJewelsInStones(String J, String S) {
        if(J.length() == 0 || S.length() == 0)
            return 0;

        // Using helper array for O(1) loopkup when traversing S
        int[] jewels = new int[58];
        for(int i=0; i<J.length(); i++) {
            jewels[(J.charAt(i) - 'A')] = 1;
        }

        int result = 0;
        for(int i=0; i<S.length(); i++) {
            if(jewels[(S.charAt(i) - 'A')] == 1) {
                result++;
            }
        }

        return result;
    }
}
```

Why an array of length 58? Two reasons

1. We need to consider both upper-case and lower-case characters
2. ASCII of 'z' - 'A' is 58. (https://www.cs.cmu.edu/~pattis/15-1XX/common/handouts/ascii.html)

Thanks to @thaliahard for pointing out the reduction in array size.

written by SkandaB original link here

Solution 3

Ruby

```ruby
def num_jewels_in_stones(j, s)
  s.count(j)
end
```

```ruby
def num_jewels_in_stones(j, s)
  s.scan(/[#{j}]/).size
end
```

```ruby
def num_jewels_in_stones(j, s)
  s.chars.count { |c| j.include?(c) }
end
```

Python

```python
def numJewelsInStones(self, J, S):
    return sum(map(J.count, S))
```

```python
def numJewelsInStones(self, J, S):
    return sum(map(S.count, J))              # this one after seeing https://discuss.leetcode.com/post/244105
```

```python
def numJewelsInStones(self, J, S):
    return sum(s in J for s in S)
```

Java

```java
public int numJewelsInStones(String J, String S) {
    return S.replaceAll("[^" + J + "]", "").length();
}
```

written by StefanPochmann original link here