1. **First Bad Version**

/\* The isBadVersion API is defined in the parent class VersionControl.

boolean isBadVersion(int version); \*/

public class Solution extends VersionControl {

public int firstBadVersion(int n) {

int l=1;

int r=n;

int ans=0;

System.out.println(l+" "+r);

while(l<=r){

int m=l+(r-l)/2;

if(isBadVersion(m)==true){

ans=m;

r=m-1;

}

else{

l=m+1;

}

}

return ans;

}

}

Learning: Initially I tried linear search as brute force but time limit was exceeded for large inputs. However I got to know since array is sorted, I can apply Binary Search and find the first value which is true.

1. **Jewels and Stones**

class Solution {

public int numJewelsInStones(String J, String S) {

int count=0;

HashSet<Character> jewels= new HashSet<Character>();

char[] j =J.toCharArray();

char[] s =S.toCharArray();

for(int i=0;i<j.length;i++){

jewels.add(j[i]);

}

for(int i=0;i<s.length;i++){

if(jewels.contains(s[i])){

count++;

}

}

return count;

}

}

Learning: I just found out that string questions are easy, but I complex it just l did above creating 2 character array and a hashset. So I have to learn Strings in Java and master it.

**Ransom Note**

class Solution {

public boolean canConstruct(String ransomNote, String magazine) {

int i=0;

HashMap<Character,Integer> map=new HashMap<Character,Integer>();

char[] mag=magazine.toCharArray();

for(char j:mag){

if(map.containsKey(j)==true)

map.put(j,(map.get(j))+1);

else

map.put(j,1);

}

System.out.println(Arrays.asList(map));

boolean match=true;

while(i<ransomNote.length() && match==true){

if(map.containsKey(ransomNote.charAt(i))){

if(map.get(ransomNote.charAt(i))!=0){

map.put(ransomNote.charAt(i),map.get(ransomNote.charAt(i))-1);

match=true;

}

else

match=false;

}

else{

match=false;

}

i++;

}

return match;

}

}

Learning: To keeping a count array in form of hashmap and decrement once an element is found.