Experiment- 3.2

Student Name: Jatin UID: 20BCS5951

Branch: BE CSE Section/Group: NTPP_605-B

Semester: 6th Subject Code: 20CSP-351

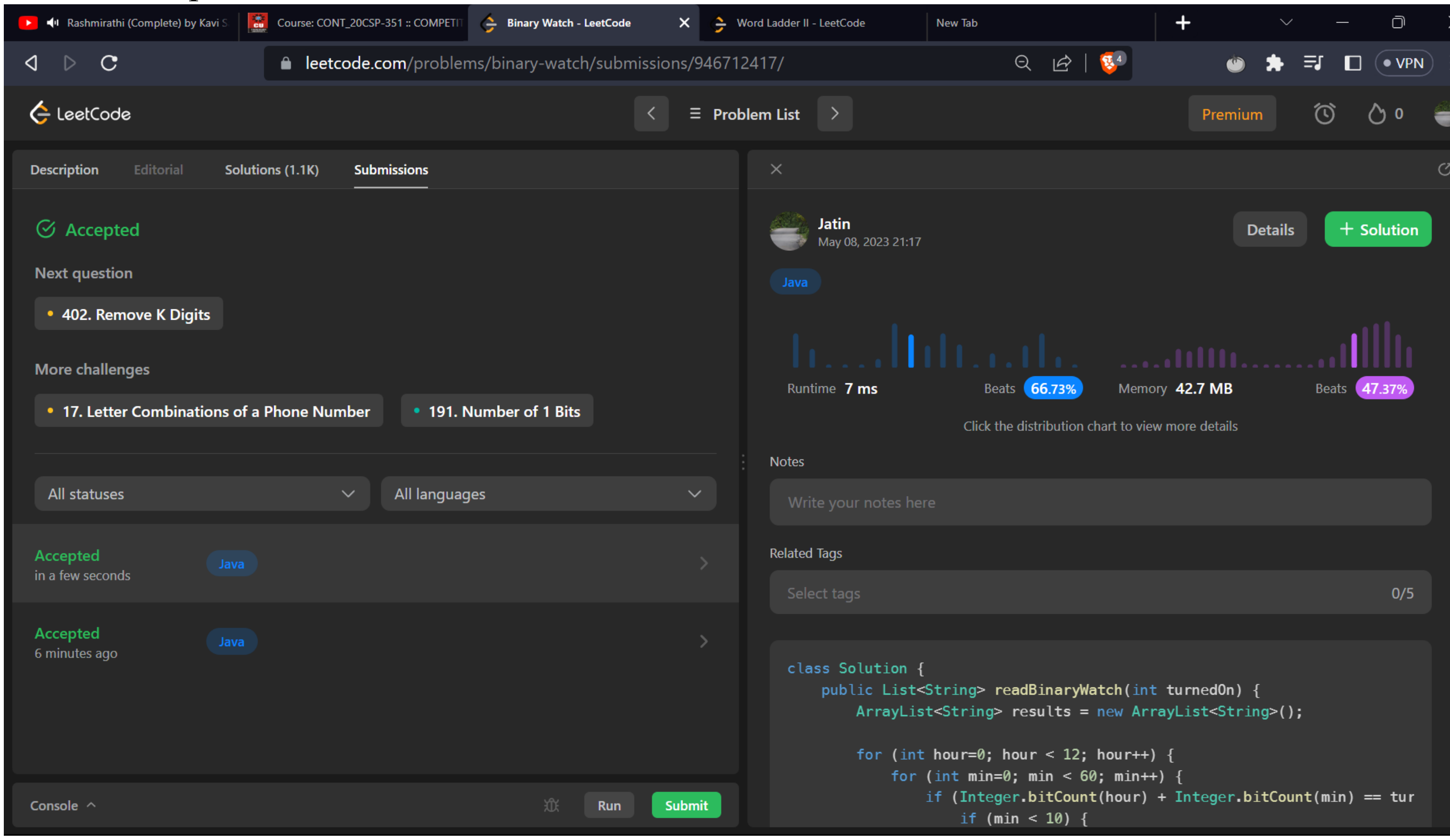
Subject Name: CC Lab

Question : Binary Watch Problem1:

. Code:

```
class Solution {
    public List<String> readBinaryWatch(int turnedOn) {
        ArrayList<String> results = new ArrayList<String>();


        for (int hour=0; hour < 12; hour++) {
            for (int min=0; min < 60; min++) {
                if (Integer.bitCount(hour) + Integer.bitCount(min) == turnedOn){
                    if (min < 10) {
                        results.add(String.format("%d:0%d", hour, min));
                    }
                    else{
                        results.add(String.format("%d:%d", hour, min));
                    }
                  }
                }
              }
            return results;
          }
        }
```

Output:


Rashmirathi (Complete) by Kavi S
Course: CONT_20CSP-351 :: COMPETI
Binary Watch - LeetCode
Word Ladder II - LeetCode
New Tab
leetcode.com/problems/binary-watch/submissions/946712417/
VPN
LeetCode
Problem List
Premium
0
Description
Editorial
Solutions (1.1K)
Submissions
Details
Solution
Console
Run
Submit

Accepted

Next question

402. Remove K Digits

More challenges

17. Letter Combinations of a Phone Number
191. Number of 1 Bits

All statuses
All languages

Accepted in a few seconds Java
Accepted 6 minutes ago Java

Jatin
May 08, 2023 21:17

Java

Runtime 7 ms Beats 66.73%
Memory 42.7 MB Beats 47.37%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```
class Solution {
    public List<String> readBinaryWatch(int turnedOn) {
        ArrayList<String> results = new ArrayList<String>();

        for (int hour=0; hour < 12; hour++) {
            for (int min=0; min < 60; min++) {
                if (Integer.bitCount(hour) + Integer.bitCount(min) == tur
                    if (min < 10) {
```

Problem 2: Word Ladder II

Code:

```
public List<List<String>> findLadders(String beginWord, String endWord, List<String> wordList) {


        HashMap<String,Integer> hm = new HashMap<>();

        Queue<Pair> q=new ArrayDeque<>();

        q.add(new Pair(beginWord,1));

        HashSet<String> visited=new HashSet<>();

        visited.add(beginWord);

        while(q.size()>0)

        {

            Pair rem=q.remove();

            String word=rem.word;

            int steps=rem.steps;

            hm.put(word,steps);

            if(endWord.equals(word)){

                break;

            }
            for(String trav:wordList)
            {
                if( visited.contains(trav)==false && isDiffOne(word,trav))
                {
                    visited.add(trav);
                    q.add(new Pair(trav,steps+1));
                }
            }
        }


    List<String> arl=new ArrayList<>();
    ans=new ArrayList<>();

    System.out.println(hm);

    if(hm.containsKey(endWord)==false)
        return ans;

    dfs(endWord,beginWord,wordList,hm,arl);

    return ans;

}
List<List<String>> ans;
```

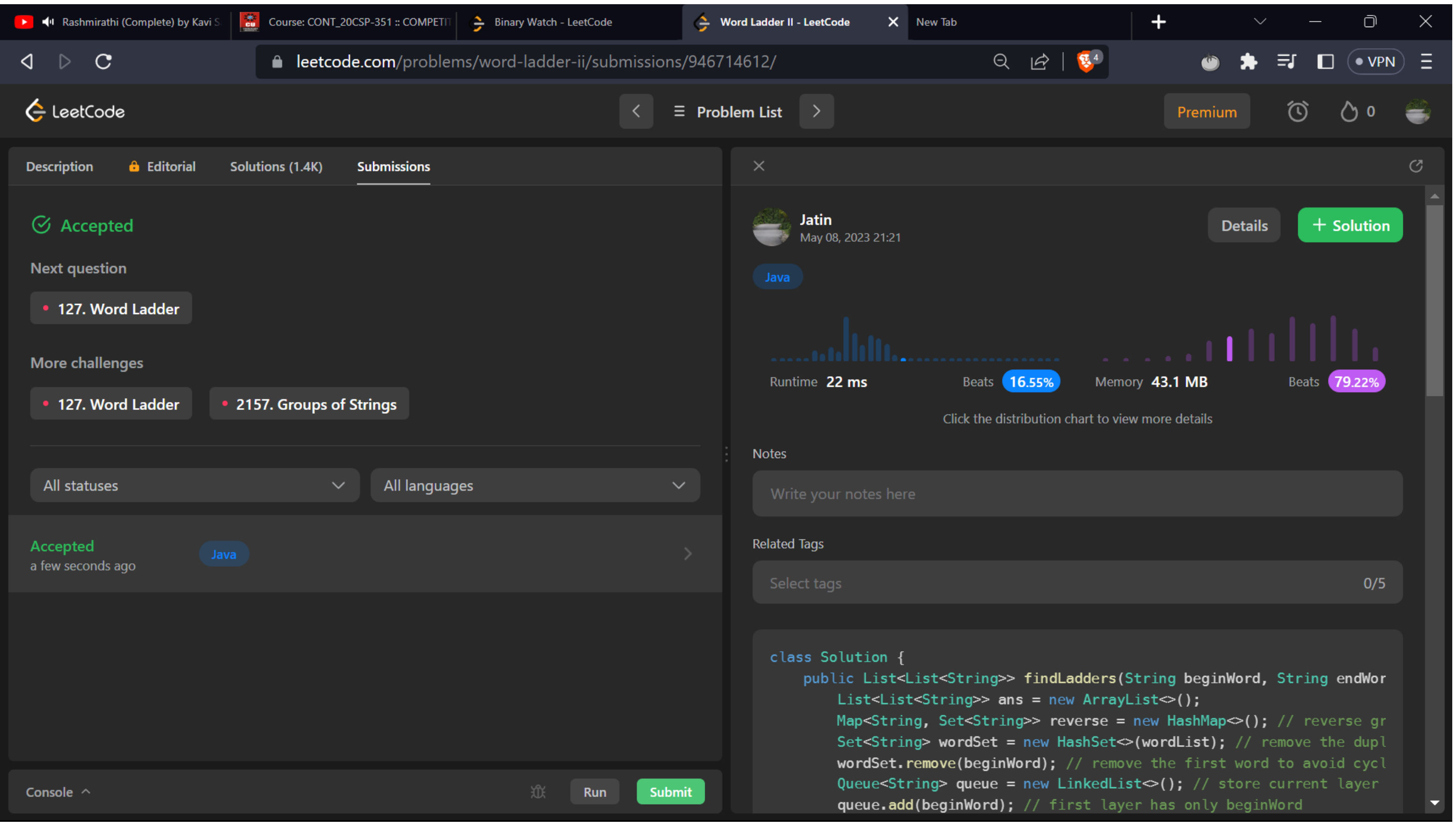
```
    public void dfs(String end, String begin, List<String> wordList, HashMap<String,
Integer> hm, List<String> arl)

    {



        if(end.equals(begin)){

            arl.add(0,begin);

            List<String> temp=new ArrayList(arl);

            ans.add(temp);

            arl.remove(0);

            return;

        }


        arl.add(0,end);

        for(String trav:hm.keySet())

        {

            if(hm.get(trav)<hm.get(end) && isDiffOne(end,trav))

                dfs(trav,begin,wordList,hm,arl);

        }

       arl.remove(0);

    }
```

```
public boolean isDiffOne(String s1, String s2)
{
    if(s1.equals(s2))return false;
    int count=0;
    for(int i=0;i<s1.length();i++){
        if(s1.charAt(i)!=s2.charAt(i))count++;
        if(count>=2)return false;
    }
    if(count==1)return true;
    return false;
}
```

```
class Pair{
    String word;
    int steps;
    Pair(String word, int steps){
        this.word=word;
        this.steps=steps;
    }
}
```

}

Output:


Rashmirathi (Complete) by Kavi S
Course: CONT_20CSP-351 :: COMPETI
Binary Watch - LeetCode
Word Ladder II - LeetCode
New Tab
leetcode.com/problems/word-ladder-ii/submissions/946714612/
VPN
LeetCode
Problem List
Premium
0
Description
Editorial
Solutions (1.4K)
Submissions

Accepted

Next question

127. Word Ladder

More challenges

127. Word Ladder

2157. Groups of Strings

Accepted Java
a few seconds ago

Jatin
May 08, 2023 21:21

Runtime 22 ms Beats 16.55% Memory 43.1 MB Beats 79.22%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```
class Solution {
    public List<List<String>> findLadders(String beginWord, String endWor
        List<List<String>> ans = new ArrayList<>();
        Map<String, Set<String>> reverse = new HashMap<>(); // reverse gr
        Set<String> wordSet = new HashSet<>(wordList); // remove the dupl
        wordSet.remove(beginWord); // remove the first word to avoid cycl
        Queue<String> queue = new LinkedList<>(); // store current layer
        queue.add(beginWord); // first layer has only beginWord
```