



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));  
  
                    }  
  
                }  
  
            }  
  
        }  
  
    }  
  
}
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}  
  
}  
  
return results;  
  
}  
  
}
```

Output:

The screenshot shows a web browser with multiple tabs. The active tab is 'Binary Watch - LeetCode'. The address bar shows the URL 'leetcode.com/problems/binary-watch/submissions/946712417/'. The page content is divided into two main sections. The left section, titled 'Submissions', shows a green checkmark and the text 'Accepted'. Below this, it says 'Next question' and lists '402. Remove K Digits' as the next challenge. There are also links to '17. Letter Combinations of a Phone Number' and '191. Number of 1 Bits'. The right section shows the submission details for user 'Jatin' on May 08, 2023 at 21:17. It displays performance metrics: Runtime 7 ms, Beats 66.73%, Memory 42.7 MB, and Beats 47.37%. There is a 'Details' button and a '+ Solution' button. Below the metrics, there is a 'Notes' section with a text input field. At the bottom, there is a 'Related Tags' section with a 'Select tags' dropdown. The code for the solution is displayed in a dark-themed editor, showing a Java class 'Solution' with a method 'readBinaryWatch' that returns a list of strings representing times.

LeetCode

Problem List

Premium

Description Editorial Solutions (1.1K) Submissions

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

Console Run Submit

Jatin May 08, 2023 21:17 Details + Solution

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) == turnedOn  
                    && (min < 10)) {  
                    results.add(hour + ":" + min);  
                }  
            }  
        }  
        return results;  
    }  
}
```



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 word: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;

    }

}
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

}

Output:

The screenshot shows a web browser with multiple tabs. The active tab is 'Word Ladder II - LeetCode'. The address bar shows the URL 'leetcode.com/problems/word-ladder-ii/submissions/946714612/'. The LeetCode interface is in dark mode. On the left, the 'Submissions' tab is selected, showing a green 'Accepted' status for the '127. Word Ladder' problem. Below this, there are filters for 'All statuses' and 'All languages', and a list of submissions. The submission shown is 'Accepted' and was made 'a few seconds ago' using 'Java'. On the right, the submission details for user 'Jatin' (May 08, 2023 21:21) are displayed. It shows a distribution chart, runtime of 22 ms (beats 16.55%), and memory of 43.1 MB (beats 79.22%). There is a 'Notes' section with a text input field and a 'Related Tags' section with a 'Select tags' dropdown. At the bottom, there is a 'Console' section with 'Run' and 'Submit' buttons. The code for the solution is displayed in a text area:

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