

## **HASHING SOLUTIONS**

#### Solution 1:

Using the concept of Horizontal Distance (HD), as discussed in Top View of a Binary Tree Question in the Trees chapter.

```
.mport java.util.*;
class Solution {
           this.hd = Integer.MAX_VALUE;
                               bottomViewHelper(Node
       if (!m.containsKey(hd)) {
           int[] p = m.get(hd);
           if (p[1] <= curr) {
               p[1] = curr;
```



```
p[0] = root.data;
        m.put(hd, p);
    bottomViewHelper(root.left, curr + 1, hd - 1, m);
    bottomViewHelper(root.right, curr + 1, hd + 1, m);
public static void printBottomView(Node root) {
    bottomViewHelper(root, 0, 0, m);
        System.out.print(val[0] + " ");
public static void main(String[] args) {
    root.left = new Node(8);
    root.right = new Node(22);
    root.left.left = new Node(5);
    root.right.left = new Node(4);
    root.right.right = new Node(25);
    root.left.right.right = new Node(14);
    printBottomView(root);
```



```
}
}
```

### Solution 2:

```
public int[] twoSum(int[] arr, int target) {
    Map<Integer, Integer> visited = new HashMap<>();
    for(int i = 0; i<arr.length; i++) {
        //diff = given target - number given at ith index
        int diff = target - arr[i];

        // check if found difference is present in the MAP list
        if(visited.containsKey(diff)) {
            //if difference in map matches with the ith index element in array
            return new int[] { i, visited.get(diff) };
        }

        //add arr element in map to match with future element if forms a pair
        visited.put(arr[i],i);
    }

    //if no matches are found
    return new int[] {0, 0};
}</pre>
```

### Solution 3:

```
public String frequencySort(String s) {
    HashMap<Character , Integer>map = new HashMap<>();
    for(int i=0;i<s.length();++i)
        map.put(s.charAt(i),map.getOrDefault(s.charAt(i),0)+1);

PriorityQueue<Map.Entry<Character,Integer>> pq =
    new PriorityQueue<>((a,b)->a.getValue()== b.getValue()?
        a.getKey()-b.getKey(): b.getValue()-a.getValue());

for(Map.Entry<Character,Integer>e:map.entrySet()) pq.add(e);
```



```
StringBuilder res = new StringBuilder();
while (pq.size()!=0) {
    char ch = pq.poll().getKey();
    int val = map.get(ch);
    while (val!=0) {
        res.append(ch);
        val---;
    }
}
return res.toString();
}
```

# APNA COLLEGE