01/08/2024

[**1208. Get Equal Substrings Within Budget**](https://leetcode.com/problems/get-equal-substrings-within-budget/)

<https://leetcode.com/problems/get-equal-substrings-within-budget/description/>

class Solution {

    public int equalSubstring(String s, String t, int maxCost) {

        int i,j,n;

        int currCost=0;

        int res=0;

        n=s.length();

        for(i=0,j=0;j<n;){

            currCost+=Math.abs(s.charAt(j)-t.charAt(j));

            while(i<n && currCost>maxCost){

                currCost-=Math.abs(s.charAt(i)-t.charAt(i));

                i++;

            }

            j++;

            res=Math.max(res,j-i);

        }

        return res;

    }

}

[Minimum Remove to Make Valid Parentheses](https://leetcode.com/problems/minimum-remove-to-make-valid-parentheses/)

<https://leetcode.com/problems/minimum-remove-to-make-valid-parentheses/description/>

class Solution {

    public String minRemoveToMakeValid(String s) {

        char ar[]=s.toCharArray();

        int i,n;

        n=ar.length;

        Stack<Integer> stk=new Stack<>();

        for(i=0;i<n;i++){

            if(ar[i]=='(')stk.push(i);

            else if(ar[i]==')'){

                if(stk.empty() || ar[stk.peek()]!='(')stk.push(i);

                else stk.pop();

            }

        }

        while(stk.size()!=0){

            ar[stk.peek()]='!';

            stk.pop();

        }

        String res="";

        for(i=0;i<n;i++){

            if(ar[i]!='!')res+=ar[i];

        }

        return res;

    }

}

02/08/2024

Que.1

[Minimum Swaps to Group All 1's Together II](https://leetcode.com/problems/minimum-swaps-to-group-all-1s-together-ii/)

<https://leetcode.com/submissions/detail/1341692553/>

class Solution {

    public int minSwaps(int[] nums) {

        int i,j,k,n;

        n=nums.length;

        k=0;

        for(j=0;j<n;j++)if(nums[j]==1)k++;

        int currCont=0;

        int maxCont=0;

        for(j=0;j<k;j++)if(nums[j]==1)currCont++;

        if(currCont==k)return 0;

        maxCont=Math.max(maxCont,currCont);

        i=0;

        while(i<n){

            if(nums[i]==1)currCont--;

            if(nums[j]==1)currCont++;

            i++;

            j++;

            j=j%n;

            maxCont=Math.max(maxCont,currCont);

        }

         return k-maxCont;

    }

}

Que. 2

[Adding Spaces to a String](https://leetcode.com/problems/adding-spaces-to-a-string/)

<https://leetcode.com/problems/adding-spaces-to-a-string/description/>

class Solution {

    public String addSpaces(String s, int[] spaces) {

        int m = s.length(), n = spaces.length;

        StringBuilder ans = new StringBuilder();

        for (int i = 0, j = 0; i < m; i++) {

            if (j < n && spaces[j] == i) {

                j++;

                ans.append(' ');

            }

            ans.append(s.charAt(i));

        }

        return ans.toString();

    }

}

03-08-2024

Que1.

[**The k-th Lexicographical String of All Happy Strings of Length n**](https://leetcode.com/problems/the-k-th-lexicographical-string-of-all-happy-strings-of-length-n/)

<https://leetcode.com/problems/the-k-th-lexicographical-string-of-all-happy-strings-of-length-n/description/>

class Solution {

    List<String> list=new ArrayList<>();

    StringBuilder sb = new StringBuilder();

    public String getHappyString(int n, int k) {

        backtrack(n,' ');

        Collections.sort(list);

        if(list.size()<k) return "";

        else return list.get(k-1);

    }

    void backtrack(int n, char prev){

        if(sb.length()==n){

            list.add(sb.toString());

            return;

        }

        else{

            for(char  ch='a';ch<='c';ch++){

              if(ch!=prev){

                sb.append(ch);

                backtrack(n,ch);

                sb.delete(sb.length()-1,sb.length());

              }

            }

        }

    }

}

Que.2[**Make Two Arrays Equal by Reversing Subarrays**](https://leetcode.com/problems/make-two-arrays-equal-by-reversing-subarrays/)

https://leetcode.com/problems/make-two-arrays-equal-by-reversing-subarrays/description/

class Solution {

    public boolean canBeEqual(int[] target, int[] arr) {

        HashMap<Integer,Integer> mp=new HashMap<>();

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

            if(mp.containsKey(target[i])){

                mp.put(target[i],mp.get(target[i])+1);

            }

            else{

                mp.put(target[i],1);

            }

        }

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

            if(mp.containsKey(arr[i])==false)return false;

            mp.put(arr[i],mp.get(arr[i])-1);

            if(mp.get(arr[i])==0)mp.remove(arr[i]);

        }

        return true;

    }

}

04/08/2024

Que 1 . [**. Range Sum of Sorted Subarray Sums**](https://leetcode.com/problems/range-sum-of-sorted-subarray-sums/)

<https://leetcode.com/problems/range-sum-of-sorted-subarray-sums/>

class Solution {

    public int rangeSum(int[] nums, int n, int left, int right) {

        ArrayList<Integer> res=new ArrayList<>();

        int i,j;

        int sum=0;

        int ans=0;

        int mod=1000000000+7;

        for(i=0;i<n;i++){

            sum=0;

            for(j=i;j<n;j++){

                sum=(sum%mod+nums[j]%mod)%mod;

                   res.add(sum);

            }

        }

        Collections.sort(res);

        for(i=left-1;i<right;i++){

            ans=(ans%mod+(res.get(i))%mod)%mod;

        }

        return ans;

    }

}

Que 2. [**Decode String**](https://leetcode.com/problems/decode-string/)

<https://leetcode.com/problems/decode-string/description/>

class Solution {

    public String decodeString(String s) {

        Stack<Character> stk=new Stack<>();

        StringBuilder str=new StringBuilder();

        int i;

        for(i=0;i<s.length();i++){

            if(s.charAt(i)==']'){

                StringBuilder temp=new StringBuilder();

                while(!stk.empty() && stk.peek()!='['){

                    temp.append(stk.peek());

                    stk.pop();

                }

                stk.pop();

                temp.reverse();

                StringBuilder nums=new StringBuilder();

                while(!stk.empty() && (stk.peek()>='0' && stk.peek()<='9')){

                    nums.append(stk.peek());

                    stk.pop();

                }

                nums.reverse();

                int n=Integer.parseInt(nums.toString());

                String r="";

                while(n!=0){

                    r=r+temp;

                    n--;

                }

                for(int ind=0;ind<r.length();ind++)stk.push(r.charAt(ind));

            }

            else{

                stk.push(s.charAt(i));

            }

        }

        StringBuilder res=new StringBuilder();

        while(!stk.empty()){

            res.append(stk.peek());

            stk.pop();

        }

        res.reverse();

        return res.toString();

    }

}

5/08/2024

Que 1. [**. Kth Distinct String in an Array**](https://leetcode.com/problems/kth-distinct-string-in-an-array/)

<https://leetcode.com/problems/kth-distinct-string-in-an-array/description/>

class Solution {

    public String kthDistinct(String[] arr, int k) {

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

        int i;

        for(i=0;i<arr.length;i++){

            if(mp.containsKey(arr[i])){

                mp.put(arr[i],mp.get(arr[i])+1);

            }

            else{

                mp.put(arr[i],1);

            }

        }

        for(i=0;i<arr.length;i++){

            if(mp.get(arr[i])==1)k--;

            if(k==0)return arr[i];

        }

        return "";

    }

}

Que 2 . [**. Car Pooling**](https://leetcode.com/problems/car-pooling/)

<https://leetcode.com/problems/car-pooling/description/>

class Solution {

    public boolean carPooling(int[][] trips, int capacity) {

        int ar[]=new int[1001];

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

            ar[trips[i][1]]+=trips[i][0];

            ar[trips[i][2]]-=trips[i][0];

        }

        for(int i=0;capacity>=0 && i<ar.length;i++){

            capacity-=ar[i];

        }

         return capacity>=0;

    }

}

06/08/2024

Que 1 . [**Minimum Changes To Make Alternating Binary String**](https://leetcode.com/problems/minimum-changes-to-make-alternating-binary-string/)

<https://leetcode.com/problems/minimum-changes-to-make-alternating-binary-string/description/>

class Solution {

    public int minOperations(String s) {

       int res=Integer.MAX\_VALUE;

       int curr=0;

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

            if(i%2==0){

                if(s.charAt(i)=='0')curr++;

            }

            else{

                if(s.charAt(i)=='1')curr++;

            }

        }

        res=Math.min(res,curr);

        curr=0;

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

            if(i%2==0){

                if(s.charAt(i)=='1')curr++;

            }

            else{

                if(s.charAt(i)=='0')curr++;

            }

        }

        return Math.min(res,curr);

    }

}

Que 2 . [**Maximum Number of Words You Can Type**](https://leetcode.com/problems/maximum-number-of-words-you-can-type/)

<https://leetcode.com/problems/maximum-number-of-words-you-can-type/description/>

class Solution {

    public int canBeTypedWords(String text, String brokenLetters) {

        String [] textArray=text.split(" ");

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

        int i,j;

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

            hashMap.put(brokenLetters.charAt(i),1);

        }

        int res=0;

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

            String str=textArray[i];

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

                if(hashMap.containsKey(str.charAt(j)))break;

            }

            if(j==str.length())res++;

        }

        return res;

    }

}