***Week – 3 (14.04.2021 – 18.04.2021)***

***LEETCODE CONTEST – 17.04.2021***

Rank - 5685/10095

Mark – 7/18

Link - <https://leetcode.com/contest/biweekly-contest-50>

1. ***Minimum Operations to Make the Array Increasing:***

class Solution {

public:

int minOperations(vector<int>& nums) {

int i, op = 0;

for(i=0; i<nums.size()-1; i++)

{

if(nums[i] > nums[i+1])

{

op = op + nums[i] - nums[i+1] + 1;

nums[i+1] = nums[i]+1;

}

else if(nums[i] == nums[i+1])

{

nums[i+1] = nums[i+1] + 1;

op++;

}

else

{

while(nums[i] >= nums[i+1])

{

nums[i+1] = nums[i+1] + 1;

op++;

}

}

}

return op;

}

};

1. ***Queries on Number of Points Inside a Circle:***

class Solution {

public:

vector<int> countPoints(vector<vector<int>>& points, vector<vector<int>>& queries) {

vector<int> res(queries.size());

int i,j;

double d;

for(i=0; i<res.size(); i++)

res[i] = 0;

for(i=0; i<queries.size(); i++)

{

for(j=0; j<points.size(); j++)

{

d = sqrt(pow((queries[i][0] - points[j][0]),2) + pow((queries[i][1] - points[j][1]),2));

if(pow(d,2) <= pow(queries[i][2],2))

res[i] = res[i]+1;

}

}

return res;

}

};

1. ***Maximum XOR for Each Query:***

class Solution {

public:

vector<int> getMaximumXor(vector<int>& nums, int maximumBit) {

int XOR = 0, i;

vector<int> res(nums.size());

for(auto& num : nums) XOR ^= num;

res[0] = XOR = XOR ^ (1<<maximumBit) - 1;

for(i=1; i<nums.size(); i++)

res[i] = XOR = XOR ^ nums[nums.size() - i];

return res;

}

};

1. ***Minimum Number of Operations to Make String Sorted:***

class Solution {

public:

long long mod =1000000007;

long long modpow(long long x,long long n,int M)

{

long long result=1;

while(n>0)

{

if(n&1){

result=(result \* x)%M;

}

x=(x\*x)%M;

n=n>>1;

}

return result;

}

int makeStringSorted(string s) {

unordered\_map<int,int>m;

int n=s.length();

vector<long long> fact(s.length() + 1, 1ll);

for (int i = 1; i <= s.size(); i++){

fact[i] = (fact[i - 1] \* i)%mod;

}

long long ans=0;

for(int i=n-1;i>=0;i--){

long long count=0;

m[s[i]-'a']++;

long long res=1;

for(auto x:m){

res= (res\*fact[x.second])%mod;

if(x.first<(s[i]-'a')){

count+=x.second;

}

}

ans=ans%mod+ (((count\*(fact[n-i-1]))%mod)\*(modpow(res,mod-2,mod))%mod)%mod;

ans=ans%mod;

}

return ans%mod;

}

};