DSA_DAY7

https://leetcode.com/problems/jump-game/

```
class Solution {
public:
    bool canJump(vector<int>& nums) {
        int maxi = 0;
        if(nums[0]==0 && nums.size()==1) return true;
        for(int i=0;i<nums.size();i++){

            if(maxi>=nums.size()) return true;
            if(nums[i]==0){
                if(maxi<=i+1) return false;
            }
            maxi = max(maxi,i+1+nums[i]);
        }
        return true;
    }
};
```

https://leetcode.com/problems/jump-game-ii/description/

```
class Solution {
public:
    int f(int i,vector<int>& nums,vector<int>& dp){
        if(i>=nums.size()-1) return 0;
        if(dp[i]!=-1) return dp[i];
        int p=INT_MAX;
        for(int k=1;k<=nums[i];k++){
            int c = f(i+k,nums,dp);
            if(c!=INT_MAX) p=min(p,1+ c);
        }
        dp[i] = p;
```

```
        return p;
    }
    int jump(vector<int>& nums) {
        vector<int> dp(nums.size(),INT_MAX);
        int n=nums.size();
        dp[n-1]=0;
        for(int i = nums.size()-2;i>=0;i--){
            int p = INT_MAX;
            for(int j=1;j<=nums[i];j++){
                if(i+j<n && dp[i+j]!=INT_MAX ) p = min(p,1+dp

            }
            dp[i]=p;
        }
        return dp[0];
        // return f(0,nums,dp);
    }
};
```

https://leetcode.com/problems/number-of-islands/description/

```
class Solution {
public:
    void dfs(vector<vector<char>>& grid, vector<vector<int>>&
             int jj) {
        int n = grid.size(), m = grid[0].size();
        queue<pair<int, int>> q;
        q.push({ii, jj});
        vis[ii][jj] = 1;
        vector<int> r = {-1, 0, 1, 0}, c = {0, -1, 0, 1};
        while (!q.empty()) {
            int i = q.front().first, j = q.front().second;

            for (int k = 0; k < 4; k++) {
                int nr = i + r[k], nc = j + c[k];
                if (nr >= 0 && nr < n && nc >= 0 && nc < m &&
                    grid[nr][nc] == '1') {
```

```
                    q.push({nr, nc});
                    vis[nr][nc] = 1;
                }
            }
            q.pop();
        }
    }

    int numIslands(vector<vector<char>>& grid) {
        int n = grid.size(), m = grid[0].size();
        vector<vector<int>> vis(n, vector<int>(m, 0));
        int ans = 0;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                if (grid[i][j] == '1' && !vis[i][j]) {
                    ans++;
                    dfs(grid, vis, i, j);
                }
            }
        }
        return ans;
    }
};
```

https://leetcode.com/problems/group-anagrams/description/

```
class Solution {
#define pb push_back
public:
    vector<vector<string>> groupAnagrams(vector<string>& strs

        unordered_map<string,vector<string>> ans;
        for(string  s: strs){
            string tmp = s;
            sort(tmp.begin(),tmp.end());
            ans[tmp].pb(s);
        }
```

```
        vector<vector<string>> g;
        for(auto i:ans){
            g.pb(i.second);
        }
        return g;

        }
    };
```

https://www.geeksforgeeks.org/problems/searching-an-element-in-a-sorted-array-ternary-search--141631/1

```
class Solution{
    public:
    // Function to find element in sorted array
    // arr: input array
    // N: size of array
    // K: element to be searche
    int ternarySearch(int arr[], int N, int K)
    {
       // Your code here
       int  l=0,h=N-1;
       while(l<=h){
           int m1 = l+(h-l)/3,m2= h - (h-l)/3;
           if(arr[m1] == K)  return 1;
           else if(arr[m2] == K) return 1;
           else if(arr[m1]>K) h=m1-1;
           else if(arr[m2]<K) l=m2+1;
           else {
               l=m1+1;
               h=m2-1;
           }
       }
       return -1;
```

```
    }
};
```

https://www.geeksforgeeks.org/interpolation-search/

```
arr = [1,2,3,4,5,6,7,8]
val = 9
l,h,cnt=0,7,1
while l<=h and val>=arr[l] and val<=arr[h]:
    probe = l + (h-l)*(val-arr[l])//(arr[h]-arr[l])
    if(arr[probe]==val) :
        print("Found")
        cnt-=1
        break
    elif arr[probe]>val: h=probe-1
    else: l=probe+1
if(cnt):
    print("Not Found")
```

https://leetcode.com/problems/3sum-closest/description/

```
class Solution {
public:
    int threeSumClosest(vector<int>& nums, int t) {
            sort(nums.begin(),nums.end());
            int ans = nums[0]+nums[1]+nums[2],n=nums.size();
            for(int i=0;i<n-2;i++){
                if(i>0 && nums[i]==nums[i-1]) continue;
                int l=i+1,r=n-1;
                while(l<r){
                    int sume = nums[l]+nums[r]+nums[i];
                    if(abs(sume-t) < abs(ans-t) ) ans = sume;
                    if(sume == t) return t;
                    if(sume>t) r--;
                    else l++;
                }
```

```
            }
            return ans;



    }
};
```

https://leetcode.com/problems/decode-ways/

```
class Solution {
public:
    int f(int i, string s, vector<int>& dp) {
        int n = s.length();
        if (i >= n)
            return 1;
        if (dp[i] != -1)
            return dp[i];
        int p = 0, p1 = 0;
        int c1 = s[i] - 48;
        if (c1 > 0)
            p = f(i + 1, s, dp);
        if (i + 1 < n) {
            int c2 = s[i + 1] - 48;
            if (c1 != 0) {
                if ((c1 == 1) || (c1 == 2 && c2 <= 6))
                    p1 = f(i + 2, s, dp);
                // cout<<s[i]<<s[i+1]<<" "<<c3<<endl;
            }
        }
        // cout<<p<<" "<<p1<<endl;
        return dp[i] = p + p1;
    }
    int numDecodings(string s) {
        int n = s.length();
        vector<int> dp(n + 2, 0);
        dp[n]=1;
```

```
        dp[n+1]=1;
        for (int i = n - 1; i >= 0; i--) {
            int p1 = 0, p2 = 0;
            int c1 = s[i] - 48;
            if (c1 > 0)
                p1 = dp[i+1];
            if (i + 1 < n) {
                int c2 = s[i + 1] - 48;
                if (c1 != 0) {
                    if ((c1 == 1) || (c1 == 2 && c2 <= 6))
                        p2 = dp[i+2];
                    // cout<<s[i]<<s[i+1]<<" "<<c3<<endl;
                }
            }
            dp[i] = p1+p2;
        }
        return dp[0];
        return f(0, s, dp);
    }
};
```

https://www.geeksforgeeks.org/problems/merge-sort/1

```
class Solution {
  public:
  #define pb push_back
    void merge(vector<int>& arr,int low,int mid,int high){
        int i=low,j=mid+1;
        vector<int> tmp;
        while(i<=mid && j<=high){
            if(arr[i]<=arr[j])
                tmp.pb(arr[i++]);
            else
                tmp.pb(arr[j++]);
        }
        while(i<=mid) tmp.pb(arr[i++]);
        while(j<=high) tmp.pb(arr[j++]);
```

```
        for(int k=low;k<=high;k++){
            arr[k]=tmp[k-low];
        }
    }



    void mergeSort(vector<int>& arr, int low, int high) {
        // code here
        if(low<high){
            int mid = (low+high)/2;
            mergeSort(arr,low,mid);
            mergeSort(arr,mid+1,high);
            merge(arr,low,mid,high);
        }

    }
};
```