**Day-23**

**Problem1**-   
[**22. Generate Parentheses**](https://leetcode.com/problems/generate-parentheses/)

class Solution {

public:

void solve(int open,int close, string s, vector<string>&v)

{

if(open==0 && close ==0)

{v.push\_back(s);

return;

}

if(open!=0)

{

string op= s;

op.push\_back('(');

solve(open-1,close,op,v);

}

if(close>open)

{

string op1=s;

op1.push\_back(')');

solve(open,close-1,op1,v);

}

}

vector<string> generateParenthesis(int n) {

int open=n;

int close=n;

string s="";

vector<string>v;

solve(open,close,s,v);

return v;

}

};

**Problem2**-   
[**71. Simplify Path**](https://leetcode.com/problems/simplify-path/)

class Solution {

public:

string simplifyPath(string path) {

int n=path.size();

stack<string>s;

int i=0;

while(i<path.length())

{

if(path[i]=='/')

{i++;

continue;}

string t="";

int j=i;

while(j<path.length() && path[j]!='/')

{

t+=path[j];

j++;

}

i=j;

if(t==".")

continue;

else if(t==".."&& s.empty())continue;

else if(t==".."&& !s.empty())

s.pop();

else

s.push(t);

}

string ans = "";

while(!s.empty()) {

ans = "/" + s.top() + ans;

s.pop();

}

if(ans.length() == 0) return "/";

return ans;

}

}; }

}

vector<vector<int>> uniquePerms(vector<int> &arr ,int n) {

// code here

vector<int>p;

vector<vector<int>>ans;

bool v[n];

for(int i=0;i<n;i++)

{

v[i]=false;

}

sort(arr.begin(),arr.end());

solve(0,n,p,ans,arr,v);

return ans;

}

};

//{ Driver Code Starts.

int main() {

int t;

cin >> t;

while (t--) {

int n;

cin>>n;

vector<int> arr(n);

for(int i=0 ; i<n ; i++)

cin>>arr[i];

Solution ob;

vector<vector<int>> res = ob.uniquePerms(arr,n);

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

{

for(int j=0; j<n; j++)

{

cout<<res[i][j]<<" ";

}

cout<<"\n";

}

}

return 0;

}

// } Driver Code Ends

**Problem 3**- **Permutations in arra**

**//{ Driver Code Starts**

**#include <bits/stdc++.h>**

**using namespace std;**

**// } Driver Code Ends**

**class Solution {**

**public:**

**bool isPossible(int k, vector<int> &a, vector<int> &b) {**

**// Your code goes here**

**int n=a.size();**

**sort(a.begin(),a.end());**

**sort(b.begin(),b.end());**

**for(int i=0;i<n;i++)**

**{**

**if(a[i]+b[n-1-i]<k)**

**return false;**

**}**

**return true;**

**}**

**};**

**//{ Driver Code Starts.**

**int main() {**

**int t;**

**cin >> t;**

**while (t--) {**

**int k;**

**cin >> k;**

**cin.ignore();**

**string s;**

**getline(cin, s);**

**stringstream ss(s);**

**vector<int> arr1, arr2;**

**int num;**

**while (ss >> num) {**

**arr1.push\_back(num);**

**}**

**getline(cin, s);**

**ss.clear();**

**ss.str(s);**

**while (ss >> num) {**

**arr2.push\_back(num);**

**}**

**Solution ob;**

**bool ans = ob.isPossible(k, arr1, arr2);**

**if (ans) {**

**cout << "true" << endl;**

**} else {**

**cout << "false" << endl;**

**}**

**}**

**return 0;**

**}**

**// } Driver Code Ends**