'''

Ms.Akhila is a famous mathematician in a school, one day she decided to

send her students to Maths Olympiad competetion

To do that MS, Akhila called N number of students and ask them to stand in line.

Each student is having distinct grade according to his/her perfomences based

on the exams she conducted so far.

In order to send the students to Olympiad, Ms. Akhila has to form a team of

3 students among N students from the line with following conditions:

-> Pick any 3 students with index(p,q,r) with

grade(grade[p],grade[q],grade[r]).

->A team is consider as valid if:

- ( grade[p] < grade[q] < grade[r] ) or ( grade[p] > grade[q] > grade[r] )

where (0 <= p < q < r < N).

You are given grades[], grades of N students in the line.

Your task is to return the number of teams Ms.Akhila can form with

given conditions (students can be part of multiple teams).

Input Format:

-------------

Line-1: AN integer N, number of students.

Line-2: N space seperated integers, grades[].

Output Format:

--------------

An integer, number of teams can be formed.

Sample Input-1:

---------------

5

3 6 4 5 2

Sample Output-1:

----------------

3

Explanation:

------------

Ms Akhila can form three teams given the conditions.

(3,4,5), (6,4,2), (6,5,2).

Sample Input-2:

---------------

5

1 2 3 4 5

Sample Output-2:

----------------

10

'''

def fun(l,ans,j,count):

if(len(ans)==3 and ((ans[0]<ans[1]<ans[2]) or (ans[0]>ans[1]>ans[2]))):

count[0]+=1

for i in range(j,len(l)):

ans.append(l[i]);

fun(l,ans,i+1,count)

ans.pop()

n=int(input())

l=list(map(int,input().split()))

count=[0]

# return count

a=fun(l,[],0,count)

print(count[0])

Indian Army setup some military-camps, sitauted at random places at LAC in Galwan.

There exist a main base camp connected with other base camps as follows:

Each military-camp is connected with atmost two other military-camps.

Each military-camp will be identified with an unique ID,(an integer).

To safeguard all the military-camps, Govt of India planned to setup protective

S.H.I.E.L.D. Govt of India ask your help to build the S.H.I.E.L.D that should

enclose all the militar-camps.

You are given the IDs of the military-camps as binary tree.

Your task is to find and return the military camp IDs, those are on the edge of

the S.H.I.E.L.D in anti-clockwise order.

Implement the class Solution:

1. public List<Integer> compoundWall(BinaryTreeNode root): returns a list.

NOTE:

'-1' in the input IDs indicates no military-camp (NULL).

Input Format:

-------------

space separated integers, military-camp IDs.

Output Format:

--------------

Print all the military-camp IDs, which are at the edge of S.H.I.E.L.D.

Sample Input-1:

---------------

5 2 4 7 9 8 1

Sample Output-1:

----------------

[5, 2, 7, 9, 8, 1, 4]

Sample Input-2:

---------------

11 2 13 4 25 6 -1 -1 -1 7 18 9 10

Sample Output-2:

----------------

[11, 2, 4, 7, 18, 9, 10, 6, 13]

import java.util.\*;

/\*

//TreeNode Structure for Your Reference..

class Node{

public int data;

public Node left, right;

public Node(int data){

this.data = data;

left = null;

right = null;

}

}

\*/

class Solution {

public List<Integer> compoundWall(Node root) {

// Implement Your Code here..

List<Integer>l=new ArrayList<>();

l.add(root.data);

leftboundary(root.left,l);

leaves(root.left,l);

// System.out.print(l);

leaves(root.right,l);

rightboundary(root.right,l);

return l;

}

public void leftboundary(Node root,List<Integer>l){

if(root==null){

return;

}

if((root.left==null && root.right==null)){

return;

}

if(root.left==null){

leftboundary(root.right,l);

}

else if(root.left!=null){

l.add(root.data);

leftboundary(root.left,l);

}

}

public void leaves(Node root,List<Integer>l){

if(root==null){

return;

}

if((root.left==null && root.right==null) ){

l.add(root.data);

return;

}

leaves(root.left,l);

leaves(root.right,l);

}

public void rightboundary(Node root,List<Integer>l){

if( root==null){

return;

}

if((root.left==null && root.right==null) ){

return;

}

if(root.right==null){

rightboundary(root.left,l);

l.add(root.data);

}

if(root.right!=null){

rightboundary(root.right,l);

l.add(root.data);

}

}

}

Mr Venu working with statements.

Each statement consists of only lowercase alphabets and whitespaces,

and each statement neither starts nor ends with a whitespace

and each word in the statement separated by a single whitespace only.

Mr Venu's task is to perform the join operation on these statements:

The joining operations is performed as follows:

- If the last word of a statement-1 and first word of statement-2 are same,

then you can join the statements-1 & statement-2 and

the joining word will appears once in the result statement.

- It is not allowed to join the statements with itself.

Note that the order of joining the statements matters.

You will be given N statements.

Your task is to help Mr Venu to join the statements, and

return them as a list of distinct statements in lexicographic order.

Input Format:

-------------

Line-1: An integer N, number of statements.

Next N lines: Single line String, the statement with one or more words.

Output Format:

--------------

Print the list of strings, the joined statements.

Sample Input-1:

---------------

3

write the code

code in java

java code

Sample Output-1:

----------------

[code in java code, java code in java, write the code in java]

Sample Input-2:

---------------

4

programming language

language is java

language is python

java is based on oops concepts

Sample Output-2:

----------------

[language is java is based on oops concepts,

programming language is java,

programming language is python]

Sample Input-3:

---------------

3

A B C

B C A

C A B

Sample Output-3:

----------------

[A B C A B, B C A B C, C A B C A]

NOTE: You have to print the list in a single line.

#include<bits/stdc++.h>

using namespace std;

int main() {

int n;

cin >> n;

cin.ignore(); // Ignore newline character

vector<string> statements(n);

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

getline(cin, statements[i]);

}

vector<string> result;

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

string lastWord = statements[i].substr(statements[i].find\_last\_of(" ") + 1);

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

if (i != j) {

string firstWord = statements[j].substr(0, statements[j].find(" "));

if (lastWord == firstWord) {

string joinedStatement = statements[i] + statements[j].substr(firstWord.size());

result.push\_back(joinedStatement);

}

}

}

}

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

cout << "[";

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

cout << result[i];

if (i < result.size() - 1) {

cout << ", ";

}

}

cout << "]" << endl;

return 0;

}

Vicky, an expert in gaming software, lives in Australia and never compromises

in his life, a principle that he adheres to since his childhood. After

long time he returns to India to establish a start-up gaming company.

He meets his uncle Jalsa Kishore and Vicky tells about his plan.

Jalsa Kishore came up with an idea for Vicky with a new game.

He draws a grid with combination of 0’s and 1’s of size P X Q, where

we can reorganize the columns of the grid in any order.

Jalsa Kishore says that build a grid such that we get the area of

the greatest subunits within the grid where every unit of the sub grid is 1

after reorganizing the columns.

Now it’s your aim to write a method which prints the area of

the greatest subunits within the grid according to idea of Jalsa Kishore.

Input Format:

-------------

Line-1: Two integers P and Q, size of the grid.

Next P lines: Q separated integers, either 0 or 1.

Output Format:

--------------

Print an integer, area of the greatest subunits.

Sample Input-1:

---------------

3 4

1 0 1 0

0 1 1 1

0 1 0 1

Sample Output-1:

----------------

4

Explanation:

----------

After Reorganizing the columns, the grid looks like as follows:

1 0 0 1

0 1 1 1

0 1 1 0

Sample Input-2:

---------------

1 5

1 0 1 0 1

Sample Output-2:

----------------

3