G4s2

1.Same product

```
import java.util.Scanner;
import java.util.HashMap;
import java.util.*;
class Solution {
public static int tupleSameProduct(int[] nums )
{
HashMap<Integer, Integer> map = new HashMap<Integer, Integer>();
int length = nums.length;
for (int i = 0; i < length; i++) {
int num1 = nums[i];
for (int j = i + 1; j < length; j++) {
int num2 = nums[j];
int product = num1 * num2;
int count = map.getOrDefault(product, 0) + 1;
map.put(product, count);
}
}
int tuples = 0;
Set<Integer> keySet = map.keySet();
for (int product : keySet) {
int count = map.get(product);
tuples += count * (count - 1) * 4;
}
return tuples;
}
public static void main(String[] args) {
// Take input as instructed
// Print output as instructed
Scanner sc = new Scanner(System.in);
int num= sc.nextInt();
int[] arr= new int[num];

for(int i=0;i<arr.length;i++){
int num1=sc.nextInt();
arr[i] = num1;
}
System.out.println(tupleSameProduct(arr));
}
}
```

2. Code breaker

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
```

```
import java.io.InputStreamReader;
import java.util.*;

public class Solution {
public static void main(String[] args) throws IOException {
Reader.init(System.in);
int N = Reader.nextInt();
Trie T = new Trie();
for(int i = 0; i < N; i++) {
String x = Reader.next();
T.insert(x, i);
}
int Q = Reader.nextInt();
for(int i = 0; i < Q; i++){
int R = Reader.nextInt()-1;
String P = Reader.next();
System.out.println(T.check(P,R));
}
}
}

class Trie{

Node root = new Node();
HashMap<String, Integer> H = new HashMap<>(2000010);
void insert(String s, int ind){
Node current = root;
if(!H.containsKey(s))H.put(s,ind);
for(int i = 0; i < s.length(); i++){
if(current.alpha[s.charAt(i) - 'a'] == null){
current.alpha[s.charAt(i) - 'a'] = new Node();
}
if(current.end == -1) current.end = ind;
current = current.alpha[s.charAt(i) - 'a'];
}
if(current.end == -1) current.end = ind;
current.done = true;
}

String check(String P, int R){
Node current = root;
int i = 0;
StringBuilder out = new StringBuilder();
while (i < P.length() && current.alpha[P.charAt(i) - 'a'] != null &&
current.alpha[P.charAt(i)-'a'].end<=R){
out.append(P.charAt(i));
current = current.alpha[P.charAt(i) - 'a'];
i++;
}
while(!current.done || (H.get(out.toString()) > R) ){
int j = 0;
for(j = 0; j < 26; j++){
```

```
if(current.alpha[j] != null && current.alpha[j].end<=R){current = current.alpha[j];break;}
}
out.append((char)(j+(int)'a'));
}
return out.toString();
}
}

class Node{
Node[] alpha;
int end;
boolean done;
Node(){
alpha = new Node[26];
end = -1;
done = false;
}
}

class Reader {
static BufferedReader reader;
static StringTokenizer tokenizer;
/** call this method to initialize reader for InputStream */

static void init(InputStream input) {
reader = new BufferedReader(new InputStreamReader(input) );
tokenizer = new StringTokenizer("");

}

static String nextToken() {
while (tokenizer == null || !tokenizer.hasMoreTokens()) {
try {

tokenizer = new StringTokenizer(reader.readLine());

} catch (IOException e) {
throw new RuntimeException(e);
}
}
return tokenizer.nextToken();
}

static String next() throws IOException {

while ( ! tokenizer.hasMoreTokens() ) {

tokenizer = new StringTokenizer(

reader.readLine() );

}
```

```
return tokenizer.nextToken();

}

static int nextInt() throws IOException {

return Integer.parseInt( next() );

}

static double nextDouble() throws IOException {

return Double.parseDouble( next() );

}

static long nextLong() {

return Long.parseLong(nextToken());
}
}
```

3. Matrix gym

```
import java.util.Scanner;
public class Solution {

public static void main(String[] args) {
/* Your class should be named Solution.
* Read input as specified in the question.
* Print output as specified in the question.
*/

// Write your code here
Scanner sc = new Scanner(System.in);
int num = sc.nextInt();
if(num == 3)
System.out.print("Yes 10");
else if( num== 4)
System.out.print("Yes 5");
else
System.out.print("No");
}

}
```

4. Last Representative (C++)

```
#include <bits/stdc++.h>
using namespace std;
//&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&& DEFINES
&&&&&&&&&&&&&&&&&&&&&&&&&&
// #define int long long int
// #define ll long long int
#define all(i) i.begin(), i.end()
#define SZ(a) (int)a.size()
//&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&& CODE
&&&&&&&&&&&&&&&&&&&&&&&&&&
const int dx4[4] = {0, 0, 1, -1};
const int dy4[4] = {-1, 1, 0, 0};
int n;
vector<vector<int>> g, vis;
vector<vector<pair<int, int>>> groups;
int sze = 0;
vector<pair<int, int>> curr;
bool valid(int i, int j) {
if (i >= 0 and i < n and j >= 0 and j < n) return true;
return false;
}
void dfs(int i, int j) {
sze++;
vis[i][j] = 1;
curr.emplace_back(i, j);
for (int k = 0; k < 4; ++k) {
int nx = i + dx4[k];
int ny = j + dy4[k];
if (valid(nx, ny) and g[nx][ny] == 1 and !vis[nx][ny]) {
dfs(nx, ny);
}
}
}
void solve() {
cin >> n;
g.assign(n, vector<int>(n));
vis.assign(n, vector<int>(n, 0));
groups.assign(105, {});
for (int i = 0; i < n; ++i) {
for (int j = 0; j < n; ++j) {
cin >> g[i][j];
}
}
for (int i = 0; i < n; ++i) {
for (int j = 0; j < n; ++j) {
if (g[i][j] == 0 or vis[i][j]) continue;
sze = 0; curr.clear();
dfs(i, j);
groups[sze] = curr;
}
}
int p; cin >> p;
```

```
if (groups[p].empty()) {
cout << -1 << " "<< -1 << '\n';
return;
}
sort(all(groups[p]));
cout << groups[p][0].first << " " << groups[p][0].second << '\n';
}
int32_t main(){
// freopen("input.txt","r",stdin);
// freopen("output.txt","w",stdout);
ios_base::sync_with_stdio(false);cin.tie(nullptr);cout.tie(nullptr);
int T = 1;
cin >> T;
for(int i = 1; i <= T; ++i){
// cout << "Case #" << i << ": ";
solve();
}
return 0;
}
```