The 2016 ACM-ICPC Thailand National Programming Contest

(Onsite Questions)

A	Punya's Scope	
	Input	Standard Input
	Output	Standard Output
	Time Limit	3 seconds

Punya dropped out from school. So, he could build the Ultimate Scope Machine (USM), a tool that he hopes it can analyze long sequences easily. The USM tool takes as input a string **S** and a resolution parameter **r** ($1 \leq \mathbf{r} < \text{len}(\mathbf{S})$), and reports all the pairs **S**[i] and **S**[i+**r**] matching certain criteria.

What criteria, might you ask? Having failed Biology in school, he wanted to gain an unorthodox understanding of DNA. The USM was therefore designed to accept only a string consisting of a, t, c, and g (all lower-case). The pairs that the USM reports are exactly pairs of a and t, and c and g. That is, the USM will return the pair (**S**[i], **S**[i+**r**]) if and only if the set {**S**[i], **S**[i+**r**]} = {a, t} or the set {**S**[i], **S**[i+**r**]} = {c, g}.

However, the strings in his study are often long and he has to properly set the parameter **r**. He wants an **r** that yields the maximum number of pairs from the USM. You will help him find this value of **r**. If there are multiple answers, find the largest **r**.

Input:

The first line contains the number of test cases, K ($1 \leq K \leq 12$). The K test cases are followed. Each case is a string on its own line. Each character of the string is one of *a*, *t*, *c*, and *g*; it may contain no other characters, and it has no space inside. For each input string S, it is guaranteed that $1 \leq \text{len(S)} \leq 30{,}000$ and there is a setting of r with at least one matching pair.

Output:

For each test case, print the number *r*, followed by a new line character \n.

Sample Input	Sample Output
4 cattagt aagatctt tggtaaatgat agagagtccttc	2 3 6 6

Sample input/output explanation: For the first input, if he uses r = 1, 2, 3, 4, 5, 6, he will find 2, 3, 0, 0, 2, 0 pairs, respectively. Therefore, setting r = 2 yields the maximum number of pairs. For the second input, if he uses r = 1, 2, 3, …, 7, he will find 1, 0, 3, 2, 1, 2, 1 pairs, respectively. This means r = 3 is the best. The third input is more interesting. If he uses r = 1, 2, 3, …, 10, then he'll find 3, 3, 2, 2, 2, 3, 0, 0, 1, 0 pairs, respectively. The best he can get is 3 pairs. However, if there are many equally-good r's, the largest value is used—i.e. r = 6.

B	Simple Cryptography	
	Input	**Standard Input**
	Output	**Standard Output**
	Time Limit	**1 second**

Cryptography is the art of hiding information. Given a plain text message m, the process called encryption "scrambles" m into another nonsensical message m' and another process called decryption transforms m' back to its original plain text message m. This whole process is done so that the encrypted message m' can be sent out and no one else can decrypt it except the intended receiver. There are several ways to do this. One scheme is given below.

Beforehand The sender and receiver agree on a large prime number p, which may be made public, and the other secret key $k \in \{1, 2, 3, \ldots, p - 1\}$.

Encryption The message m can be any integer in the set $\{0, 1, 2, 3, \ldots, p - 1\}$. The sender encrypts the message m to produce m' by computing:

$$m' = mk \text{ modulo } p$$

Decryption The message m can be recovered from m' by computing:

$$m = m'k^{-1} \text{ modulo } p$$

where k^{-1} is an integer such that $kk^{-1} - 1$ is a multiple of p.

For example, suppose $p = 11$, $k = 10$, and $m = 9$. The encrypted message m' in this case is 2. Given the definition of k^{-1}, we know that $k^{-1} = \frac{tp+1}{k}$ for some integer t. In this case, let $k^{-1} = 10$. We recover the message m by computing 2×10 modulo 11 = 9.

Well, this seems easy to crack since we only have one secret key k. To add the complexity to the computing, r distinct secret keys $\{k_1, k_2, \ldots, k_r\}$ are used instead of just one, where each key $k_i \in \{1, 2, 3, \ldots, p - 1\}$. Everything else is the same as before. The encryption now becomes $m' = mk_1k_2 \ldots k_r$ modulo p. Your job is to decrypt m' to get the original message m.

Input:

The first line of the input contains the number of test cases, t (where $1 \le t \le 22$). The next t lines describe information for each test case. The second line contains $r + 2$ integers for the first test case. The first integer is the message and the second integer is p, the third k_1, the fourth k_2 , …, and the last integer is k_r. The third and the following lines contain the same information as the first test case. In all test cases, $p \le 65{,}414$ and $1 \le r \le 5$. In all test cases, the unsigned long integer type is sufficient for the largest value.

Output:

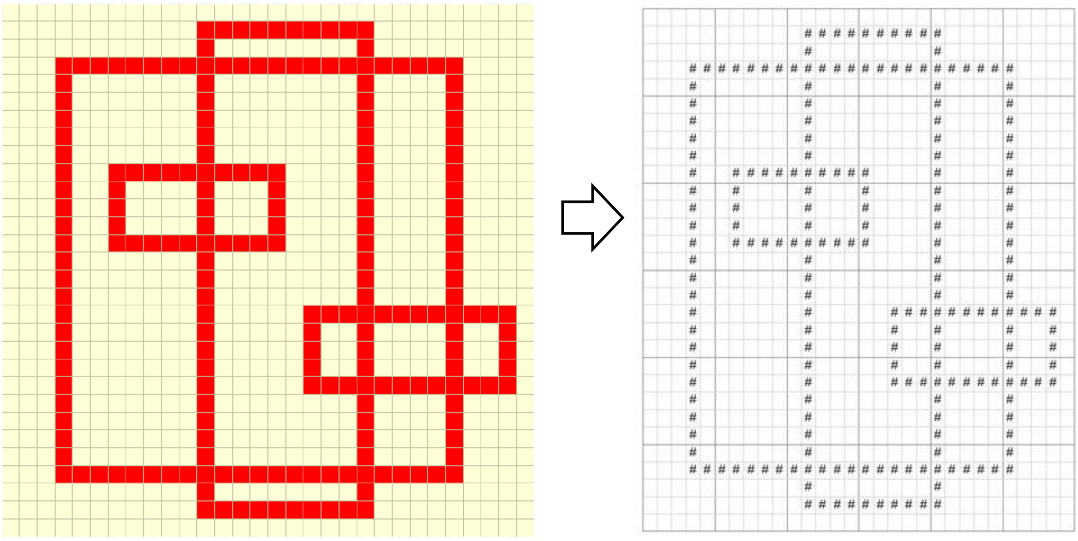
In t lines, report one integer per line. The first integer is the original message m for the first test case, the second integer for the second test case, and so on.

Sample Input	Sample Output
2 2 11 10 4 13 10 5	9 11

Remark: The first test case describes the example in the problem.

C Rectangle Tracking

Input	Standard Input
Output	Standard Output
Time Limit	1 second



(a) (b)

Fig. 1 shows a picture containing overlapped rectangles.

Fig. 1 shows a picture of overlapped rectangles on a canvas. The canvas size is N x N. Fig 1 (b) shows the result after the original picture (Fig 1(a)) has been pre-processed. For the pre-processing mechanism, the pixels of each rectangle in Fig 1(a) are converted to ‘#’ in Fig 1(b). The other pixels, where do not display as rectangles, are translated to ‘0’. In the above figure, there are 24 rectangles consisting of 2 squares and 22 non-square rectangles.

Note:

If all sides of a rectangle are equal, the rectangle is then a square. Otherwise, it is the non-square rectangle.

Your task:

Given the pictures of overlapped rectangles, write a program to count the number of all rectangles in each picture.

Input:

The first line contains integer ***T***, the number of test cases; where *(1 ≤ **T** ≤ 20)***.**

The first line of each test case contains the size of the picture *(10 ≤ **N** ≤ 30)*.

Each test case takes ***N*** lines per picture. Each line contains ***N*** characters, representing '0' or '#'.

Output:

The output contains ***T*** lines of ***T*** test cases. For each test case, each line shows the number of total rectangles, squares and non-square rectangles respectively. For example, in the first line of the sample output, total rectangles = 24, squares = 2 and non-square rectangles = 22.

Sample Input

```
2
30
000000000000000000000000000000
00000000000##########000000000
00000000000#00000000#000000000
000#######################0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#00##########0000#0000#0000
000#00#0000#000#0000#0000#0000
000#00#0000#000#0000#0000#0000
000#00#0000#000#0000#0000#0000
000#00##########0000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000############0
000#0000000#00000#00#0000#00#0
000#0000000#00000#00#0000#00#0
000#0000000#00000#00#0000#00#0
000#0000000#00000############0
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#0000000#00000000#0000#0000
000#######################0000
00000000000#00000000#000000000
00000000000##########000000000
000000000000000000000000000000
10
0####00000
0#00#00000
0#0#####00
0####00#00
000#000#00
000#000#00
000#######
000#000#0#
000#######
0000000000
```

Sample Output

```
24 2 22
7 4 3
```

D

Spiderman

Input	**Standard Input**
Output	**Standard Output**
Time Limit	**1 second**

Recently, Spiderman joined a group called the Avengers, leading by THE Ironman. His job as the protector of NYC is simple. Whenever there is a robbery, a fire, a runaway train, etc., he must go there to help out the people in NYC, as fast as possible. Recently, he has learned that newspaper love-hate relationship depends on how he traveled. He always travels from the corner where two streets crossed, to another corner. If he travels on the street, nobody pays him any attention. Yet, if he travels through the blocks that the journalists who write news live in, there can be two outcomes. If he traveled through the block of communities where the journalist who loves him lives in, he would get positive support from the press.

However, if he traveled through the block of communities where the journalist who hates him lives in, he would get negative support from the press. The group of journalists who love him, and the group of journalists who hate him, obviously hate one another, so they never live in the same community. As such he decided the following, he would go to help out the people at the location as fast as possible. Yet, he would go through the block if it makes the trip faster and would not make his popularity level drops below zero. He wants you to write him an application to find the shortest route with such condition. Luckily for you, the NYC is the city of rectangular blocks. Since he always goes from one corner to another corner, so he cannot swing over multiple communities. Note that, Spiderman can travel both ways on each street.

Your task is simply to find the shortest possible route, such that his popularity level never drops below 0 along the trip. For example, if he wants to go from the top left corner (0,0) to the lower right corner (2,2). Where the distance from (0,0) to (0,1) is 1, and (0,1) to (0,2) is 3, while the distances from (0,0) to (1,0) is 1, and (1,0) to (2,0) is 3. Note that you should use the Euclidean distance function to calculate the distance between two corners, as follows.

$$dist\big((x_1, y_1), (x_2, y_2)\big) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

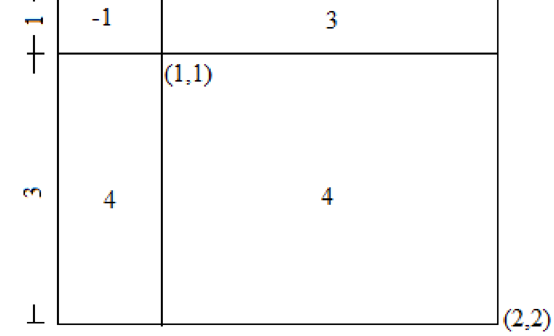


Figure 1. A community map

If Spiderman starts with the popularity level from 1, then from Figure 1. his shortest path from (0,0) to (2,2) is to go from (0,0) to (1,1) and then from (1,1) to (2,2) with the total distance of 5.656. The final popularity level for superman is now 4. But, if he starts with the popularity level of 0, his shortest path is increased to 6.242, either from (0,0) -> (0,1) -> (1,1) –> (2,2) path, or (0,0) -> (1,0) -> (1,1) –> (2,2). Now, his popularity level is 4. Note that, if he has multiple paths, the path where he can accumulate the highest popularity level is chosen. However, if he starts from (2,2) with popularity level of 0 and wants to go to (0,0) instead, then he can travel faster and gain popularity to 4 by going from (2,2) to (1,1), and then go from (1,1) to (0,0) with the shortest total distance of 5.656. But, his final popularity level is 3 in this case.

Input:

In the first line, there is a single number, $1 \leq T \leq 5$, the number of test cases. For each test case, there are 2+ (N-1) + Q lines of input. In the first line, it contains three numbers, N, M, and Q. N and M indicate the numbers of streets running across the city from North to South starting from 0 to N-1, and from West to East starting from 0 to M-1 respectively. Q is the number of Query that your system will be tested on, $1 \leq Q \leq 5$.

- For the first two lines, the first line has N-1 numbers where $1 < N \leq 1000$ numbers indicating the distances between two adjacent streets running North to South. The second line has M-1 numbers where $1 < M \leq 250$ numbers indicating the distances between two adjacent streets running West to East. The distance between two adjacent streets are between [1, 100] (inclusive)
- Then, for the next N-1 lines, each line contains M-1 numbers, each number indicates the popularity level that Spiderman will get if he travels through that community. Starting from the community whose smallest corner is (0, 0) to the community whose smallest corner is
  (N-1, M-1). Note that, the popularity level is between [-100, 100] (inclusive)
- For the final Q lines, each line has 5 numbers in the following order, P, X1, Y1, X2, Y2. $-100 \leq P \leq 100$ indicate the starting popularity level of Spiderman. (X1,Y1) and (X2,Y2) are the starting and ending points respectively. $0 \leq X1,X2 \leq M$, and $0 \leq Y1, Y2 \leq N$.

Output:

For each test case, you have to output Q lines, where each line contains the shortest distance, and the optimal popularity level in order of the given query. For the distance, you need to print out only two decimal places "#.00". Note that, you should not roundup the number when printing.

Sample Input	Sample Output
1 3 3 3 1 3 1 3 -1 3 4 4 1 0 0 2 2 0 0 0 2 2 0 2 2 0 0	5.65 4 6.24 4 5.65 3

E — Elevator GO+

Input	Standard Input
Output	Standard Output
Time Limit	1 second

The head office of ACM Company is a very special building. A very special elevator (called **"Elevator GO+")** is installed in the center of the building. It can move in four directions (Up, Down, Left and Right). The building has **N** floors (1 to N). Each floor has **N** rooms and a lobby. Also, there is the **G** floor (ground floor) under the first floor. This ground floor is the base station of the Elevator, where the elevator is always parked there.

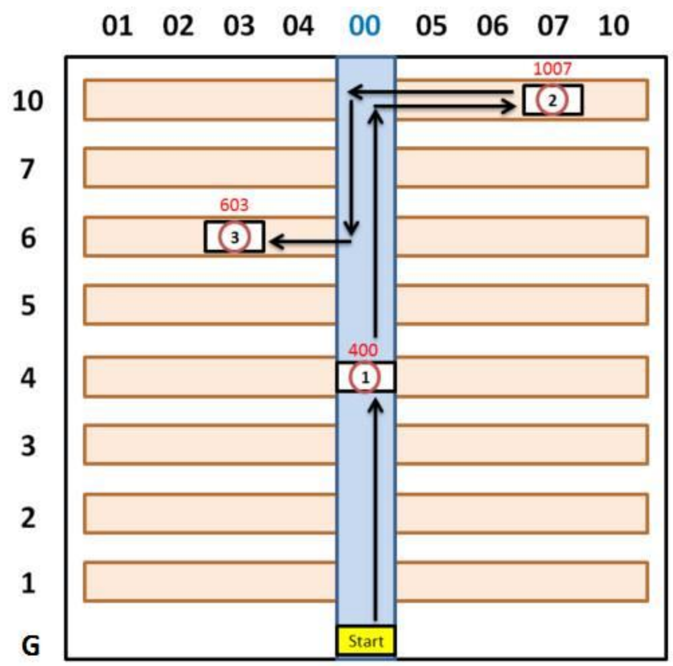
The followings show the time spending for each elevator operation:

1. moving up = 6 seconds per floor
2. moving down = 4 seconds per floor
3. moving left/right = 3 seconds per room
4. stopping to deliver passengers at each destination = 5 seconds per stop
5. changing direction **while moving** to a destination = 5 seconds per time

For each trip, the elevator always starts at the ground floor (G), and may stop to deliver the passengers at several destinations.

Your task:

Compute the total time, spending to reach all destinations in the trip, specified by the test cases.



The elevator spends time = **1 minute 50 seconds**

The figure represents a test case where the building has 8 floors. As mentioned previously, the number of room of each floor is equal to the number of floor. So, there are 8 rooms in each floor. Four rooms are on the left, which are 01, 02, 03, 04 respectively. The other four rooms are on the right, which are 05, 06, 07, 10 respectively. Also, there is one lobby in the middle of the building, which is 00**.** Please note that the floor and room numbers are represented in **Octal**.

On a trip, the destinations for the elevator are **400** (the lobby at the fourth floor), **1007** (the room number 07 at the eighth floor) and **603** (the room number 03 at the sixth floor) respectively. So the total time spent is **1m 50s**.

Input:

There are multiple test cases, which are not more than 200.

Each test case contains two lines.

The first line contains an integer **N**, indicating the number of floors in the building, where **0 ≤ N ≤ 60**. **N** is always an even number. If it is zero, the program is terminated.

The second line represents multiple integers, separated by a space. The first integer in the line indicates the number of destinations, **D** where **1 ≤ D ≤ 100**. The remaining integers in the line are the destinations (room numbers).

Notes: Only floor and room numbers are represented in Octal. The other numbers in this question are represented in Decimal as normal.

Output:

The output of each test case represents in the pattern **"#C: [h m s]"**, where **C** is the case number, and **[h m s]** is the total time spent, representing ***hours, minutes and seconds*** respectively. If the value of **h** or **m or s** is **0**, then do not show it. The output must contain no whitespace at the end of line, and must represent in the exact format in the sample output.

Sample Input	Sample Output
8 3 400 1007 603 6 1 202 10 2 1105 106 12 4 600 1213 1004 1010 0	#1: 1m 50s #2: 28s #3: 2m #4: 2m 37s

F	Atomic Cake	
	Input	**Standard Input**
	Output	**Standard Output**
	Time Limit	**1 second**

Michael (the mathematician) and Paula (the physicist) are close friends. Tomorrow will be Paula's birthday. So, Michael wants to surprise her with a birthday cake. Instead of decorating the cake with candles, he decides to use some particles. Michael tries to use electrons (e), protons (p) and neutrons (n) from Paula's lab to decorate the cake, and in addition, tell her age.

Michael defines how these particles represent in integers. The value of each particle is the same as its charge, e = -1, p = 1 and n = 0. The integer can be represented in a binary way. Given a sequence of particles, $S_kS_{k-1} \ldots S_1S_0$, the integer corresponding to the particle sequence is $\sum_{i=0}^{k} s_i 2^i$. Each S_i can be only e, p or n and k ≥ 0. Not all the particle sequences are possible. Michael finds that he cannot place protons or electrons in 2 consecutive positions due to the electric forces. That is for all $1 \le i \le k$, $S_i \times S_{i-1} = 0$. It is the same meaning as particle sequences cannot include "pp", "ee", "pe" and "ep" subsequences.

For example, you can represent 5 with particle sequence "pnp" ($5 = 1x2^2 + 0x2^1 + 1x2^0$). A particle sequence for 3 can be "pne" ($3 = 1x2^2 + 0x2^1 + -1x2^0$). Notice that particle sequence "pp" is forbidden due to the electric force, even though the represented value is also 3 ($3 = 1x2^1 + 1x2^0$).

Tomorrow, Paula will be x years old. Please help Michael write a program to find a possible particle sequence that represents Paula's age. Assume that he has infinite numbers of particles.

Input:

The first line will contain the number of test cases T, ($1 \le T \le 10$). Each test case contains only one line. There will be an integer x, Paula's age, for the input. Since her age may be calculated relatively to some future years, her age can be negative! The range of x is: $-2{,}000{,}000{,}000 \le x \le 2{,}000{,}000{,}000$.

Output:

For each test case, you have to print out a particle sequence that represents Paula's age, one test case per line. Use only character *p*, *e* and *n*. If there are more than one possible sequences, you can print out any shortest particle sequence (have smallest value of *k*) that satisfies the problem statement.

Sample Input	Sample Output
2 5 3	pnp pne

G	Number Recognition
Input	Standard Input
Output	Standard Output
Time Limit	1 second

Number recognition has benefits in many applications, for example, automatic number plate recognition, signboard recognition and optical character recognition. This problem wants to recognize the numbers 0-9 from a binary image. The binary image consists of pixels (with the value of 0 or 1). The pixels of the numbers are represented as 1, and the pixels of the background are represented as 0 as shown in Fig. 1. The basic recognition uses templates to classify the numbers from the image. The different ten templates for the number 0 to 9 are assigned, as shown in Fig. 2. All templates are represented in 7 x 3 pixels, except the template for the number 1 that is represented in 7x1 pixels.

Your tasks: To recognize number in the image and count a total amount of each number appearing in the image. The numbers can also be recognized with the larger sizes of the templates. For the 7x3 pixel template, it can be enlarged to the 14x6 pixel template. For the 7x1 pixel template, it can be enlarged to the 14x1 template. Please note that the numbers in the images (of the test cases) will not be overlapped, and the orientation of each number follows the pattern of the templates, cannot be rotated to any direction.

1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0
1	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0
1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0
1	0	0	1	1	1	0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1
0	1	0	0	1	1	1	1	0	0	1	1	1	1	1	1
0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0
0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1

Fig. 1 a binary image of numbers

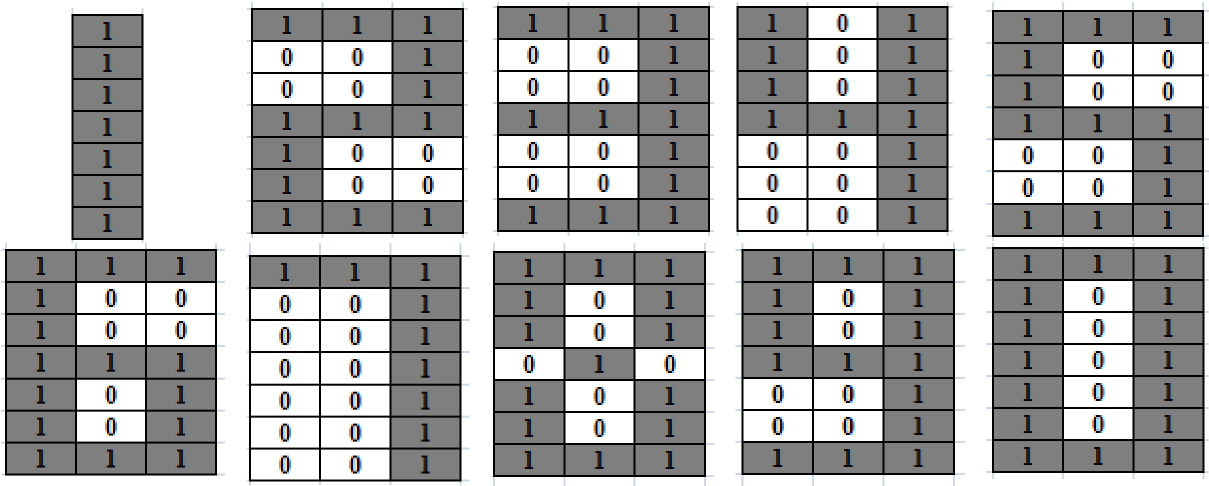


Fig. 2 the templates of the number 0-9

Input

The first line is the number of test cases T, $(1 \le T \le 10)$.

The second line contains the size of image H and W, separated by comma. H is the height of the image, and W is the width of the image. The ranges of H and W are: $12 \le H \le 64$ and $7 \le W \le 64$ respectively.

The next H lines contain the binary image, consisting of 0 and 1 values, separated by comma.

Output

The output contains two lines for each test case.

The first line shows the recognized numbers from the image, separated by comma. These numbers are sorted from the minimum to the maximum value.

The second line shows a total amount of recognition for each number of the first line, respectively. The amount is separated by comma. Please make sure that the output is displayed in the exact format, shown in the sample output.

Sample Input	Sample Output
2 22,16 1,0,0,1,1,1,0,0,0,0,1,1,1,1,1,1 1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1 1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1 1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1 1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1 1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1 1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,1 0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1 0,1,0,1,1,1,1,1,1,0,1,0,0,0,0,0 0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,0 0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,0 0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,0 0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,0 0,1,0,1,0,0,0,0,1,0,1,1,1,1,1,1 0,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0 0,1,0,0,1,1,1,1,0,0,1,1,1,0,0,0 0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0 0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0 0,1,0,1,0,0,0,0,1,0,1,1,1,0,0,0 0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,0 0,1,0,1,0,0,0,0,1,0,1,0,0,0,0,0 0,1,0,1,1,1,1,1,1,0,1,1,1,0,0,0 12,7 0,1,0,1,0,1,0 0,1,0,1,0,1,0 0,1,1,1,0,1,0 0,1,1,1,0,1,0 0,1,1,1,1,1,0 0,1,1,1,1,1,0 0,1,1,1,1,1,0 0,0,1,0,1,0,0 0,0,1,0,1,0,0 0,0,0,0,1,0,0 0,0,0,0,1,0,0 0,0,0,0,0,0,0	1,2,8 2,2,2 1 5

H	Binary-Plan Income	
	Input	**Standard Input**
	Output	**Standard Output**
	Time Limit	**1 second**

A Binary Plan is an organizational structure, used in Multi-Level Marketing (MLM) organizations. In this structure, new members are introduced into a Binary Tree, where each "node" has a left and a right sub-tree. Considered one of the simplest of the popular marketing compensation plans, the binary plan becomes one of the four most commonly used organizational structures by the early 2000s.

The MMK Company has designed a ***Binary-Plan Income*** scheme by calculating the incomes from complete matched nodes between left and right sub-trees. The scheme has been created in order to provide income for VIP member (Silver and Gold states).

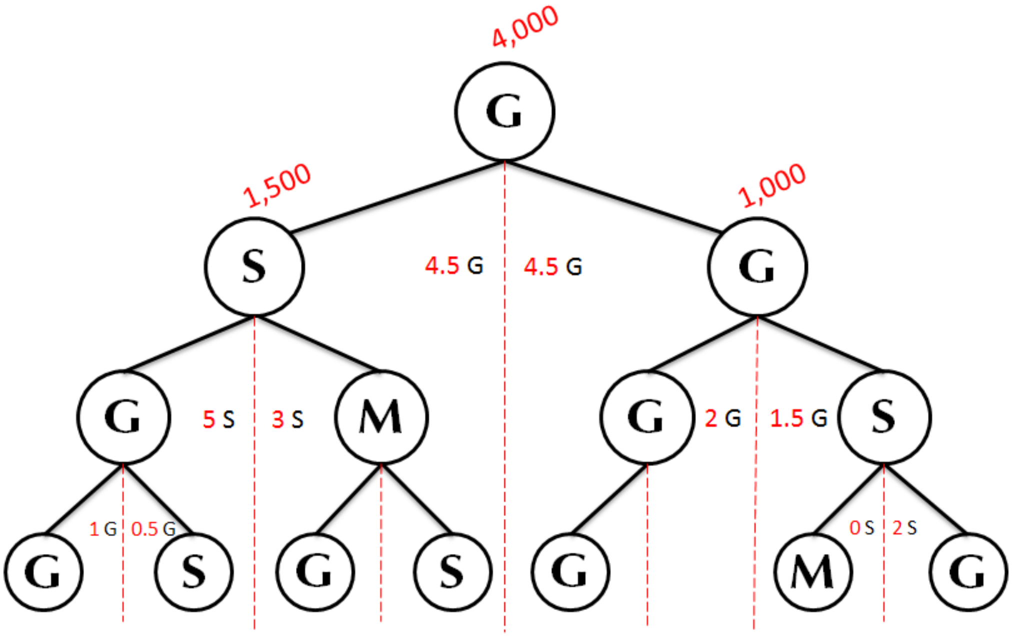
The company defines three states for each node, as shown in the below table.

Node Status	**Point Value (PV) required for each state**	**Daily income (Baht/matched node)**
Member	0	0
Silver	1700	500
Gold	3400	1000

Table 1: Node's PV and Income States

From the table, **Member** (0-1699 PV) is a consumer, who can buy products in the member price and get cumulative PV (Point Value) to be promoted to the next state. The **Member** is not awarded any income. **Silver** has all the **Member** level's benefits. Yet, the **Silver** (1700-3399 PV) can make a daily income of 500-7500 Baht/day by using the ***Binary-Plan Income*** scheme. **Gold** (3400 PV) is the highest state. **Gold** can make the daily income of 1000-15000 Baht/day using the ***Binary-Plan Income*** scheme.

The following figures illustrate the details of ***Binary-Plan Income*** scheme.



Total Income = 4000+1500+1000 = 6,500 Baht

- From the figure, there are four levels of the binary tree with 14 nodes.
- From the first (top) level, the node is **Gold (G)**. So, its income is <u>based on the PV of its status</u>. The calculation steps are as follows.
    - First, calculate the value of **G** from the left nodes (on the left sub-tree) and the right nodes (on the right sub-tree) <u>regarding to the PV of its state</u>.
    - On the left sub-tree, there are **1 Member (M), 3 Silver (S), and 3 Gold (G)**. The total **PV** is **(1x0) + (3x1700) + (3x3400) = 15300,** which is **4.5G**. However, the company policy will consider only the full node status (which is **Gold** in this case) to calculate the income. So, the left sub-tree gets only **4G**.
    - Using the same scheme, we obtain **4G** from the right sub-tree.
    - The next process is matching between the left and right sub-trees by taking the lowest value. So, the matching result of the first-level node is **4G**; four pairs are matched.
    - The total income is then calculated from the number of matched nodes and the node's income state (Table 1) as **4x1000 = 4,000 Baht**.
- By calculating all nodes in the binary tree, the total income of the binary-tree organization (in the figure) is **4000+1500+1000 = 6,500 Baht.**
- The maximum income per day will be calculated for <u>no more than 15 pairs</u>. So, the maximum income per day is 7,500 Baht/day for **Silver** and 15,000 Baht/day for **Gold**.

Note:

"M", **"S"** and **"G"** in the input of the test cases always have Point Values of **0 PV**, **1700 PV** and **3400 PV** respectively.

Your task: Compute the total income from the given binary-tree organization.

Input:

The first line contains an integer **T**, representing the number of test cases, where **1 ≤ T ≤ 50**. The next lines contain **T** data sets. The first line of each data set contains an integer **L** (where **1 ≤ L ≤ 10**), indicating the level of binary tree. The next **L** lines consist of the sequences of characters **("M", "S", "G" and "0")**, representing the state of nodes in binary tree (where **0** is no node). The maximum length of each sequence is ($2^{(level-1)}$), where level is the level of the sequence in the binary tree (1≤ level ≤ **L**). If the binary tree at that level has no node at the rightmost, **"0"** will not be appeared in the input.

Output:

The output of each case displays in two lines.

The first line displays **the total income of all nodes**, in the following pattern: **"Case#C: Total Income = M Baht."**, where **C** is the case number, and **M** is the total income. For **M**, the "*thousands separator (,) format*" must be applied, e.g., **"6,500"**.

The second line displays **the details of the income**, in the following pattern: **"Details => m_1+m_2+m_3+…+m_n"**, where **m_i (i=1 to n)** are the incomes of each node, ordered by the highest to the lowest income. If some specific incomes (**m_i**) repeat several times (as shown in Case#4 in the sample output), the output format must then be shortened, by covering **m_i** with the round brackets and multiplying by the number of repetitions, e.g., **"(1000*2)"**. If some nodes have no income, do not show it. The output contains no whitespace in the end of line. **Significantly, the exact output format as shown in the "Sample Output" (below) must be ensured!**

Sample Input	Sample Output
4 2 G SS 3 S GM SMSS 4 G SG GMGS GSGSG0MG 5 G GS S00G MG0000GM S0GG00000000000G	Case#1: Total Income = 0 Baht. Details => 0 Case#2: Total Income = 1,000 Baht. Details => 1000 Case#3: Total Income = 6,500 Baht. Details => 4000+1500+1000 Case#4: Total Income = 5,500 Baht. Details => 3000+(1000*2)+500

I	An ACM Banquet	
	Input	**Standard Input**
	Output	**Standard Output**
	Time Limit	**3 seconds**

The Association of Cat Mutants (ACM) is renting a special complex for a gathering. The complex is divided into cells, **W** units in width, **D** units in depth, and **H** units in height. Each cell is identified by a 3D coordinate (x, y, z), where $0 \le x < \mathbf{W}$, $0 \le y < \mathbf{D}$, and $0 \le z < \mathbf{H}$. When the gathering begins, cat mutants from all around the world will occupy cells of the complex. In particular, cell (x, y, z) will house $\mathbf{T}(x, y, z)$ mutants.

On the last day of the gathering, there will be a big party. All participants will come together to celebrate their awesomeness in one cell. Any cell is big enough to house all of them. But the ACM is a big fan of efficiency. They will add up <u>the distances all the cat mutants have to travel</u>, and they want to minimize that. Mutants have high dexterity and can move in 6 directions (left, right, forward, backward, up, and down), though not diagonally. Hence, when they travel, the distance between two points is measured by Manhattan distance.

Your task is to find the sum of the distances all the cat mutants have to travel to the optimal location in this complex. However, cat mutants are peculiar and their cell assignment is somewhat unusual. Instead of giving you $\mathbf{T}(x, y, z)$ directly, you will be given *three* vectors **A**, **B**, and **C** ($|\mathbf{A}| = \mathbf{W}$, $|\mathbf{B}| = \mathbf{D}$, and $|\mathbf{C}| = \mathbf{H}$) such that

$$\mathbf{T}(x, y, z) = \mathbf{A}[x] \text{ XOR } \mathbf{B}[y] \text{ XOR } \mathbf{C}[z].$$

Input:

The first line contains the number of test cases **K** ($0 < \mathbf{K} \le 7$). There are **K** test cases following that. Each test case has 3 lines, structured as follows:

- The first line contains the number **W** followed by **W** numbers, listing **A**[0], **A**[1], …, **A**[**W**-1].
- The second line contains the number **D** followed by **D** numbers, listing **B**[0], **B**[1], …, **B**[**D**-1]..
- The third line contains the number **H** followed by **H** numbers, listing **C**[0], **C**[1], …, **C**[**H**-1].

On every line, the numbers are space-separated. It is guaranteed that $\mathbf{W} \times \mathbf{D} \times \mathbf{H} \le 125{,}000{,}000$; $\max(\mathbf{W}, \mathbf{D}, \mathbf{H}) \le 10^5 \times \min(\mathbf{W}, \mathbf{D}, \mathbf{H})$; and $0 \le \mathbf{A}[x], \mathbf{B}[y], \mathbf{C}[z] \le 1{,}000$. It is further promised that the answer will be at most $2^{63} - 1$.

Output:

For each test case, print the answer on its own line (i.e., one number and the new line character "\n").

Sample Input:

```
4
3 2 5 1
2 7 9
4 1 2 3 4
11 3 1 4 1 5 9 2 6 5 3 5
5 2 7 1 8 2
12 0 1 2 3 101 2 1 0 9 8 77 2
10 1 2 3 4 5 6 7 8 9 10
1 0
1 0
20 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21 121 120 119 118 117 116 115 114 113 112 111 110 109 108 107 106 105 104 103 102 101
19 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
```

Sample Output:

```
346
86726
112
13329656
```

J	Matchstick	
	Input	Standard Input
	Output	Standard Output
	Time Limit	1 second

Matchstick puzzle is a very popular game. We represent a number with some matchsticks as shown in the below figure.

In this problem, we would like to know if the represented number is the same when viewing it up-side-down. For example, 8 and 69 are the same when you view them up-side-down, while 3 or 74 are not.

Input:

The first line contains the number of test case, T ($1 \leq T \leq 100$). There is exactly 1 line for each test case which contains only 1 integer X. The length of the input X is guaranteed to be at most 100 characters long per case. It is possible to have some leading zeroes. '08' should be considered different when viewing it up-side-down ('80').

Output:

There should be 1 line of output for each test case. If X is the same when viewing it up-side-down, output 'Y' with no quotation marks. Otherwise, output 'N' with no quotation marks.

Sample Input	Sample Output
4 8 69 3 74	Y Y N N

K

Rubik Number 2D

Input	Standard Input
Output	Standard Output
Time Limit	5 seconds

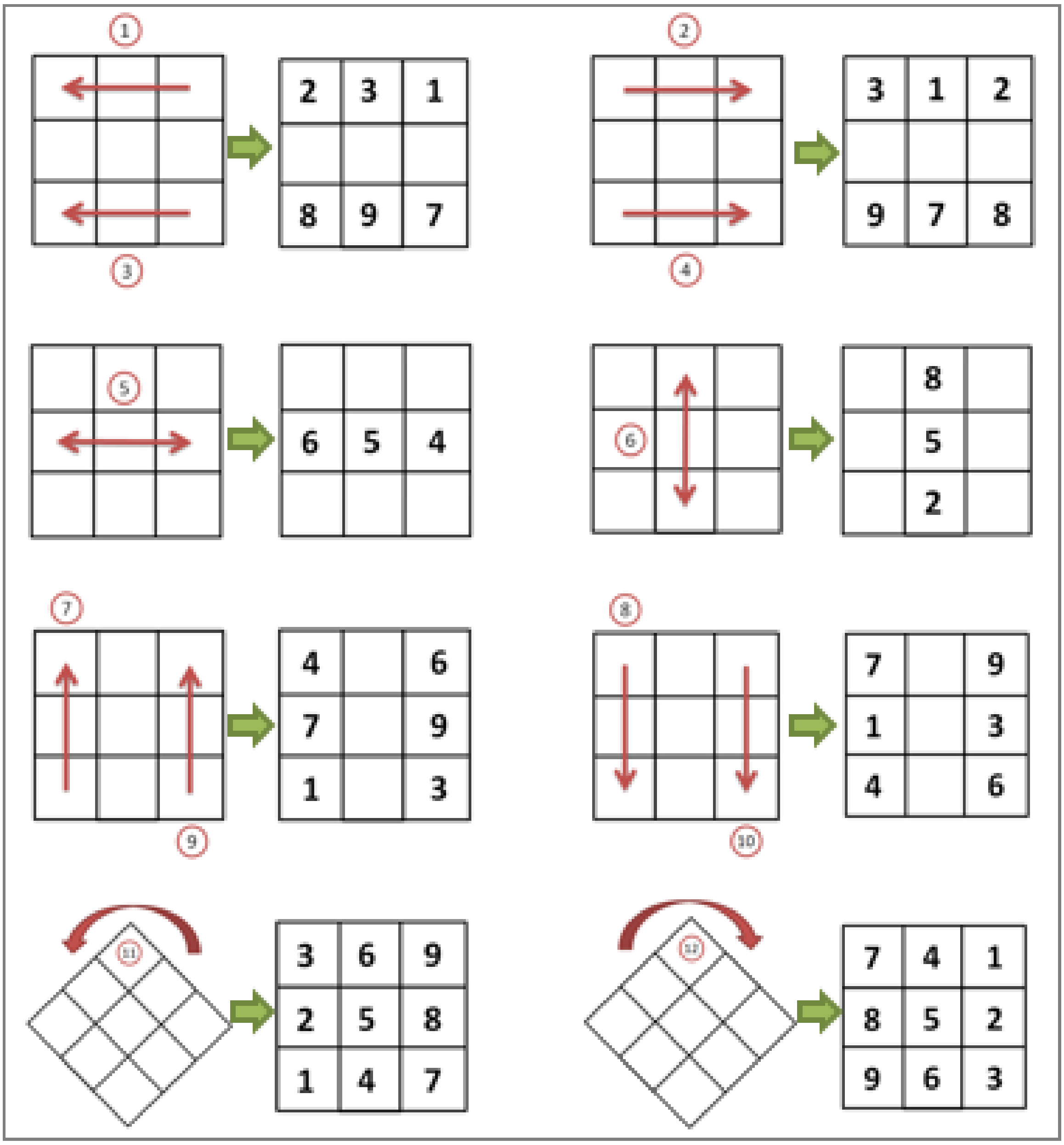
Rubik's Cube is a 3D combination puzzle invented in 1974 by Hungarian sculptor and professor of Architecture ***Ernő Rubik***. As of January 2009, 350 million cubes had been sold worldwide, making it the world's top-selling puzzle game.

In 2016, the **Rubik Number 2D** was invented by Mr. KakaMik. It is a 2D combination puzzle game, which is so much easier than the regular Rubik's Cube. You can make it yourself by drawing a table size 3x3 on a paper and bringing 9 pieces of the bottle lids that have number 1-9 written on them. The aim of this game is to arrange the lids on the paper where the numbers on the lid is matched with one on the table. (See the Fig. (a))

1	2	3
4	5	6
7	8	9

(a)

Moving Rules	
1. Top move Left	2. Top move Right
3. Bottom move Left	4. Bottom move Right
5. Middle swap Horizontally	6. Middle swap Vertically
7. Left move Up	8. Left move Down
9. Right move Up	10. Right move Down
11. Counter-Clockwise	12. Clockwise



(b)

Your task: Find the minimum moves to solve the *Rubik Number 2D* as shown in figure (a).

Input:

The first line contains an integer **T**, indicating the total test cases, where **$1 \le T \le 70$.** Each test case contains the sequence number of 3 digits per line. All three lines represent the random numbers of 1-9 in the table of the *Rubik Number 2D* puzzle. Each table of the puzzle is separated by a blank line.

Output:

Print the minimum moves for the best solve in the **"Case#C: N move(s)."** pattern, where **C** is the case number, and **N** is the minimum number of moves. **Significantly, the exact output format as shown in the "Sample Output" (below) must be ensured!**

Sample Input	Sample Output

`4`
`123`
`456`
`789`
 	
`312`
`456`
`789`
 	
`987`
`654`
`321`
 	
`371`
`659`
`482`
	`Case#1: 0 move(s).`
	`Case#2: 1 move(s).`
	`Case#3: 2 move(s).`
	`Case#4: 4 move(s).`

L	Telecom	
	Input	Standard Input
	Output	Standard Output
	Time Limit	5 seconds

You are assigned on a mission to counter terrorism which has been on a rise lately. You know that even for terrorist, communication is very important, and global terrorist groups rely heavily on telecommunication. However, they do not use traditional telecom operators as those that are legally obliged not to serve them. They instead has their own vast networks of cables for communication.

The specific group that you are tracking has several telecommunication cable networks. The group has two main headquarters, and they can communicate via such network with high redundancy. You know that if you can stop their communication, the group would be much weaker.

However, cutting underground and underwater cable is not easy. It is resource intensive, and you have limited resource. Cutting some parts of the network while still allowing two headquarters to communicate would be futile. So you need to cut the network enough for the two headquarters to be separated.

Each network is controlled by the terrorist group that has different size and shape, and serve different purposes. You want to cut as many of these networks as possible. Your spy has already sent the detailed network information including the resource required to cut each cable.

Input:

The first line has the number of test cases, **T** (where $1 \leq \mathbf{T} \leq 10$).
For each test case, the first line contains two integers **N M** ($1 \leq \mathbf{N} \leq 50$, $1 \leq \mathbf{M} \leq 50{,}000{,}000$), specifying the number of networks that the terrorist group has, and the resource that you have.
Next, there are **N** representation of the telecom network. For each telecom network, the first line contains 2 integers **V E** ($2 \leq \mathbf{V} \leq 100$, $1 \leq \mathbf{E} \leq 1{,}000$).

The following **E** lines contains 3 integers $\mathbf{a_i}$ $\mathbf{b_i}$ $\mathbf{w_i}$ indicating the cable between terminal $\mathbf{a_i}$ and $\mathbf{b_i}$ that requires $\mathbf{w_i}$ (resource) to cut, where $0 \leq \mathbf{a_i} < \mathbf{V}$, $0 \leq \mathbf{b_i} < \mathbf{V}$, $1 \leq \mathbf{w_i} \leq 1{,}000$. There **may be more than** one cable between each terminal. The network is always connected, and the headquarters are always at terminal 0 and terminal 1.

Output:

For each test case, output the maximum number of network that can be sabotage to prevent communication between the two headquarters.

Sample Input	Sample Output
1 3 3 2 1 0 1 1 3 3 0 1 4 0 2 1 1 2 1 4 3 0 1 2 0 2 3 2 3 4	2