2008 National Collegiate Programming Contest

- Problems: There are 10 problems (21 pages in all, not counting this cover page) in this packet.
- Problem Input: Input to the problems are through the input files. Input filenames are given in the table below. Each input file may contain one or more test cases. Test cases may be separated by any delimiter as specified in the problem statements.
- Problem Output: All output should be directed to standard output (screen output).
- Time Limit: The judges will run each submitted program with certain time limit (given in the table below).

Table 1: Problem Information Sheet

	Problem Name	Input File	Time Limit
Problem A	Summation Formula	pa.in	3 sec.
Problem B	Divide Game	pb.in	5 sec.
Problem C	Sierpinski's Triangle	pc.in	5 sec.
Problem D	Maximum Sum of Non-overlapping Rectangles	pd.in	3 sec.
Problem E	Generators of A Cyclic Group	pe.in	3 secs.
Problem F	Triangle and Diamond	pf.in	5 secs.
Problem G	A Shopping Problem	pg.in	10 secs.
Problem H	Permanent Associated to a Directed Graph	ph.in	30 sec.
Problem I	Maze Problem	pi.in	5 sec.
Problem J	Mystery Codes	pj.in	1 sec.

Problem A

Summation Formula

Input File: *pa.in*

Time Limit: *3 sec.*

Most of you remember the following formula:
$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n$

$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} = \frac{1}{3}n^3 + \frac{1}{2}n^2 + n$

How about $\sum_{i=1}^{n} i^3, \sum_{i=1}^{n} i^4, ..., \sum_{i=1}^{n} i^k$?

There is a simple way to derive the closed form formula of these summations. Let's consider the case for $k = 2$.

$$\begin{array}{rrrrrrrrrr}
 & n^3 & - & (n-1)^3 & = & 3n^2 & - & 3n & + & 1 \\
 & (n-1)^3 & - & (n-2)^3 & = & 3(n-1)^2 & - & 3(n-1) & + & 1 \\
 & \cdots & & & & & & & & \\
 & 2^3 & - & 1^3 & = & 3(2)^2 & - & 3(2) & + & 1 \\
+ & 1^3 & - & 0^3 & = & 3(1)^2 & - & 3(1) & + & 1 \\
\hline
 & n^3 & & & = & 3\sum_{i=1}^{n} i^2 & - & 3\sum_{i=1}^{n} i & + & n
\end{array}$$

Note on the righthand side, most terms are canceled, we get $n^3 = 3\sum_{i=1}^{n} i^2 - 3\sum_{i=1}^{n} i + n$. Substitute $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$. We get $3\sum_{i=1}^{n} i^2 = n^3 + 3\frac{n(n+1)}{2} - n$, i.e., $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} = \frac{1}{3}n^3 + \frac{1}{2}n^2 + n$. You are to write a program to compute the formula with a given k ($1 \le k \le 20$). The input for each test case contains two numbers, k, d, and the output should be the coefficient of the n^d terms of the polynomial for $\sum_{i=1}^{n} i^k$. Since the coefficients are all rational numbers, therefore the final output should be a fractional number in its irreducible form. You might need Pascal's triangle to get the coefficients of $(n-1)^k$.

Input File Format

The first line of the input file contains an integer indicating the number of test cases to follow. Each test case contains two integers k, d in a single line.

Output Format

For each test case output an integer if the coefficient is an integer otherwise output the numerator and denominator of the coefficient in irreducible form in one line.

Sample Input

```
5
3 3
3 0
7 6
17 2
17 15
```

Output for the Sample Input

```
1 2
0
7 12
-3617 60
0
```

Problem B
Divide Game

Input File: *pb.in*
Time Limit: *5 sec.*

The rule of playing a divide game is as follows: The game begins with M ($M \leq 12$ and M is even) non-zero integers. In each round, you can choose arbitrary way to arrange these integers into pairs from left to right. In a pair, the bigger integer is divided by the smaller one to get a remainder. The sum of the remainders is the points you get in each round.
The game continues using the remainders as input for the next round. if zero is produced in a round, it is discarded. After zeros are discarded, If the number of non-zero integers is odd, the rightmost integer is discarded so that an even number of non-zero integers will be ready for next round. This game repeats until no more pairs can be formed to perform any division. The final score of a play is the sum of the scores in each round.
For example, let (1 2 3 4 5 6) be the input. In one play, integers are paired as (1 2) (4 6) (3 5). The divide game can be played as follows:

```
(1 2) (4 6) (3 5)  -> First round
  0     2     2    -> you get 4 points.
 (2     2)         -> Second round
     0             -> you get 0 points.
Final score = 4 + 0 = 4
```

In another play, the integers are paired as (1 2) (3 4) (5 6), the game can be played as follows:

```
 (1 2) (3 4) (5 6)  -> First round
   0     1     1    -> you got 2 points.
        (1     1)   -> Second round after right most number is discarded
         0          -> you get 0 points.
Final score = 2 + 0 = 2
```

This pairing generates less final score than the previous one.

Given a list of integers, please write a program to compute the maximum final score that can be reached in a given divide game.

Input File Format

The first integer N ($N \leq 9$) is the number of test cases. Each test case begins with M the number of integers in the first line. The second line of a test case contains M integers separated by space.

Output Format

For each test case, please output the maximum final score that can be reached.

Sample Input

```
2
4
1 2 3 4
6
1 2 3 4 5 6
```

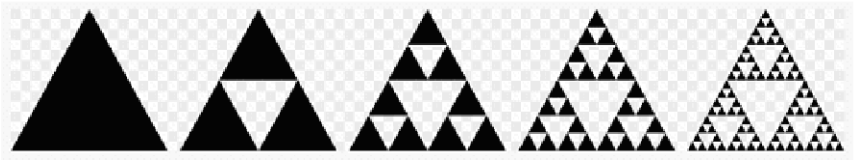
Output for the Sample Input

```
2
4
```

Problem C

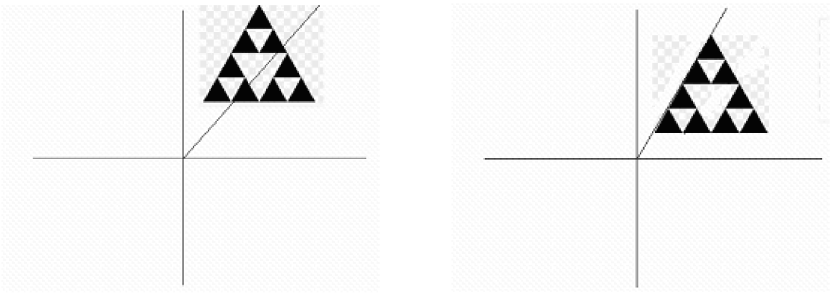
Input File: *pc.in*
Time Limit: *5 sec.*

The interesting shapes in the following are called Sierpinski's Triangle (or gasket), after the Polish mathematician Waclaw Sierpinski who described some of its interesting properties in 1916. Among these is its fractal or self-similar character. The shapes from the left to the right are a large black triangle, a large black triangle consists of three smaller black triangles, each of which itself consists of three smaller black triangles, each of which ..., a process of subdivision which could, with adequate screen resolution, be seen to continue indefinitely. The following shapes list from level 0 to level 4



Level 0 level 1 level 2 level3 level 4

You are given a Sierpinski's triangle in the plane of a certain level and a ray from the origin and passing through a given point. You are asked to write a program to determine how many black triangles the ray intersects with a Sierpinski's triangle. Note that when the ray intersects with a triangle it means that the ray must intersect two edges of the triangle. For example, in Figure 1a the Sierpinski's triangle starts from an equilateral triangles with level 2 and a ray intersects with two smaller black triangles. In Figure 1b the ray (the angle between the ray and x axis is 60 degree) intersects with four triangles, due to it intersects with two vertices of each triangle



(a) (b)

Figure 1: A ray Intersects with Level 2 Sierpinski's triangle

Input File Format

The input will consist of a number of test cases and their data. The first line contains that a positive number n ($n \leq 9$) indicates how many test cases available. The second line to $n+1^{st}$ line are the information of three vertices and level of Sierpinski's triangle (The number of levels is less than eight) and a given point the ray passes. Each line contains the Sierpinski's triangle given by its three vertices, each is a point in the plane, and a level, which is a positive integer, and the given point the ray passes through. Calculation value that is less than 0.00001 is regarded as zero.

Output Format

The output contains one line for each test case. Each line contains an integer to indicate how many triangles the ray intersects.

Sample Input

```
4
1.0  1.0  1.0  6.0  6.0  11.0  1.0  3 -1.0  1.0
1.0  1.0  1.0  6.0  6.0  11.0  1.0  2  1.0  1.0
1.0  1.0  2.0  6.0  7.0  11.0  2.0  4  1.0  1.5
1.0  1.0  2.0  6.0  7.0  11.0  2.0  4  2.0  1.0
```

Output for the Sample Input

```
0
4
4
5
```

Problem D

Maximum Sum of Non-overlapping Rectangles

Input File: *pd.in*

Time Limit: *3 sec.*

Given a 2-dimensional array of positive and negative integers, you are to write a program to find two non-overlapping sub-rectangles with the largest sum. The sum of two rectangles is the sum of all the elements in the two rectangles. A sub-rectangle is any contiguous sub-array of size 1*1 or greater located within the whole array. Two rectangles are non-overlapping if they have no common element. For example, in the following array,

0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

The rectangles | 0 | -2 | / | 9 | 2 | | and | -4 | / | 0 | | are non-overlapping, and

| 9 | 2 | -6 | | and | -4 | 1 | -4 | | are also non-overlapping. But

| 9 | 2 | -6 | | and | -6 | 2 | | `are overlapping since they have a common element -6.`

As shown in the next figure, the sum of the two rectangles is 20, which is the maximum of any two non-overlapping rectangles in the array.

0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

Input File Format

The input consists of a number of test cases. Each test case begins with a positive integer n ($n \leq 100$) on a line by itself indicating the size of the square array. This is followed by n^2 integers separated by white-space (newlines and spaces). These n^2 integers make up the array in row-major order (i.e., all numbers on the first row, left-to-right, then all numbers on the second row, left-to-right, etc.). The numbers in the array will be in the range [-127, 127]. The case with n=0 indicate the end of the input.

Output Format

For each test case, your program has to output the maximum sum in a line. You don't need to process the case with n=0.

Sample Input

```
 4
 0 -2  -7  0
 9  2  -6  2
-4  1  -4  1
-1  8   0 -2
 2
-1 -2
-3 -4
 0
```

Output for the Sample Input

```
 20
 -3
```

Problem E

Generators of A Cyclic Group

Input File: pe.in

Time Limit: 3 seconds

A group is said to be cyclic if it can be generated by a single element. For each of the positive integer, n, consider a cyclic group $Z_n = \{0, 1, 2, \cdots, n-1\}$ under the binary operation $\oplus$ defined as

$$c = a \oplus b \ = \ (a + b) \ mod \ n \ \ \forall \ a, b \in Z_n$$

where c is the remainder of $a + b$ divides n.

In particular, for $g \in Z_n$ and any positive integer k, kg is defined as

$$kg = \underbrace{(g \oplus g \oplus \cdots \oplus g)}_{k} \ mod \ n$$

A generator for Z_n is $g \in Z_n$ such that $\{g, 2g, \cdots, ng\} = Z_n$. This problem is to ask you to write a program to compute $\psi(n)$, the number of elements which can generate the cyclic group for any given positive integer n, where $3 \leq n \leq 500$.

Input File Format

The first line of the input file always contains one integer, K ($K \leq 6$), indicating the number of test cases to come. Each of the following lines is the integer n, based on which, you need to compute $\psi(n)$.

Output Format

For each n, output $\psi(n)$, the number of generators for the group Z_n.

Sample Input

```
6
3
7
10
202
257
451
```

Output for the Sample Input

```
  3    2
  7    6
 10    4
202  100
257  256
451  400
```

Problem F

Triangle and Diamond

Input File: pf.in

Time Limit: 5 seconds

We have an equilateral triangular land on which we like to build diamond-shaped houses. The equilateral triangular land consists of many equilateral triangles that have sides of unit length, and they will be refereed to as "unit equilateral triangles". Two unit equilateral triangles are *adjacent* if and only if they share a common side, and any two adjacent unit equilateral triangles can be used to build a diamond-shaped house. There are unit equilateral triangles that are not suitable for construction, which will be referred to as "bad" triangles. Now given the locations of these bad triangles, please determine the maximum number of houses that can be built in this equilateral triangular land. Houses cannot overlap each other.

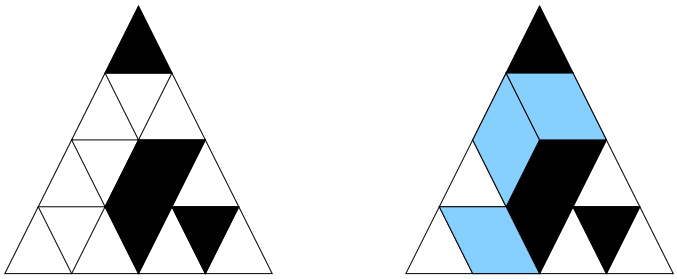


Figure 2: An equilateral triangular land that has 5 bad triangles (left) and 3 houses (right).

Figure 2 illustrates a problem instance. The five bad triangles are illustrated in black and the three houses are illustrated in gray.

Input File Format

The first line of the input file has the number of test cases, there are at most 10 test cases. The first line of a test case contains S ($1 \leq S \leq 100$), the number of unit equilateral triangles along each side of the equilateral triangular land. The next S lines indicate the status of unit equilateral triangles in each row. The unit equilateral triangles are listed from top to bottom, and from left to right. An 1 indicates that the unit equilateral triangle can be used for house construction, and a 0 indicates a bad triangle.

Output Format

For each test case, output the maximum number of houses that can be built in this equilateral triangular land.

Sample Input

```
2
4
0
1 1 1
1 1 0 0 1
1 1 1 0 1 0 1
2
1
1 1 1
```

Output for the Sample Input

```
3
1
```

Problem G

A Shopping Problem

input file: pg.in

Time Limit: 10 seconds

Taimin has married Thinging for k years. When Thinging's birthday comes, Taimin promised Thinging that he will buy k different presents for her. After they arrived at a shopping center, they found that there were n, $n \geq k$, different presents in the shopping center. Furthermore, for each present, there are two kinds of products: perfect and defective. However, all of them are very expensive and might have different prices. Thinging told Taimin that she sees two different defective products as being one perfect product if they are cheaper than that perfect product. Suppose that you are a secretary at Taimin's company. Can you suggest to Taimin what the best way that he can spend his money the least is? Recall that all of the selected presents must be different.

This problem can be described formally as follows. Let S be a set of n integer pairs $s_i = (a_i, b_i)$, $i = 1, 2, \cdots n$, with $a_i > b_i$ and k an integer. Two sets S_a and S_b are called a *k-composition* of S if sets S_a and S_b are two disjoint subsets of S such that $|S_a| + |\frac{S_b}{2}| = k$ and $|S_b|$ is even. The *composition value* with respect to S_a and S_b, denoted by $\lambda(S_a, S_b)$, is defined as $\sum_{s_i \in S_a} a_i + \sum_{s_i \in S_b} b_i$. The *composition problem* is to find two disjoint subsets S_a and S_b from S such that $\lambda(S_a, S_b)$ is minimum. Note that, in the above description, S is the set of presents. Integer a_i (respectively, b_i) denotes the price of perfect (respectively, defective) present s_i.

Input File Format: The first line of the input file contains an integer m ($m \leq 4$), which represents the number of test cases. Each test case contains three lines of input data. The first line of a test case contains two positive integers n ($n \leq 25$) and k ($k \leq 6$) which are the variables described in the story. The second line of a test case contains n integers which are the prices of n perfect products. Note that no price is greater than 50. The third line also contains n integers which are the prices of n defective products corresponding to their perfect products listed in the previous line. For example, in the following sample input, the first test case contains three presents. The prices of their perfect products are $9, 10$, and 16, respectively, and the prices of their defective products are $6, 9$, and 2, respectively.

Output Format: For each test case, output the prices of selected presents in two lines. The first line lists the prices of selected perfect presents in

nondecreasing order and the second line lists the prices of selected defective presents also in nondecreasing order. If no perfect present is selected, output a zero at the first line and if no defective present is selected, output a zero in the second line.

Sample Input

```
2
3 2
9 10 16
6 9 2
6 3
8 10 14 20 42 44
7 6 12 18 20 1
```

Output for the Sample Input

```
10
2 6
8 14
1 6
```

Problem H

Permanent Associated to a Directed Graph

Input File: *ph.in*
Time Limit: *30 sec.*

Suppose G is a directed graph with n vertices $v_1, v_2, \ldots, v_n$ and m directed edges $e_1, e_2, \ldots, e_m$. Each directed edge e_j is an ordered pair of vertices. If $e_j = (v_i, v_{i'})$, then e_j is called an edge from v_i to $v_{i'}$, and e_j is said to be *incident* to v_i and $v_{i'}$. If there is an edge from v_i to $v_{i'}$, then $v_{i'}$ is called an *out-neighbour* of v_i, and v_i is called an *in-neighbour* of $v_{i'}$. The matrix $A(G)$ is an $m \times m$ matrix defined as follows: Assume $e_j = (v_i, v_{i'})$. Then

$$a_{jt} = \begin{cases} 1, \text{ if } e_t \text{ is incident to } v_i, \\ -1, \text{ if } e_t \text{ is incident to } v_{i'} \\ 0, \text{ otherwise.} \end{cases}$$

Given an $m \times m$ matrix $A = [a_{ij}]$. The Ryser's formula for calculating the *permanent* $\mathrm{per}(A)$ of A is as follows:

$$\mathrm{per}(A) = (-1)^m \sum_{S \subseteq \{1,2,\ldots,m\}} (-1)^{|S|} T(S),$$

where for each subset S of $\{1, 2, \ldots, m\}$,

$$T(S) = \prod_{j=1}^{m} \sum_{i \in S} a_{ij}.$$

You are given some directed graphs. Your task is to produce the matrix $A(G)$ for each directed graph G and calculate the permanent of $A(G)$.

Input File Format

The input file consists of a number of test cases. Each test case is a directed graph. The first line of each test case is a positive integer n, which is the number of vertices of G. These n vertices are denoted by positive integers $1, 2, \ldots, n$. The next n lines are information about directed edges incident to these vertices. The i^{th} line of these n lines consists of a sequence of numbers that are the out-neighbours of vertex i. I.e., if j is in the i^{th} line,

then (i, j) is a directed edge of G. Each line ends with a 0 at the end. If the i^{th} line of these n lines consists of a single 0, it means that vertex i has no out-neighbour. The next case starts immediately after these n lines. If the next case starts with a line consisting of a single 0, then that indicates the end of the input file.

There are at most 15 test cases. Each graph has at most 15 vertices and at most 20 directed edges.

Output Format

The output contains one line for each test graph G, which contains a single number: permanent of $A(G)$.

Sample Input

```
4
3 4 0
3 4 0
0
0
8
5 6 7 8 0
5 6 7 8 0
5 6 7 8 0
5 6 7 8 0
0
0
0
0
3
2 0
3 0
1 0
0
```

Output for the Sample Input

```
4
331776
0
```

Problem I
Maze Problem

Input File: *pi.in*
Time Limit: *5 sec.*

Ten Flags amusement park wants to build the world largest maze within the park. The maze is to be the newest addition to a range of human competitive game. Up to 10 contestants can play the game at once. The contestants are assigned to some random positions in the maze and the winner is the first person makes it out of the maze. In order to make the game as fair as possible, the initial position should be chosen carefully so that all contestants are sufficiently apart and that the best-way-out for each contestant is approximately of the same distance. In other words, the distances from each of the initial positions to the closest exit are no more than d unit distances apart. If the set of initial positions meet the above criteria, then it is a good set of initial positions. Please write a program to determine if a set of initial positions for a given maze is a good set of initial positions. For all test cases, output the maximal difference in distance between any pair of given initial positions.

Below are some details about the maze that you should know.

1. The size of any given maze is $rows \times cols$, where $0 \leq rows, cols \leq 100$.
2. The maximum number of initial positions is p, where $0 \leq p \leq 10$. Each given initial positions has at least one way out of the maze.
3. The paths in the maze are all one unit wide. Furthermore, the paths consist of only vertical and/or horizontal segments. There are no diagonal paths.

Input File Format

The first line of the input contains one integer, indicating the number of test cases to follow. For each test case, the fist line consists of three blank space separated integers, $rows$, $cols$, and p. The next $rows$ lines contain data about the given maze. Each of those $rows$ lines contains $cols$ consecutive characters. Each of those characters can be any of the following:

"W" - representing a wall at that position.

"B" - representing an empty position.

"S" - representing a target initial position. Also considered as a "B".

"E" - representing an exit at the outer skirt of the maze.

Output Format

For each test case, please output a single integer on a line representing the difference between the largest and the small distances to the closest exit among all the initial positions.

Sample Input and Output for the Sample Input

Sample Input	
2	
9 10 3	
WWWWWWWWW	
EBBBWWBBBE	
WWSWBWSWWW	<< The first S needed 3 steps to reach an exit.
WWBWBWWBBE	<< The second S needed 4 steps to reach an exit.
WWBWSWWBWW	<< The third S needed 10 steps to reach an exit.
WWBWBWWBWW	
WWBWBBBBBB	
WWBBBBWWWB	
WBWWWWWBBB	
7 6 2	
WWWEWW	
WWWBWW	
WWWBWW	
WWBBBB	
WWSWWB	<< The first S needed 5 steps to reach an exit.
WWBBBS	<< The second S needed 7 steps to reach an exit.
WWWWWW	
Output for the Sample Input	
7	<<The difference between the largest and smallest distances is (10-3)
2	<< The difference between the largest and smallest distances is (7-5)

Problem J
Mystery Codes
Input File: *pj.in*
Time Limit: *1 sec.*

Neptune Communication Protocol Corporation (NCPC) is a renowned consulting company in the science park and specializes in designing protocols, codes for communication and related applications. One day Joe, a star software engineer of NCPC, invents a set of new code, where each code word is simply a permutation from $1, 2, 3, 4, 5, 6, 7, 8$ and can be used to represent 5 bits of information. Together with conventional codes, such as ASCII code, it can be used to encode alphabets. For example, a possible code book is shown in Table 2, where "a" is represented by "00000" and "12345678", etc. Note that the blank character is represented by "11010" and "87162345" in the table.

Table 2: Code book

a	00000	12345678	b	00001	12348567	c	00010	12384567
d	00011	12387456	e	00100	12834567	f	00101	12837456
g	00110	12873456	h	00111	12876345	i	01000	18234567
j	01001	18237456	k	01010	18273456	l	01011	18276345
m	01100	18723456	n	01101	18726345	o	01110	18762345
p	01111	18765234	q	10000	81234567	r	10001	81237456
s	10010	81273456	t	10011	81276345	u	10100	81723456
v	10101	81726345	w	10110	81762345	x	10111	81765234
y	11000	87123456	z	11001	87126345		11010	87162345
.	11011	87165234	,	11100	87612345			

According to Joe, the mapping from 5-bit information to the permutation is done by Algorithm Convert as in Figure 3. Given a valid permutation in the able, we can translate it into an alphabet easily via the associated 5-bit information. Therefore, with the table, we can translate the encoded information: "12876345128345671827634518276345187623458716523 4" into "hello." by table lookup.

ALGORITHM CONVERT(x_1, x_2, x_3, x_4, x_5)
Input: $x_1, \ldots, x_5 \in \{0, 1\}$
Output: $(\pi_1, \ldots, \pi_8)$: a permutation of 1,2,3,4,5,6,7,8.

```
max ← 8; min ← 1;
for i ← 1 to 5 do
  if x_i = 1
    then {π_i ← max; max ← max − 1;}
    else {π_i ← min; min ← min + 1;}
for i ← 6 to 8 do
  {π_i ← min; min ← min + 1;}
Output (π_1, ..., π_8).
```

Figure 3: CONVERT encodes 5 bits of data into a permutation.

A primary objective of transmitting information is to minimize errors. During the transmission of encoded data, assume that any symbol $j \in \{1, \ldots, 8\}$ in a permutation may become $j - 1$ or $j + 1$ because of some interference. This type of error is called $\pm error$. For instance, permutation "12345678" may turn into "21434667" when received. But, unfortunately, this is not a legitimate permutation in the table and it becomes not clear how to recover the original character by merely table lookup. You are asked to write a program, based on the code book in Table 2, to translate a document encoded with permutations and plagued with potential $\pm error$ into plain English. In other words, you need to design an inverse procedure for CONVERT that can correct $\pm error$.

Input File Format

The input file contains a document encoded with a sequence of permutations. Note that each permutation may degenerate into 8 digits from $\{1, 2, 3, 4, 5, 6, 7, 8\}$. I.e., each character is represented with 8 digits from $\{1, 2, 3, 4, 5, 6, 7, 8\}$. If an encoded character is error free, then it is a legitimate permutation and can be found in Table 2; otherwise, because of the $\pm error$, it may not even be a permutation. To simplify the task, we use "*" to indicate newline in the encoded file. The file ends with a single "0". Therefore the encoded file consists of symbols from: 1, 2, 3, 4, 5, 6, 7, 8, * and 0. There is actually no "return" character in the input file.

Output Format

The output is a plain English document with alphabets from Table 2. The last line of the output shows the total number of characters in the original file.

Sample Input

```
12876345128345671827634518276345187623458761234587162345
17623451876234581237456182763451238745687165234*228853462
28435682828534628285346287513467762134677151346717513462 8
75134671248455282853462237845577156233 0
```

Output for the Sample Input

```
hello, world.
hello, world.
Total: 27 characters.
```