Problem A. Coincidence

Input file: `maxcon.im`
Output file: `maxcom.out`

Common subsequence of two strings s_1 and s_2 is a pair of sequences of indices $(\{a_i\}, \{b_i\})$ such that $a_1 < a_2 < \ldots < a_k$, $b_1 < b_2 < \ldots < b_k$, and $s_1[a_i] = s_2[b_i]$ for all $1 \le i \le k$.

Find a longest common subsequence of two strings.

Input

First and second line of an input file contain two strings of French lowercase characters `a`...`z`. There are no spaces before, inside or after the strings. Lengths of strings do not exceed 100.

Output

In the first line of output file output k – the length of a longest common subsequence. On the second line output k numbers – indices of a common subsequence in the first input string. On the third line output the same for the second input string. Index of the first character in the string is 1. Indices should be output in ascending order.

Example

maxcon.im	maxcom.out
abcd cxbydz	2 3 4 1 5

Problem B. Longest Common Subpair

Input file: `subpair.in`
Output file: `subpair.out`

A pair of strings (α, β) is called a *subpair* of a string γ if $\gamma = \gamma_1\alpha\gamma_2\beta\gamma_3$ for some (possibly empty) strings γ_1, γ_2 and γ_3. The length of the pair is the sum of lengths of its strings: $|(\alpha, \beta)| = |\alpha| + |\beta|$.

Given two strings ξ and η find their longest common subpair, that is — such pair (α, β) that it is a subpair of both ξ and η and its length is greatest possible.

Input

Input file contains two strings ξ and η, one on a line. Both strings contain only small letters of the English alphabet. Both string are not empty. The length of each string doesn't exceed 3000.

Output

Output α on the first line of the output file and β on the second line.

Example

subpair.in	subpair.out
abacabadabacaba acabacadacabaca	acaba abaca
ab bc	b

Problem C. Little Brackets

Input file: `brackets.in`
Output file: `brackets.out`

onsider all regular bracket sequences with one type of brackets. Let us call the *depth* of the sequence the maximal difference between the number of opening and the number of closing brackets in a sequence prefix. For example, the depth of the sequence "`()(()())`" is 2, and the depth of "`(((()(()))))`" is 4.

Find out the number of regular bracket sequences with n opening brackets that have the depth equal to k. For example, for $n = 3$ and $k = 2$ there are three such sequences: "`()(())`", "`(()())`", "`(())()`".

Input

Input file contains several test cases. Each test case is described with n and k ($1 \le k \le n \le 50$).

Last testcase is followed by two zeroes. They should not be processed.

Output

For each testcase output the number of regular bracket sequences with n opening brackets that have the depth equal to k.

Separate output for different testcases by a blank line. Adhere to the format of the sample output.

Example

brackets.in	brackets.out
3 2 37 23 0 0	Case 1: 3 Case 2: 203685956218528

Problem D. Travelling Salesman Returns!

Input file: `salesman.in`
Output file: `salesman.out`

Travelling Salesman plans to return to the Alpha Centauri system! All the people wait it! They want new best goods from other systems!

But the Salesman as usual wants to minimize the travel expenses. He selects any starting planet, flies there on the intergalactic spaceship, visits all planets in the system in order which minimizes the total cost, and then flies on the intergalactic spaceship away. Of course he does not want to visit any planet more than once. Your task is to calculate the optimal route for the Salesman. The people can wait no longer!

Input

The Alpha Centauri system contains n planets. This number is written on the first line of the input file ($1 \le n \le 19$). The next n lines contain n numbers each: j-th number of the i-th line is the travel cost from i-th planet to j-th. The numbers are separated by spaces. Numbers a_{ii} should be ignored. All numbers are positive integers which do not exceed 10^8.

Output

Output the minimal total cost in the first line. In the second line output n numbers — the route on which the total cost is minimized.

Example

salesman.in	salesman.out
3 8 1 6 3 5 7 4 9 2	5 3 1 2

Problem E. Long Dominoes

Input file: `dominoes.in`
Output file: `dominoes.out`

Find the number of ways to tile an $m \times n$ rectangle with long dominoes — 3×1 rectangles.

Each domino must be completely within the rectangle, dominoes must not overlap (of course, they may touch each other), each point of the rectangle must be covered.

Input

The input file contains m and n ($1 \le m \le 9$, $1 \le n \le 30$).

Output

Output the number of ways to tile an $m \times n$ rectangle with long dominoes.

Example

dominoes.in	dominoes.out
3 3	2
3 10	28

Problem F. Holidays

Input file: `holidays.in`
Output file: `holidays.out`

ou must know that our country is well known for its strange holidays system. We celebrate anything we can, and often new public holidays are announced, though sometimes some are cancelled.

In year 3141 some archeologists have discovered the document that they consider to be the log of changes in the system of public holidays in our country from year Y_s till year Y_f, inclusively. Each record of the document has the form

`<date-record>, added <date-holiday>`
or
`<date-record>, removed <date-holiday>`

The record of the first type means that the public holiday on `<date-holiday>` was announced on day `<date-record>`, and the record of the second type means that the public holiday on `<date-holiday>` was cancelled on day `<date-record>`.

Unfortunately, all dates of records only include day and month, but not the year. Now the archeologists wonder, what maximal number of public holidays could have been there during the years described. Your task in this problem is to answer this question and provide the version of the document with years inserted, that would guarantee this number of the holidays.

You must assume that all records in the document are in the chronological order and that there were no day when two different events took place, that is, all dates of records in the document must be different.

Note that if the holiday was announced on its own day, the year it is announced there is no public holiday on this date. Analogously, if the holiday is cancelled on its own day, there is still the holiday this year (so people do not have to go to work after listening to morning radio programs).

Also recall, that the day of February 29 exists only in leap years, that is, years that are divisible by 4, except those divisible by 100, except those divisible by 400. For example, years 1996, 2000 and 2004 are leap, while 1999 or 2100 are not.

Input

The first line of the input file contains Y_s and Y_f ($1800 \le Y_s \le Y_f \le 2200$, $Y_f - Y_s \le 100$). Next line contains n — the number of records in the document ($1 \le n \le 100$). Next n lines contain the the document records, one on a line. See sample input for more detailed information.

You must not consider any other holidays except those explicitly specified in the document. You may assume that no holiday is removed before it is announced.

Output

On the first line of the output file print the maximal number of public holidays for the given period. After that print n lines — the version of the document with years inserted that provides the specified number of holidays. Adhere to the format of the sample output.

If it is impossible to interpret the document in the specified way, print -1 on the first and the only line of the output file.

Example

holidays.in

```
1900 1999
9
January 1, added January 1
January 1, added January 7
February 29, added February 29
November 7, added November 7
November 7, removed January 7
September 1, added May 1
August 21, removed November 7
September 1, added June 12
September 1, added December 12
```

holidays.out

```
406
January 1 1900, added January 1
January 1 1901, added January 7
February 29 1904, added February 29
November 7 1904, added November 7
November 7 1905, removed January 7
September 1 1906, added May 1
August 21 1907, removed November 7
September 1 1907, added June 12
September 1 1908, added December 12
```

Problem G. Fibonacci Subsequence

Input file: `fibsubseq.in`
Output file: `fibsubseq.out`

sequence of integer numbers $a_1, a_2, \ldots, a_n$ is called a *Fibonacci sequence* if $a_i = a_{i-2} + a_{i-1}$ for all $i = 3, 4, \ldots, n$.

Given a sequence of integer numbers $c_1, c_2, \ldots, c_m$ you have to find its longest Fibonacci subsequence.

Input

The first line of the input file contains m ($1 \le m \le 3\,000$). Next line contains m integer numbers not exceeding 10^9 by their absolute value.

Output

On the first line of the output file print the maximal length of the Fibonacci subsequence of the given sequence. On the second line print the subsequence itself.

Example

fibsubseq.in	fibsubseq.out
10 1 1 3 -1 2 0 5 -1 -1 8	5 1 -1 0 -1 -1

Problem H. Dowry

Input file: dowry.in
Output file: dowry.out

The daughter of the King of Flatland is going to marry the beautiful prince. The prince is going to give the generous dowry for the King's daughter, but he is unsure which jewels from his collection to give.

There are n jewels in the collection of the prince, each jewel is characterized by its weight w_i and its value v_i. Prince would like to give as valuable dowry to the King as possible. But the King is wise and he would accept the jewels only if their total weight doesn't exceed R. On the other side the prince would consider himself greedy for the rest of his life if he gave the jewels with the total weight less than L.

Help the prince to choose jewels from his collection so that their total weight was between L and R (inclusive), and the total value of the selected jewels was maximal possible.

Input

The first line of the input file contains n ($1 \le n \le 32$), L and R ($0 \le L \le R \le 10^{18}$). The following n lines describe jewels and contain two numbers each — the weight and the value of the corresponding jewel ($1 \le w_i, v_i \le 10^{15}$).

Output

The first line of the output file must contain k — the number of jewels to present to the king. The second line must contain k integer numbers — the numbers of jewels. Jewels are numbered from 1 to n in order they are given in the input file.

If it is impossible to choose the jewels, output 0 at the first line of the output file.

Example

dowry.in	dowry.out
3 6 8 3 10 7 3 8 2	1 2

Problem I. Order-Preserving Codes

Input file: codes.in
Output file: codes.out

Binary code is a mapping of characters of some alphabet to the set of finite length bit sequences. For example, standard ASCII code is a fixed length code, where each character is encoded using 8 bits.

Variable length codes are often used to compress texts taking into account the frequencies of occurence of different characters. Characters that occur more often get shorter codes, while characters occuring less often — longer ones.

To ensure unique decoding of variable length codes so called *prefix codes* are usually used. In a prefix code no code sequence is a proper prefix of another sequence. Prefix code can be easily decoded scanning the encoded sequence from left to right, since no code is the prefix of another, one always knows where the code for the current character ends and the new character starts.

Among prefix codes, the optimal code is known, so called Huffman code. It provides the shortest possible length of the text among all prefix codes that separatly encode each character with an integer number of bits.

However, as many other codes, Huffman code does not preserve character order. That is, Huffman codes for lexicographically ordered characters are not necessarily lexicographicaly ordered.

In this problem you are asked to develop a prefix code that would be optimal for the given text among all order-preserving prefix codes. Code is called order-preserving if for any two characters the code sequence for the character that goes earlier in the alphabet is lexicographically smaller.

Since text itself is not essential for finding the code, only the number of occurences of each character is important, only this data is given.

Input

The first line of the input file contains n — the number of characters in the alphabet ($2 \le n \le 2000$). The next line contains n integer numbers — the number of occurences of the characters in the text for which the code must be developed (numbers are positive and do not exceed 10^9). Characters are described in the alphabetical order.

Output

Output n bit sequences, one on a line — the optimal order-preserving prefix code for the described text.

Example

codes.in	codes.out
5 1 8 2 3 1	00 01 10 110 111

Problem J. Prime Sum

Input file: prime.in
Output file: prime.out

Let us consider a representation of a positive integer number n as a sum of one or more integer numbers:

$$n = x_1 + x_2 + \ldots + x_k.$$

Let us call such sum *prime*, if all terms in it are pairwise relatively prime. Recall, that x and y are called relatively prime, if their greatest common divisor is 1.

Given n, find the number of ways it can be represented as a prime sum. For example, there are six such representation for $n = 5$:

$$5 = 5$$
$$5 = 4 + 1$$
$$5 = 3 + 2$$
$$5 = 3 + 1 + 1$$
$$5 = 2 + 1 + 1 + 1$$
$$5 = 1 + 1 + 1 + 1 + 1$$

Input

Input file contains one number n ($3 \le n \le 150$).

Output

Print one number — the number of ways n can be represented as a prime sum.

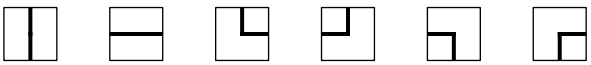
Example

prime.in	prime.out
5	6

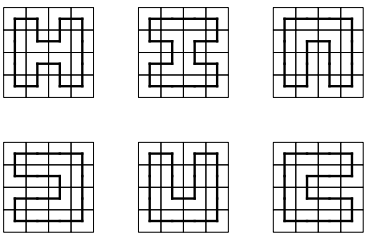
Problem K. Pipe Layout

Input file: pipe.in
Output file: pipe.out

he city is building a centralized heating system in some of the city districts. City district occupies rectangular area and consists of square blocks arranged in a grid. Centralized heating system is a closed circuit of pipes that will be used to run hot water through every block in the district. The city council is considering different layouts of pipes for each district. In order to minimize the total length of pipes but still provide hot water to every block in the district, exactly one pipe must run through every block. A pipe in every block must be connected to the pipes in two neighboring blocks. So, there are at most six possible pipe configurations in every block:



In order to plan their planning activities, the city council wants to know the number of different possible pipe layouts for a given city district. For example, there are exactly 6 different pipe layouts for a district with 16 blocks arranged into 4 by 4 grid:



Input

The input file contains two integer numbers r ($r > 1$) and c ($c > 1$). They specify, correspondingly, the number of rows and columns of blocks in the district. The total number of blocks in a district does not exceed 100 ($r \times c \le 100$).

Output

Output the number of different possible pipe layouts for this district.

Examples

pipe.in	pipe.out
4 4	6
5 7	0
2 8	1
12 8	102283239429

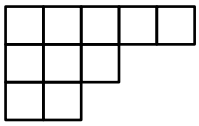
Problem L. Die Young

Input file: young.in
Output file: young.out

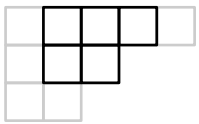
oung diagram is a well known way to describe a partition of a positive integer number. A partition of a number n is a representation as a sum of one or several integer numbers $n = m_1 + m_2 + \ldots + m_k$ where $m_1 \ge m_2 \ge \ldots \ge m_k$.

A diagram consists of n boxes arranged in k rows, where k is the number of terms in the partition. A row representing the number m_i contains m_i boxes. All rows are left-aligned, and sorted from longest to shortest.

The diagram on the picture below corresponds to the partition $10 = 5 + 3 + 2$.



Sometimes it is possible to *inscribe* one Young diagram into the other. Diagram X can be inscribed into the diagram Y if it is possible to delete some boxes from diagram Y so that it turns to diagram X. Note that it is only allowed to remove some boxes, it is not allowed to rotate or flip the diagram. For example, the picture below shows that the diagram for $5 = 3 + 2$ can be inscribed into the diagram for $10 = 5 + 3 + 2$.



On the other hand, for example, it is impossible to inscribe the diagram for $8 = 4 + 4$ into the diagram for $10 = 5 + 3 + 2$.

Given n, your task to find such partition of n that the corresponding Young diagram has the greatest possible number of diagrams that can be inscribed into it.

For example, there are 36 Young diagrams that can be inscribed into the diagram for $10 = 5+3+2$. However, it is not the maximal possible value. The diagram for $10 = 4 + 2 + 2 + 1 + 1$ has the better value, there are 41 diagrams that can be inscribed into it.

Input

Input file contains n ($1 \le n \le 100$).

Output

At the first line of the output file print the maximal number of Young diagrams that can be inscibed into some Young diagram for the partition of n.

At the second line print one or more integer numbers — the number of boxes in each row of the optimal diagram.

Examples

young.in	young.out
10	41 4 2 2 1 1