SGU 149. Computer Network

time limit per test: 0.50 sec.
memory limit per test: 4096 KB
input: standard input
output: standard output

A school bought the first computer some time ago. During the recent years the school bought N-1 new computers. Each new computer was connected to one of settled earlier. Managers of school are anxious about slow functioning of the net and want to know for each computer number Si - maximum distance, for which i-th computer needs to send signal (i.e. length of cable to the most distant computer). You need to provide this information.

Input
There is natural number N (N<=10000) in the first line of input, followed by (N-1) lines with descriptions of computers. i-th line contains two natural numbers - number of computer, to which i-th computer is connected and length of cable used for connection. Total length of cable does not exceed 10^9. Numbers in lines of input are separated by a space.

Output
Write N lines in output file. i-th line must contain number Si for i-th computer (1<=i<=N).

Sample test(s)

Input
```
3
1 1
1 2
```

Output
```
2
3
3
```

SGU 221. Big Bishops

time limit per test: 1 sec.
memory limit per test: 65536 KB
input: standard
output: standard

A bishop is a piece used in the game of chess which is played on a board of square grids. A bishop can only move diagonally from its current position and two bishops attack each other if one is on the path of the other.

Given two numbers n and k, your job is to determine the number of ways one can put k bishops on an $n \times n$ chessboard so that no two of them are in attacking positions.

Input

The input file contains two integers n ($1 \leq n \leq 50$) and k ($0 \leq k \leq n^2$).

Output

Print a line containing the total number of ways one can put the given number of bishops on a chessboard of the given size so that no two of them are in attacking positions.

Sample test(s)

Input

```
Test #1

4 4

Test #2

8 6
```

Output

```
Test #1

260

Test #2

5599888
```

SGU 225. Little Knights

time limit per test: 0.50 sec.
memory limit per test: 65536 KB
input: standard
output: standard

Oh no, here it comes again
Can't remember when we came so close to love before
Hold on, good things never last
Nothing's in the past, it always seems to come again
Again and again and again

...

Bloodied angels fast descending
Moving on a never-bending light
Phantom figures free forever
Out of shadows, shining ever-bright

Neon Knights!

Black Sabbath, "Neon Knights"

The knight is the piece in the game of chess. The knight moves the following way: first he moves 2 squares in horizontal or vertical direction and then 1 square in the perpendicular direction. Figures on the way of knight do not interfere with its movement. Two knights are in the attacking position if they can move to each other's cells.

Input

The input file contains two integers n ($1 \le n \le 10$) and k ($0 \le k \le n^2$).

Output

Print a line containing the total number of ways one can put k knights on a chessboard of size n × n so that no two of them are in the attacking positions.

Sample test(s)

Input

```
Test #1

3 2

Test #2

4 4
```

Output

```
Test #1

28

Test #2

412
```

SGU 327. Yet Another Palindrome

Time limit per test: 1 second(s)
Memory limit: 65536 kilobytes
input: standard
output: standard

You are given N words. Find the shortest palindrome (a word that is the same with its reverse) containing all of them as contiguous substrings.

Input

The first line of input contains an integer N, $1 \le N \le 14$. The next N lines contain one word each, between 1 and 30 characters long. All the characters are small English letters ('`a`' through '`z`').

Output

Output the required palindrome. If there're several possible solutions, output any.

Example(s)

sample input	sample output
1 avtobus	avtobusubotva

sample input	sample output
3 bacd edcab cabac	edcabacde

Beetle

A beetle finds itself on a thin horizontal branch. *"Here I am on a thin horizontal branch,"* thinks the beetle, *"I feel pretty much like on an x-axis!"* It surely is a beetle of deep mathematical thought.

There are also n drops of dew on that same branch, each holding m units of water. Their beetle–based integer coordinates are $x_1, x_2, \ldots, x_n$.

It is clear that the day will be hot. Already in one unit of time one unit of water goes away from each drop. The beetle is thirsty. It is so thirsty that if it reached a drop of dew it would drink it in zero time. In one unit of time the beetle can crawl one unit of length. But would all this crawling pay off? That's what buzzes the beetle.

So you are to write a program which, given coordinates of dew drops, calculates the *maximal* amount of water the beetle can possibly drink.

Input

The input is read from standard input. The first line contains two integers n and m. The next n lines contain integer coordinates $x_1, x_2, \ldots, x_n$.

Output

The program should write one line to standard output containing a single integer: the maximal amount of water the beetle can possibly drink.

Example

Input	Output
3 15 6 -3 1	25

Constraints

$0 \leq n \leq 300$, $1 \leq m \leq 1{,}000{,}000$, $-10{,}000 \leq x_1, x_2, \ldots, x_n \leq 10{,}000$, $x_i \neq x_j$ for $i \neq j$.

SGU 366. Computer Game

Time limit per test: 4 second(s)
Memory limit: 65536 kilobytes
input: standard
output: standard

BerSoft company recently released new computer game, where you play against N opponents. During the game you need to tell to K opponents your opinion about them. You feel pleasure after that and get several score points after that. Each opponent described by two parameters a_i and b_i, where a_i is the amount of pleasure you get when you tell your opinion about this opponent; b_i amount of score points you get in that case. Let us denote A and B summary pleasure and score points that you get during the game. You have never played this game; therefore you don't know what is now what is more advantageous: get more pleasure or score points. You decided to make these values as close as possible. Your task is to select K opponents in a way that minimizes $|A - B|$. If there are several ways to do it, choose one that maximizes $A + B$.

Input
The first line of the input file contains integer number N, K ($1 \le N \le 60000$; $1 \le K \le min(N, 20)$). Next N lines contain two integer numbers each — i-th opponent parameters a_i and b_i ($0 \le a_i \le 50$; $0 \le b_i \le 50$).

Output
On the first line of the output file print values A and B. Print numbers of K selected opponents on the second line. Print numbers in ascending order. If there are several solutions, output any of them.

Example(s)

sample input	sample output
4 2 1 2 2 3 4 1 6 2	6 4 2 3

sample input	sample output
5 3 13 11 3 17 15 20 6 13 17 9	36 33 1 4 5

sample input	sample output

3 1 1 1 3 3 2 2	3 3 2

Back in the day when the famous sixteen-bit three-letter operating system most often used on an 25x80 terminal dominated the PC market, "Nibbles" was everyone's favorite computer game. This problem, however, is not related to Nibbles – it is related to a game called "Connect" which is almost, but not quite, entirely unlike Nibbles.

Connect is played on a board composed of squares organized into R rows and C columns, where both R and C are **odd numbers**. Rows and columns are numbered 1 to R and 1 to C, respectively. Each position on the board is either free or blocked by a wall. Additionally, each board satisfies the following constraints:

- Squares with **both coordinates even** are called rooms. They are **never blocked**.
- Squares with **both coordinates odd** are called barriers. They are **always blocked**.
- All other squares are called **corridors**. They may or may not be blocked.
- Corridors **along the edge** of the board are **always blocked**.

Barriers are represented by the '+' (plus) character, blocked horizontal corridors by the '|' (pipe) character, while blocked vertical corridors are represented by the '–' (minus) character. Rooms and free corridors are represented by the blank character.

At the beginning of the game an **even number** of figures (represented by the uppercase letter 'X') are placed on the board – each in a separate **room**. A *path* between figures A and B is a sequence of **free** squares starting from A, ending with B and moving in one of the **four possible directions** in each step (the path includes both endpoint squares, A and B). The length of the path is the total number of steps needed to get from A to B (which is equal to the number of squares contained in the path minus one).

The goal of the player is to first **divide all the figures into pairs**, and then, for each pair, **connect** the two figures with a path. The pairs should be connected in such a way that **no two paths share a common square**. For a completed game, the score is defined as **the sum of the lengths** of all paths.

```
+-+-+-+-+-+-+-+        +-+-+-+-+-+-+-+
|             |        |  .......    |
+ + + + + + + +        + +.+ + +.+ + +
|X  |   |     |        |X..|   |.    |
+ + + + + + + +        + + + + +.+ + +
|   |   |  X  |        |   |   |..X  |
+-+ + + + + + +        +-+ + + + + + +
|       |     |        |       |     |
+ + + +-+-+-+-+        + + + +-+-+-+-+
|           X|         |           X|
+ + +-+-+-+-+ +        + + +-+-+-+-+.+
|             |        |  ...........|
+ + + + + + + +        + +.+ + + + + +
|  X|         |        |  X|         |
+ + + + + + + +        + + + + + + + +
|   |         |        |   |         |
+-+-+-+-+-+-+-+        +-+-+-+-+-+-+-+
```

Write a program that, given the starting position of the Connect game, plays the game in such a way that the **minimum possible score** is achieved.

The test data will guarantee that a solution, although not necessarily unique, will always exist.

INPUT

The first line of input contains two odd integers R and C, ($5 \le R \le 25$, $5 \le C < 80$) – the numbers of rows and columns.

Each of the following R lines of input contains C characters representing one row of the board. Every character will either be one of the three characters '+', '|' and '–', representing a barrier or a blocked corridor, the blank character representing a free corridor or room, or the 'X' character representing a figure in a room.

There will be at least two figures on the board.

Examples of reading an entire line of input (just be sure that the first line with numbers R and C has been read entirely, including newline character):

PASCAL	**C**	**C++**
`var s : string;` `readln(s);`	`char s[81];` `gets(s);`	`string s;` `getline(cin,s);`

OUTPUT

The first line of output should contain a single integer, the minimum possible score.

The following R lines of output should contain the description of the board after the game has been played. The board should be formatted exactly the same way as in the input, except that each unoccupied square contained in some path should be represented by the '.' (dot) character.

Note: If there are multiple solutions, you should output any one of them.

GRADING

Partial credit is awarded for incorrect or incomplete solutions that correctly find the minimum possible score.

If the minimum possible score is correct, you will receive 80% of the points for the corresponding test case.

If you choose to find the minimum possible score only, you do not need to output anything else.

EXAMPLES

input

```
17 15
+-+-+-+-+-+-+-+
|             |
+ + + + + + + +
|X  |   |     |
+ + + + + + + +
|   |   |  X  |
+-+ + + + + + +
|       |     |
+ + + +-+-+-+-+
|            X|
+ + +-+-+-+-+ +
|             |
+ + + + + + + +
|  X|         |
+ + + + + + + +
|   |         |
+-+-+-+-+-+-+-+
```

output

```
30
+-+-+-+-+-+-+-+
|  .........  |
+ +.+ + + +.+ +
|X..|   |  .  |
+ + + + + +.+ +
|   |   |  X  |
+-+ + + + + + +
|       |     |
+ + + +-+-+-+-+
|            X|
+ + +-+-+-+-+.+
|  ...........|
+ +.+ + + + + +
|  X|         |
+ + + + + + + +
|   |         |
+-+-+-+-+-+-+-+
```

input

```
15 15
+-+-+-+-+-+-+-+
|X|           |
+ + + +-+ + + +
| |   |X|  X  |
+ + + + + + +-+
|     | |   |X|
+ +-+-+ + + + +
|       |   | |
+-+-+ + + +-+ +
|     |     | |
+ + + +-+-+ + +
|     |X  | | |
+ +-+-+ + + + +
|    X|       |
+-+-+-+-+-+-+-+
```

output

```
56
+-+-+-+-+-+-+-+
|X|           |
+.+ + +-+ + + +
|.|   |X|  X  |
+.+ + +.+ +.+-+
|.    |.|  .|X|
+.+-+-+.+ +.+.+
|..... .|...|.|
+-+-+.+.+.+-+.+
|    .|...  |.|
+ + +.+-+-+ +.+
|.....|X..| |.|
+.+-+-+ +.+ +.+
|....X|  .....|
+-+-+-+-+-+-+-+
```

Almost blank page

SPOJ 2202. Tan and His Interesting Game

Problem code: TAN1

Background

Tan always creates some interesting and strange games to kill time,and the Pick-Number Game on Tree is his favorite one.He got the idea from his another game(Pick-Number Game on sequence): there is an integer sequence, he picks a number from the head or the tail of the sequence each turn.When the sequence gets empty,he gets another sequence A,in which A[i] is the *i*-th integer he picks, then he calculates:

$S=A[0]*5^0+A[1]*5^1+...+A[n-1]*5^{n-1}$, while n is the length of the sequence.If S modudo 8 equals to 3,he wins, otherwise he loses(Tan is such a strange person that he likes games with strange rules).

Tan got tired of generating sequence randomly before playing a game, and he changed the rule to avoid it.This time he plays the game on trees. He generates a big tree. Every time he wants to play, he chooses two nodes (A,B) randomly and he finds the path connected A,B (including A,B).In this way he gets a sequence and he can play games.He calls this game "Game(A,B)".He can play many times on a big tree without generating a new one.If he can win in Game(A,B),he says that Game (A,B) is a good game,otherwise Game(A,B) is a bad game.

If a game is a bad game,he can never win,so he has to find a way to identify if a game is bad or good.

He played this game for a long time,and he thought he found a great law: if Game(A,B) is a good game and Game(B,C) is a good game,then Game(A,C) is a good game.And if Game(A,B) is a bad game and Game(B,C) is a bad game,the (A,C) is a bad game.But soon he found it was wrong,but he wanted to know in how many cases it is right.

P.S:"Tan" in Chinese means funny and droll. And Mr.Tan in the story is a real person.

Task

The input data describes a tree with integer numbers on each of its nodes.You should count the number of triple (A,B,C) (A,B,C are distinct nodes) that (A,B),(B,C),(A,C) are all good games or all bad games((A,B,C) and (B,C,A) are supposed to be counted once).

Input

The first line of the test data is the number of test case t,then t test case follow.

For each test case:

The first line contains a single integer M,the number of nodes in the tree(M<=100000).

M lines follow, each contains two integers F_i and V_i. F_i is the father of node i (F_i=0 if node i is the root).V_i is the number on the node i.(0<=V_i<=40000)

Output

For each test case:

The first and only line contains a single integer S, which means there are S triples(A,B,C) that (A,B),(B,C),(A,C) are all good games or all bad games.

Example

Input:
```
1
3
0 3
1 5
1 7
```

Output:
```
0
```