tasks	super95	kokos	trokut	banana
source file	`super95.pas` `super95.c` `super95.cpp`	`kokos.pas` `kokos.c` `kokos.cpp`	`trokut.pas` `trokut.c` `trokut.cpp`	`banana.pas` `banana.c` `banana.cpp`
input data	stdin			
output data	stdout			
time limit (Intel Celeron 2.66Ghz)	1 sec	1 sec	5 sec	2 sec
memory limit (heap)	32 MB			
memory limit (stack)	8 MB			
points	**60**	**70**	**80**	**90**
	300			

This task is based on the pronunciation of integers in the Croatian language. The following table lists pronunciation rules for some of the numbers. All numbers that don't fit into the table start with the same letter as the first digit in them.

1	**j**edan	**10**	**d**eset	**1xx**	**s**to
2	**d**va	**11**	**j**edanaest	**1xxx**	**t**isuću
3	**t**ri	**12**	**d**vanaest	**1xxxx**	**d**eset tisuća
4	**č**etiri	**13**	**t**rinaest	**1xxxxx**	**s**to tisuća
5	**p**et	**14**	**č**etrnaest	**1xxxxxx**	**m**ilijun
6	**š**est	**15**	**p**etnaest	**1xxxxxxx**	**d**eset milijuna
7	**s**edam	**16**	**š**esnaest	**1xxxxxxxx**	**s**to milijuna
8	**o**sam	**17**	**s**edamnaest	**1xxxxxxxxx**	**m**ilijarda
9	**d**evet	**18**	**o**samnaest	**1xxxxxxxxxx**	**d**eset milijardi
		19	**d**evetnaest	**1xxxxxxxxxxx**	**s**to milijardi

Consider the sequence of positive integers whose pronunciation **starts** with the given letter.

For example, for the letter **P** the sequence is:

5, 15, 50, 51, 52, ..., 59, 500, 501, ...

Write a program that finds the N-th number in the sequence of numbers whose pronunciation starts with the given letter.

input data

The first and only line of input contains a single letter (one of '**D**', '**J**', '**M**', '**O**', '**P**', '**S**' or '**T**') and an integer N.

Note: the test cases will be such that the solution will be less than 10^{12} (use 64-bit integral types – int64 in Pascal, long long in C/C++).

output data

Output the required number on a single line.

examples

input

```
S 1
```

output

```
7
```

input

```
P 13
```

output

```
500
```

input

```
M 1000006
```

output

```
1000000005
```

A set of N words is given with the length of each word being exactly **2K** characters.

A **directed graph** with each vertex containing a single letter is called a "**kokos**" if, for **each** word in the set, there **exists a directed path** in the graph such that the labels on the vertices along that path **form the word**. Additionally, for all vertices on that path the following conditions have to be satisfied:

· the in-degree of the first vertex is 0

· the in-degrees of the next K-1 vertices is 1

· the out-degrees of the next K-1 vertices is 1

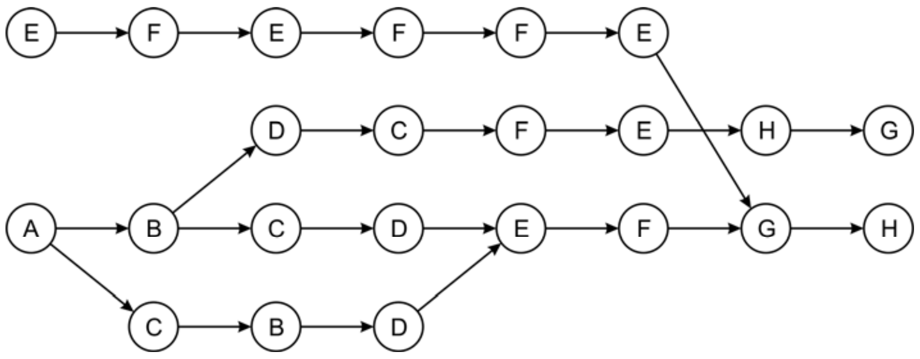
· the out-degree of the last vertex is 0

In other words, paths can fork only on the first K letters, and they can meet only on the last K letters.

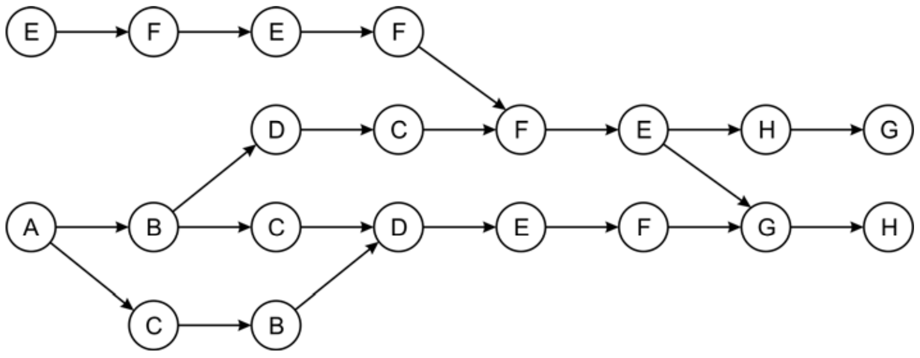
For the given set of the words, we say that the "kokos" is **minimal** if the total number of vertices is **as small as possible**.

Write a program that will find **the number of vertices** in a minimal kokos.

An example of a minimal kokos (the set of the words is from the third example):



It may seem that we can compact the graph like this:



However, this graph is not a **kokos** because paths meet on the 4th letter (D), and they fork on the 6th letter (E).

input data

The first line of input contains two integers N and K, 1 ≤ N ≤ 10 000, 1 ≤ K ≤ 100.

Each of the following N lines contains one word from the set. All letters will be uppercase letters of the English alphabet ('A'-'Z').

output data

The first and only line of output should contain the number of vertices in a minimal kokos.

examples

input

```
2 4
ABCDEFGH
EFGHIJKL
```

output

```
16
```

input

```
4 3
XXZZXX
XXYYZZ
AABBCZ
ABCZZZ
```

output

```
18
```

input

```
4 4
ABCDEFGH
ACBDEFGH
ABDCFEHG
EFEFFEGH
```

output

```
23
```

trokut

There are N different points in a plane.

We say that the triangle formed by some three points as corners is a **super-triangle** if the number of points inside this triangle is **as large as possible**. We consider the points **in the corners** or **on the sides** of the triangle as being **inside** the triangle.

Write a program that will, among given points, find three points that form some super-triangle.

input data

The first line of input contains an integer N, $3 \leq N \leq 300$.

Each of the following N lines contains two integers – the coordinates of one point.

Note: the test data will be such that there will be at least three non-collinear points.

output data

The first line of output should contain the number of points inside the super-triangle.

The second line should contain three numbers – **indices** of the three corner points of the super-triangle, **in any order**.

examples

input

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

output

```
5
1 4 6
```

input

```
9
1 1
2 2
3 3
2 1
3 2
3 1
4 2
4 1
5 1
```

output

```
9
1 9 3
```

input

```
13
1 3
2 4
3 1
4 1
4 2
4 3
4 4
4 5
5 1
5 2
6 1
6 5
7 3
```

output

```
9
3 11 8
```

banana

The traffic network in a country consists of N cities (labeled with integers 1 to N) and N-1 roads connecting the cities. There is a **unique** path between **each pair of different cities**.

Because of the many years of lazy maintenance the roads are pretty damaged and for each road two numbers A and B are known – the integer A represents the **current** time (in seconds) needed to travel along the road, and the integer B represents the **smallest possible time** (in seconds) needed to travel along this road if we repair **all the damage**.

We want to invest a certain amount of money into road repair. For a particular road, the result will be proportional to the amount of invested money. **For each euro** invested in some road, the time needed to travel along that road will be **reduced by one second** (the amount of money invested in some road has to be an **integer**). The travel time cannot be reduced beyond the smallest possible time B described above.

We are given a certain amount of money. We want to distribute this money along different roads in such a way that the **time** needed to travel from the **city 1** to the **most distant city** (after all the repairs) is **as small as possible**.

Write a program that will find this smallest time.

input data

The first line of input contains two integers N and K, 2 ≤ N ≤ 100 000, 0 ≤ K ≤ 1 000 000, the number of cities and the total amount of money (in euros).

Each of the next N-1 lines contains four integers X, Y, A and B, 0 ≤ B ≤ A ≤ 10 000. It means that there is a road between cities X and Y, with the numbers A and B representing the current time and the minimum time as described above.

output data

The first and only line of output should contain a single integer – the minimum time from the task description.

examples

input

```
3 200
1 2 200 100
2 3 450 250
```

output

```
450
```

input

```
5 11
1 2 10 5
1 3 3 2
1 4 9 6
3 5 7 3
```

output

```
6
```

input

```
11 12
1 2 7 5
1 3 20 15
2 4 10 8
2 5 5 3
2 6 6 2
4 7 3 0
4 8 7 2
5 9 8 4
5 10 9 8
5 11 6 5
```

output

```
17
```