UCL Algorithm Contest

Round 2 - 2016

Do not open before the start of the contest.

This page is not really blank.

Problem A
Radio Contest
2pt

We are pleased to inform you that you have just won our latest radio contest!

Now you have the amazing opportunity to go our store and fill a shopping cart with all you want. We have enough stock for you to take as many items of any kind as long as they fit into the shopping cart.

To make things easier for you we will send you a list of our products so that you can choose what you want and get it delivered to your house.

Hurray! You say. But what will you take?...

Write an algorithm that determines what is the maximum total value you can put into a shopping cart given the list of items available.

Input

The first line of the input contains two integers n and W giving the number of different items in the store and the capacity of the shopping cart.

Then follow n lines each with two integers v and w representing the value and the space that the object occupies in the cart, respectively.

Constraints:

- $1 \le n \le 1000$
- $1 \le W \le 1000$
- $1 \le v, w \le 1000$

Output

A single line with the maximum total value that you can take in the shopping cart without exceeding its capacity.

Example

Input 1	Output 1
5 15 4 12 2 2 2 1 1 1 10 4	36

Input 2	Output 2
2 101 100 51 3 2	175

UCL ALGORITHM CONTEST

Problem B
Discrete nail game
1pt + 1pt

Have you ever heard of the nail game? The game is played by putting a nail of length n centimeters in tree a stump and then the players take turns bashing the nail with a hammer. The players have a good control of their strength and can make it go down by exactly 1 centimeter up to k centimeters. For instance, if $k = 3$ then a player can make the nail go down by 1, 2 or 3 centimeters.

If the nail is fully inside the tree stump at the begining of a player's turn, that player loses. Your task, given n and k, is to figure out who the winner will be assmuing that both players play optimally.

Input

The first line of the input contains two integers n and k separated by a single space representing the length of the nail and the maximum amount of centimeters that each player can put the nail down, respecitvely.

Easy constraints (B easy):

- $1 \leq n \leq 10$
- $k = 3$

Hard constraints (B hard):

- $1 \leq n \leq 10^8$
- $1 \leq k \leq n$

Output

A single line with 1 if the first player wins and 2 otherwise.

Example

Input 1

```
1 3
```

Output 1

```
1
```

Input 2

```
4 3
```

Output 2

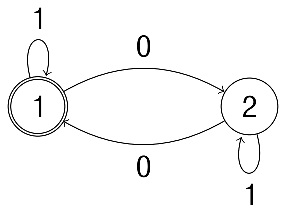
```
2
```

● Problem C
Closest word
1pt + 1pt

An automata provides a compact way of representing a set of strings over an alphabet $\mathcal{A}$. It consists of n states and m transitions. A transition is a triple (s_1, s_2, a) where $s_1, s_2 \in \{1, \ldots, n\}$ are states and a belongs to the alphabet $\mathcal{A}$. We call a the label of the transition. A subset of the states is marked as terminal.

A word w belongs to the set defined by the automata if and only if there is a path of length $|w|$ starting from state 1 such that the i-th transition is labeled with w_i, the i-th character of w, and the path ends at a terminal state.

For example, the following automata represents the set of all binary strings that have an even number of 0's. In the example the initial state 1 is also the only terminal state (this is represented by using a double circle).



The edit distance between two strings is given by the minimum number of operations required to transform one string into the other. The operations that are allowed are: (1.) delete characters, (2.) replace characters and (3.) add characters.

Your taks is, given the description of an automata and a string w, to find the minimum edit distance between w and any word belonging to the set defined by the automata. In other words, if W is the set of all words that the automata represents, you have to compute

$$\min_{w' \in W} \mathit{editDistance}(w, w').$$

Input

The first line of the input contains two integers n and m giving the number of states and the number of transitions, respectively.

Then follows one line with n binary integers separated by single spaces giving which states are terminal. The i-th state is terminal if and only if the i-th integer on that line is equal to 1.

Then follow m lines each with two integers s_1 and s_2 and a character a representing the transition. Each such line means that there is a transition from state s_1 to state s_2 labeled with a.

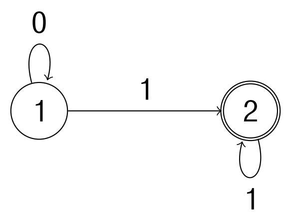
All transitions labels belong to the following alphabet:

$$\{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

The last line of the input contains the string w as described above.

Easy constraints (C easy):

In the easy input the automata is always the same and equal to the following one:



- $1 \le |w| \le 100$

Hard constraints (C hard):

- $1 \le n \le 100$
- there are at most 10 transitions between any two states
- $1 \le |w| \le 100$

Output

A single line with the minimum edit distance between w and any word represented by the automata.

Example

Input 1	Output 1

```
2 3
0 1
1 1 0
1 2 1
2 2 1
00111
```

```
0
```

Input 2	Output 2

```
2 3
0 1
1 1 0
1 2 1
2 2 1
11000
```

```
3
```

Input 3	Output 3

```
4 12
0 0 1 0
1 2 a
1 3 a
1 4 b
2 1 b
2 3 a
2 4 b
3 1 a
3 2 b
3 4 a
4 1 b
4 2 a
4 3 b
ccbcabccaa
```

```
5
```

The third sample test case is only valid for the hard version of the problem.

This page is not really blank.

Problem D
Permutation encryption
1pt + 1pt

The world is at war and you are our last hope. A very important message from our enemies has been intercepted. The problem is that the message is encrypted. Fortunately you know almost all the information necessary to decrypt it!

The message is written over an alphabet $\mathcal{A}$ of size n. To encrypt the text, our enemies chose a permutation of the alphabet and replaced each letter by the corresponding one in the permutation. For instance, if $\mathcal{A}$ = abc (so $n = 3$) and the permutation is cba, then the word cab is encrypted as acb (c $\leftrightarrow$ a, a $\leftrightarrow$ c, b $\leftrightarrow$ b).

For an alphabet of n characters there are $n!$ permutations. A common way to index them is by alphabetical order. For instance, with $\mathcal{A}$ = abc we have $3! = 6$ permutations and their indexes are as shown on the table.

index	permutation
0	abc
1	acb
2	bac
3	bca
4	cab
5	cba

You have the encrypted message and the index of the permutation that was used to encrypt it. Your task is to decrypt the message and save the world.

Input

The input consists of three lines. The first line of the input consists of a string $\mathcal{A}$ representing the alphabet used. The second line of the input consists of a string E representing the encrypted text. The third line contains the index k of the permutation that was used to encrypt the text.

The alphabet A can contain any characters from the set

$$\{\_, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$$

and is given with its characters sorted.

Easy constraints (D easy):

- $|A| \leq 8$
- $|E| \leq 1000$
- $0 \leq k < |A|!$

Hard constraints (D hard):

- $|A| \leq 20$
- $|E| \leq 1000$
- $0 \leq k < |A|!$

Output

A single line with the decrypted text.

Example

Input 1

```
_abc
_bc_a
12
```

Output 1

```
a_cab
```

Input 2

```
_abcdehnprstuy
tcbrsnyrcdsrebbhr_dntupba
58118860800
```

Output 2

```
the_spy_has_been_captured
```

This input is only valid for the hard constraints.

Explanation of the first example

In the first example, the 24 permutations of _abc are:

0	_abc	1	_acb	2	_bac	3	_bca	4	_cab	5	_cba
6	a_bc	7	a_cb	8	ab_c	9	abc_	10	ac_b	11	acb_
12	b_ac	13	b_ca	14	ba_c	15	bac_	16	bc_a	17	bca_
18	c_ab	19	c_ba	20	ca_b	21	cab_	22	cb_a	23	cba_

The permutation with index 12 is b_ac. Thus

$$(\_ \leftrightarrow \mathtt{b}, \mathtt{a} \leftrightarrow \_, \mathtt{b} \leftrightarrow \mathtt{a}, \mathtt{c} \leftrightarrow \mathtt{c})$$

So _ is decrypted to a, a is decrypted to b, b is decrypted to _ and c is decrypted to c giving a_cab.

This page is not really blank.

● Problem E
Two villages deal
1pt + 1pt

Deep in the north of Benin there are two villages. The two villages would like to come together as a single village. To promote that the chiefs of the two villages have decided to arrange n marriges between women of the first village and men of the second village. However each family has some wealth and depending on that, the man's family has to give an offering to the woman's family.

The chiefs disagree on the amount that should be given. One of them says that if the man's family has wealth m and the woman's family has wealth w then $m \cdot w$ should be given. The other thinks it should be the difference, $|m - w|$.

What a conundrum... They need your help to figure out how to arrange the marriges so that the total cost is minimum. Since they don't agree how the wealth should be given, you will have to do both.

Input

The first line contains a single integer n giving the number of women and men to be matched. Then follows a line with n integers separated by single spaces giving the wealth of each woman. Finally the last line contains n integers separated by single spaces giving the wealth of each man.

Constraints:

- $1 \leq n \leq 200$
- the weath of a family does not exceed 100

Output

This problem has two versions, easy and hard. The constraints are the same but what changes is the output.

Easy output (E easy):

A single line with the minimum cost when the cost of marrying a men with a wealth m to a woman with wealth w is $m \cdot w$.

Hard output (E hard):

A single line with the minimum cost when the cost marrying a men with a wealth m to a woman with wealth w is $|m - w|$.

Example

Input 1

```
10
82 63 17 54 49 70 81 79 54 28
5 76 77 51 56 93 50 10 68 10
```

Output 1

```
22880
```

This first sample case is for the easy output.

Input 2

```
10
82 63 17 54 49 70 81 79 54 28
5 76 77 51 56 93 50 10 68 10
```

Output 2

```
103
```

This second sample case is for the hard output.

This page is not really blank.