

**COCI 2016/2017**

Round #1, October 15th, 2016

**Tasks**

Task

Time limit

Memory limit

Score

**Tarifa**

1 s

64 MB

50

**Jetpack**

1 s

64 MB

80

**Cezar**

1 s

64 MB

100

**Mag**

4 s

256 MB

120

**Kralj**

2 s

128 MB

140

**Vještica**

2 s

64 MB

160

**Total**

650

**COCI 2016/2017**

**Task Tarifa**

**Round #1, October 15th, 2016**

**1 s / 64 MB / 50 points**

Pero has negotiated a Very Good data plan with his internet provider. The provider will let Pero use up X megabytes to surf the internet per month. Each megabyte that he doesn’t spend in that month gets transferred to the next month and can still be spent. Of course, Pero can only spend the megabytes he actually has.

If we know how much megabytes Pero has spent in each of the first ​ *N* months of using the plan, determine how many megabytes Pero will have available ​ in the ​ *N* + 1 month of using the plan.

**INPUT**

The first line of input contains t​he inte​ger ​ *X* (1 ≤ ​ *X* ≤ 100).

The second line of input contains ​the int​eger ​ *N*​(1 ≤ ​ *N* ≤ 100).

Each of the following ​ *N* lines contains​ an inte​ger ​ *P* (0 ≤

*i*

​ *Pi*​≤ 10 000), the number of

megabytes spent

in

each

o

f t he first ​ *N*

mo

nths o

f u

sing t he pla

​ n. N

umbers ​ *P* will b

e su

ch t hat

*i*

Pero will never use more megabytes than he actually has.

**OUTPUT**

The first and only line of output must contain the required value from the task.

**SAMPLE TESTS**

**input**

**input**

**input**

10

10

15

3

3

3

4

10

15

6

2

10

2

12

20

**output**

**output**

**output**

28

16

15

**Clarification of the first test case:**

In the first month, out

of 10

total megabytes,

Pero has spent 4

and

transferred

6

into

the

next

month.

In

the second month, out of 16 (10+6) total megabytes, Pero has spent 6 and transferred 10. In the third month, out of 20 (10+10) total megabytes, Pero has

spent 2 and

transferred

18.

In

the

fourth

month,

he

had a total of 28 megabytes to spend.

**COCI 2016/2017**

**Task Jetpack**

**Round #1, October 15th, 2016**

**1 s / 64 MB / 80 points**

Little Mirko got a new mobile phone for his birthday! As all kids nowadays, he quickly downloaded all of the popular mobile games, including Jetpack Joyride.

In the game, the protagonist Barry is running across a field consisting of 10 rows and ​ *N*

columns of squares of equal size. Initially, Barry is located in the ce nter of t he sq

uare in

the

lower left corner. Barry is constantly running to the right at the speed of one square per second. Additionally, he must avoid obstacles that are in his way.

When Mirko presses the phone screen, Barry turns on his super-duper special jetpack and starts his ascent at the speed of one square per second (still moving to the right, now moving diagonally up at an angle of 45°, until he reaches the ceiling, when he will continue moving to the right until Mirko releases the scre

en). W

hen Mi

rko r eleases the p

hone screen,

Barry starts falling down at the speed of one square per second (now moving diagonally again, but this time facing down, until he reaches the floor, when he w ill co

ntinue moving t o

the right).

Mirko just started playing the game recently and he’s still not good at it. He saw on YouTube that someone managed to complete the game​ by crossing all ​ *N* columns, so he is asking you for your help. He will give you the layout of the fields in the game, and you must output the moves he has to play in order to win.

**INPUT**

The first line of input cont

​

ain​s the integer ​ *N* (1 ≤ ​ *N* ≤ 10​5​), the size of the field.

Each of the following 10 li

​

nes contains ​ *N* characters ‘.’ and ‘X’, the layout of the field in the game. The characters ‘X’ denote obstacles, and ‘.’ walkable fields.

**OUTPUT**

The first line of output must

​

co

​ ntain the integer ​ *P* (0 ≤ ​ *P* ≤ 5⋅10​4​), the number of moves Mirko has to make.

​

In the following ​ *P*​ lines, output any series of ​ *P* moves, each in its own line, such that it solves Mirko’s problem from the task.

A move is determined by

​

​

​ two

​

​

​

​**integers**​ *t* and , where denotes the second in which Mirko *i*

​ *xi*

​ *ti*

​ has to press the screen, and ​ *x* i​ denotes how long he needs to keep the screen pressed.

A series of moves must be sorted in chronological ord​er. I​n other words, it must hold

​

​

​ *t* + ≤

*i*

​ *xi*

​

*t*

​

.

*i+1*

Also, no move should beg

​

in​ after the end of the game,

​

​ *t* < .

*i*

​ *N*

The input data will be such that a solution will surely exist.

**COCI 2016/2017**

**Task Jetpack**

**Round #1, October 15th, 2016**

**1 s / 64 MB / 80 points**

**SAMPLE TESTS**

**input**

**input**

11

20

.....XX...X

X..................X

....XX...XX

.X................X.

...XX...XX.

..X..............X..

...........

...X............X...

....XXX....

....X..........X....

...........

.....X........X.....

.....X.....

......X......X......

....XX...X.

.......X....X.......

...XX...XX.

........X..X........

...X...XX..

.........XX.........

**output**

**output**

2

1

1 4

8 10

7 2

**Clarification of the first test case:**

The path Mirko has to take is denoted with ‘\*’:

.....XX...X

....XX...XX

...XX...XX.

...........

....XXX....

.....\*...\*.

....\*X\*.\*.\*

...\*XX.\*.X.

..\*XX...XX.

\*\*.X...XX..

**COCI 2016/2017**

**Task Cezar**

**Round #1, October 15th, 2016**

**1 s / 64 MB / 100 points**

Mirko has an array of

​ ​ *N* ​**different**​ words that he wants to encrypt using a substitution cypher.

We encrypt the text using

a substitution cypher by f irst choosing a ​ *key* –

a permutation o

f t he

English alphabet. Then we replace all occurrences of letter ‘a’ with the f irst le tter of the ke

y,

all occurrences of letter ‘b’ with the second letter of the key, and so on until letter ‘z’.

Besides the words,

Mirko

h

as an array ​ *A*

co

nsisting o

f n

umbers f rom 1 to ​ *N*

given in a

ce

rtain

order (in other words,​ array ​ *A*  is a permutation of numbers fro​m 1

to ​ *N*). M

irko w

ants t o pick

a key such that the array of words after encrypting and le

xicographic sorting corresponds t o

​

array ​ *A*. More precisely, he wants the word initially lo​cated at ​ *A* to be at location *i*

​ *i*  after

encryption and sorting.

Let’s recall that the lexicographic word order is the order in which the words appear in the dictionary. If we are comparing two words, going from left to right, we search for the first position in both words where the letters differ and, based on that, w e determine w

hich w

ord

is lexicographically smaller. If word ​ *X* is the beginning of the word Y, then word X is lexicographically smaller t

​

han word ​ *Y*.

Mirko is currently not in the mood for encrypting, so he kindly asks you to do it for him.

**INPUT**

The first line of input cont

​

ain​s the integer ​ *N* (2 ≤ ​ *N* ≤ 100).

Each of the following

​

​ *N* lines contains a single word that consists of at most 100 lowercase letters of the English alphabet. The words will be mutually distinct.

​ The last line contains ​ *N* integers – ​the elements of array ​ *A*.

**OUTPUT**

In the case when a solution doesn’t exist, output “NE”.

Otherwise, output “DA” in the first line, and in t he second lin

e o

utput a word consisting of 2

6

different letters of the English alphabet – the key for the substitution cipher.

If multiple solutions exist, output any.

**SCORING**

In test cases worth 30 points total, the words will consist of only the first 6 letters of the English alphabet.

**COCI 2016/2017**

**Task Cezar**

**Round #1, October 15th, 2016**

**1 s / 64 MB / 100 points**

**SAMPLE TESTS**

**input**

**input**

**input**

2

3

3

ab

abc

bbb

bc

bcd

ccc

2 1

add

ddd

1 2 3

2 3 1

**output**

**output**

**output**

DA

NE

DA

bacdefghijklmnopqrst

adbcefghijklmnopqrst

uvwxyz

uvwxyz

​

*Note:*  Outputs are split into multiple lines due to lack of horizontal space.

**Clarification of the first test case:**

After encrypting, the words become “ba”, “ac”, after lexicographic sorting, the array becomes “ac”, “ba”, which means the first word ended up in the second spot, and the second word in the first spot.

**Clarification of the third test case:**

After encrypting, the words become “ddd”, “bbb”, “ccc”, after lexicographic sorting, the array becomes

“bbb”, “ccc”, “ddd”, which means the first word ended up in the third spot, the third word in the second spot, and the second word in the first spot.

**COCI 2016/2017**

**Task Mag**

**Round #1, October 15th, 2016**

**4 s / 256 MB / 120 points**

You are given an undirected tree 1with each of its node assigned a mag​ic

​

​ *X*. *i*

The magic of a path is d

2

efined as the product of the magic of the nodes on that path divided by the number of the nodes on the path. For example, the magic of a path that consists of nodes with magic 3 and 5 is 7.5 (3⋅5 / 2).

In the given tree, find the path with the minimal magic and output the magic of that path.

**INPUT**

The first line of input contains the​ intege​r ​ *N* (1 ≤ ​ *N* ≤ 10​6​), the number of nodes in the tree.

Each of the following N - 1 lines contains two​ ​intege​rs,

​

​

​ *A*  a​nd​ ​ (1​ ≤ , ≤ ), the labels of

*i* ​

​ ​ *Bi*

​ *Ai* ​ *Bi* ​ *N*

nodes connected with an edge.

​

The ​ *i* th​ of the following

​

​ *N* lines contains ​t​he​ inte​ge​r

th​

​

​

​ *X* (​ ≤ ≤ 10 ​ magic of the node.

*i* ​ *1* ​  *Xi* ​

​9 ​)​ *,*

​ *i*

**OUTPUT**

Output the magic of the path with minimal magic in the form of a completely reduced fraction

​ ​ ​

​

*P*/​ *Q* (​ *P* and ​ *Q* are relatively prime integers).

In all test cases, it will hold​ that th​e required ​ *P* and ​ *Q* are smaller than 10​18​.

**SCORING**

In test cases worth 24 poi​nts total, it will hold ​ *N* ≤ 1 000.

In test cases worth 36 additional points total, there will not be a node that is connected to more than 2 other nodes.

**SAMPLE TESTS**

**input**

**input**

2

5

1 2

1 2

3

2 4

4

1 3

5 2

2

1

1

1

1

​

​An ​ *undirected tree*  is a c

​

onnected ​graph that consists of ​ *N* nodes and ​ *N* - 1 undirected edges.

2 ​A ​**path**​ in a ​graph​ is a finite ​sequence​ of ​edges​ which connect a sequence of ​vertices

which are all distinct from one another

**COCI 2016/2017**

**Task Mag**

**Round #1, October 15th, 2016**

**4 s / 256 MB / 120 points**

3

**output**

**output**

3/1

1/2

**Clarification of the first test case:**

Notice that the path may begin and end in the same node. The path with the minimal magic consists of the node with magic 3, so the entire path’s magic is 3 / 1.

**Clarification of the second test case:**

The path that consists of nodes with labels 2 and 4 is of magic (1⋅1) / 2 = 1 / 2.

That is also the path with the minimal possible magic.

**COCI 2016/2017**

**Task Kralj**

**Round #1, October 15th, 2016**

**2 s / 128 MB / 140 points**

Young ruler Mirko has declared himself king of dwarves. Upon hearing this, Slavko felt threatened and soon declared himself king of elves! As there cannot be more than one king in the land, they have decided to resolve the issue of power once and for all.

Slavko will, along with

​

​ *N* strongest elves of the kingdom, labeled with numbers from​ 1 to ​ *N*, go visit Mirko’s castle. In the castle hall, they will be gree​ted by ​ *N* strongest dwarves sitting in a circle, labeled ​**clockwise** ​with numbers fro​m 1 to ​ *N*.

Mirko has, upon entering the castle, given a​ number

​

​ *A* to each of Slavko’s elves – the label

*i*

of the dwarf it will fight against. Unfortunately, he didn’t make sure that each elf should get a unique adversary, and soon a terrible fight broke out.

They have decided to solve the problem in the following way:

● Slavko will send his elves to the hall one by one, in the order he chooses. The next elf can enter the hall only after the one before him found a place to sit.

● The elf labeled

​

​ *k* will first approach the ​dw​arf labeled

​

​ *A* . If there isn’t an elf sitting

*k* ​

beside the dwarf, he will sit there. Otherwise, he will continue walking, from dwarf to dwarf, clockwise, until he finds an unclaimed dwarf.

​

Now the ​ *N* resulting pairs of elves and dwarves compete in armwrestling, and ​**the stronger** **one always wins**​.

Slavko is well prepared for this event. He has studied all the fighters and determined the strength of each one. Now he wants to send the elves to the hall in the order which, after they all sit down, will bring the most victories for him.

Help him and calculate ​**the highest number of victories**​ in duels that can be achieved by **elves**​!

**INPUT**

The first line of input cont

​

​

ains the integer ​ *N* (1 ≤ ​ *N* ≤ 5⋅10​5​)

The second line of input c

​

​

​ on​tains

​

​

​ *N* integers ​ *A* (1 ≤ ≤ ), the adversaries chosen by Mirko.

*i*

​ *Ai* ​ *N*

The third line of input con

​

​

​

tains

​

​

​ *N* integers ​ *P* (1 ≤ ≤ 10

*i*

​ *Pi*

​9​), the dwarves’ strengths.

The fourth line of input co

​

​

​

ntains

​

​

​ *N* integers ​ *V* (1 ≤ ≤ 10

*i*

​ *Vi*

​9​), the elves’ strengths.

All strengths from the input will be mutually distinct.

**OUTPUT**

The first and only line of input must contain the maximum number of victories that can be achieved by elves.

**COCI 2016/2017**

**Task Kralj**

**Round #1, October 15th, 2016**

**2 s / 128 MB / 140 points**

**SCORING**

In test cases worth 40% of total po​ints, Mirko will choose the dwarf labeled with 1 (

​

​ *A* = 1 for

*i*

​

​

each ​ *i* from 1 to ​ *N*) as an adversary in each elf duel.

**SAMPLE TESTS**

**input**

**input**

**input**

3

4

3

2 3 3

3 1 3 3

1 2 3

4 1 10

5 8 7 10

8 4 3

2 7 3

4 1 2 6

9 2 6

**output**

**output**

**output**

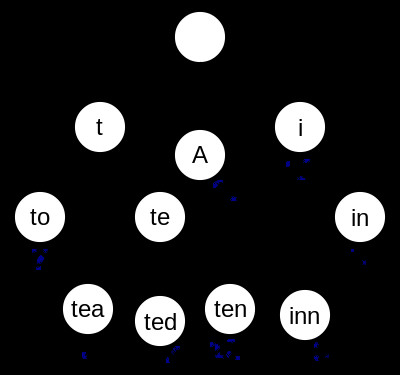
2

1

2

**Clarification of the first test case:**

Slavko can sort the elves in the following way: 3, 2, 1. This way, the elf number 3 will sit beside dwarf number 3, elf 2 will have to move one seat clockwise and sit beside dwarf 1, and the elf number 2 will sit beside the dwarf number 2. Elves 1 and 2 will win their duels, and elf 3 will lose.



**COCI 2016/2017**

**Task Vještica**

**Round #1, October 15th, 2016**

**2 s / 64 MB / 160 points**

Young hero, an adventurer Matej, h

as, a

fter a long a

nd strenuous jo

urney, a

rrived t o h

is f inal

destination – the house of evil witch Marija. In order to complete his adventure, he must solve the final puzzle the witch gives him.

To even begin solving her puzzle, our hero needs to b

ecome f amiliar w

ith the data st

ructure

called prefix tree (trie).

A prefix tree is a data

structure that re

presents a

ll prefixes1 o

f w

ords from a ce

rtain se

t in t he

following way:

● Each edge of the tree is denoted with a letter from the alphabet.

● The root of the tree represents an empty prefix.

● All other nodes in the tree represent a non-empty prefix in a way that each node represents a prefix obtained by concatenating letters written on the edges that lead from the root of the tree to that node (in that order).

● There will never be two edges labeled with the same letter coming out of a single node (this way we minimize the number of nodes necessary to represent all prefixes).

*Prefix tree for words: “A”, “to”, “tea”, “ted”, “ten”, “i”, “in”, i “inn”.*

Only after Matej learned what a prefix tree was does the real puzzle begin!

The witch, as you may have guessed, has ​ *N* words that consist of lowercase letters of the English alphabet. The puzzle would be very simple if the witch wanted to know the number 1 A prefix of a word is a consecutive subarray of letters from the beginning of the word to a certain position in the word.

**COCI 2016/2017**

**Task Vještica**

**Round #1, October 15th, 2016**

**2 s / 64 MB / 160 points**

of nodes of the prefix

tre

e f or t hat set o

f words, but sh

e is n

ot interested in

t his. S

he w

ants to

know ​**the minimal number of nodes** a prefix tree can have ​**a** **fter p**

**ermuting the letters of**

**each word in an arbitrary manner**​.

Help Matej find the answer to the puzzle!

**INPUT**

​

​

The first line of input contains the integer ​ *N*​(1 ≤ ​ *N* ≤ 16).

Each of the following ​ *N* lines contains a single word consisting of lowercase letter of the English alphabet.

The total length of all words will be less than 1 000 000.

**OUTPUT**

The first and only line of output must contain a number, the answer to witch Marija’s puzzle.

**SAMPLE TESTS**

**input**

**input**

**input**

3

3

4

a

a

baab

ab

ab

abab

abc

c

aabb

bbaa

**output**

**output**

**output**

4

4

5

**Clarification of the third test case:**

All words can be permuted

into

the

word

“aabb”,

so

the

prefix

tree

will

have

5

nodes

(4

+

1

for

the

root

of

the tree – the empty prefix).