2019 regionals | icpc latin america region

icpc international collegiate programming contest

icpc.foundation

International Collegiate Programming Contest 2019

Latin American Regional Contests

November 8th-9th, 2019

Warmup Session

This problem set contains 3 problems; pages are numbered from 1 to 4.

This problem set is used in simultaneous contests hosted in the following countries:

Argentina, Bolivia, Brasil, Chile, Colombia, Costa Rica, Cuba, Ecuador
El Salvador, México, Perú, República Dominicana and Venezuela

v1.0

General information

Unless otherwise stated, the following conditions hold for all problems.

Program name

1. Your solution must be called *codename*`.c`, *codename*`.cpp`, *codename*`.java`, *codename*`.kt`, *codename*`.py2` or *codename*`.py3`, where *codename* is the capital letter which identifies the problem.

Input

1. The input must be read from standard input.
2. The input consists of a single test case, which is described using a number of lines that depends on the problem. No extra data appear in the input.
3. When a line of data contains several values, they are separated by *single* spaces. No other spaces appear in the input. There are no empty lines.
4. The English alphabet is used. There are no letters with tildes, accents, diaereses or other diacritical marks (ñ, Ã, é, Ì, ô, Ü, ç, etcetera).
5. Every line, including the last one, has the usual end-of-line mark.

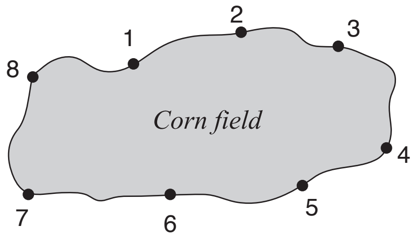
Output

1. The output must be written to standard output.
2. The result of the test case must appear in the output using a number of lines that depends on the problem. No extra data should appear in the output.
3. When a line of results contains several values, they must be separated by *single* spaces. No other spaces should appear in the output. There should be no empty lines.
4. The English alphabet must be used. There should be no letters with tildes, accents, diaereses or other diacritical marks (ñ, Ã, é, Ì, ô, Ü, ç, etcetera).
5. Every line, including the last one, must have the usual end-of-line mark.
6. To output real numbers, round them to the closest rational with the required number of digits after the decimal point. Test case is such that there are no ties when rounding as specified.

Problem A – Farm Robot

To discourage birds such as crows and sparrows from feeding on his crops a farmer needed to put some scarecrows in his corn field. His nephew really likes robots, and suggested that he should use a robot scarecrow instead: "A single robot scarecrow can better protect the whole corn field and will last way more than ten traditional ones!", he said.

Since the farmer thinks his nephew is a smart boy, he took his advice and bought a robot scarecrow. The robot moves along a pathway that surrounds the corn field. In the pathway there are N unmanned charging stations, numbered sequentially in clockwise order starting from 1. The figure below shows an example with eight charging stations.



The robot begins every day at station number 1, and is issued a sequence of commands that are to be performed in order during the day. These commands are generated based on advanced machine learning algorithms that work on data collected by sensors spread through the corn field, ensuring an optimal coverage of the crop. Each command results in the robot moving to another charging station next to the one it is currently at, either in clockwise or counter-clockwise direction.

Despite the promises of optimal coverage by the robot, at the end of a certain day the farmer found part of his crop devastated. To figure out what might have happened the farmer wants to know how many times the robot was at the charging station closest to the devastated area. Given the number of the station closest to the devastated area and the sequence of commands for a single day, can you help the farmer find this number?

Input

The first line contains three integers N, C and S representing respectively the number of posts ($2 \leq N \leq 100$), the number of commands ($1 \leq C \leq 1000$) and the charging station closest to the devastated area ($1 \leq S \leq N$). The second line contains C integers $X_1, X_2, \ldots, X_C$, representing the sequence of commands received by the robot scarecrow. For $i = 1, 2, \ldots, C$, if X_i is 1 then the i-th command means "move to the next charging station in clockwise order", whereas if X_i is -1 then the i-th command means "move to the next charging station in counter-clockwise order". The robot always starts at station number 1.

Output

Output a single line with an integer indicating the number of times the robot was at station number S during the day.

Sample input 1	**Sample output 1**

`8 8 3`
`1 -1 1 1 1 -1 1 1`
	`2`

Sample input 2	**Sample output 2**

`5 4 1`
`1 1 1 1`
	`1`

Sample input 3	**Sample output 3**

`2 1 1`
`1`
	`1`

Sample input 4	**Sample output 4**

`2 1 2`
`1`
	`1`

Sample input 5	**Sample output 5**

`2 2 1`
`-1 1`
	`2`

Sample input 6	**Sample output 6**

`2 2 1`
`-1 -1`
	`2`

Problem B – A Symmetrical Pizza

Bob has a symmetry craze. Everything in his life must be symmetric: his house, his clothes, his car, even his food. And pizza is no exception. For him to eat a pizza, all the toppings, like tomatoes, olives, pepperoni or basil, must be arranged with some degree of rotational symmetry.

This evening Bob ordered some pizza. As usual, when it arrived, he asked the delivery driver to demonstrate that the pizza met his demands for rotational symmetry. The driver demonstrated the symmetry then, as they are trained to do, using the following procedure:

- take a picture of the pizza with a cellphone;
- rotate the pizza by R degrees around its center;
- take another picture;
- show Bob the two pictures side by side, so that he sees that the pizza appears identical in both.

Satisfied, Bob paid for the pizza and took it to the kitchen. In order to test his brand new laser pizza cutter, he decided to cut the pizza in as many slices as possible. Of course, Bob wants to cut the slices in a way that all of them look exactly the same, in accordance with another of his crazes. Now given the angle R of the symmetry demonstration, Bob wants to know the maximum amount of equal slices he can cut the pizza in.

Input

The input consists of a single line that contains a rational number R ($0 < R < 360$) indicating the angle of the rotational symmetry demonstration. This number has exactly two digits after the decimal point.

Output

Output a single line with an integer representing the maximum amount of equal slices Bob can cut the pizza in, based on the provided information.

Sample input 1	**Sample output 1**
`45.00`	`8`

Sample input 2	**Sample output 2**
`180.00`	`2`

Sample input 3	**Sample output 3**
`240.00`	`3`

Sample input 4	**Sample output 4**
`35.00`	`72`

Sample input 5	**Sample output 5**
`2.50`	`144`

Sample input 6	**Sample output 6**
`11.34`	`2000`

Problem C – Database of Clients

Nowadays there are billions of email users. A little-known fact is that some email providers offer way more than the usual `username@provider.com` email address.

Some providers simply ignore dots in usernames. Thus, if John owns the username `johnsmith`, he could tell people that his email address is `johnsmith@provider.com`, `john.smith@provider.com` or `john.s.mith@provider.com`, among others. Emails sent to any of these addresses would end up on his mailbox.

Other providers allow appending the character "`+`" followed by any combination of letters and/or digits after the username. With this feature, by registering the username `johnsmith`, John would also be able to use `johnsmith+friends@provider.com` and `johnsmith+2x3is6@provider.com`.

Sometimes both features are available at once and in those cases `john.smith+icpc@provider.com` and `john.smith+wants.2.eat.lemon.3.14@provider.com` are valid addresses that John could use.

This is quite useful for users, who can manage different addresses to help organize their mailboxes and easily filter the newsletters eventually sent after registering on a new website. Unfortunately, this also opens up space for abuse. Some websites rely upon the fact that each email address identifies a single user. However, a misbehaving user might easily create multiple accounts by taking advantage of the multiple addresses allowed by the email provider.

After learning all of this your boss got really worried. What if the number of unique users that has been reported to the shareholders is not accurate, bloated by duplicate accounts instead? That brings you to the task at hand: given the list of all email addresses from the users database of the company, you must determine the real number of unique users, assuming that all email providers have both described features available.

Input

The first line contains an integer N ($1 \leq N \leq 1000$) representing the number of email addresses in the database. Each of the next N lines contains a string of at most 100 characters representing an email address in the database. Each email address has the form *localpart*`@`*provider* where *localpart* is a non-empty list of labels with a "`.`" (dot) or a "`+`" (plus sign) between each pair of consecutive labels, and *provider* is a non-empty list of labels always with a "`.`" (dot) between each pair of consecutive labels. A label is a non-empty sequence of lowercase letters and/or digits. The character "`+`" (plus sign) appears at most once in each email address.

Output

Output a single line with an integer indicating the number of unique users in the database.

Sample input 1

```
2
two.different.providers@now.here
two.different.providers@nowhere
```

Sample output 1

```
2
```

Sample input 2

```
2
1.2.3@testing
testing@1.2.3
```

Sample output 2

```
2
```

Sample input 3

```
7
alice@e.mail
eve@another.mail
bob@e.mail
joe90@e.mail
b.o.b@e.mail
bob+new@e.mail
bob@another.provider
```

Sample output 3

```
5
```