icpc international collegiate programming contest

ICPC North America Regionals 2019

ICPC Southern California Regional Contest

Official Problem Set

icpc.foundation

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 1
Lychrel Search

A Lychrel number is defined as a positive integer for which the following sequence never terminates:

- Take the number (for example, 57) and create its "reverse"; e. g. $57 \rightarrow 75$
- Add the number and its reverse together, resulting in a sum; $57 + 75 = 132$
- Repeat the process until the number is a palindrome; $132 + 231 = 363$

363 is a palindrome—a word, phrase, or number that reads the same backward as forward. 57 is not a Lychrel number because the sequence terminates after two iterations.

The above example is in base ten, but the process can be done in any base. If the base is a power of two, many Lychrel numbers are known; the smallest in base two is 10110_2 (22_{10}).

There is as yet no known Lychrel number in base ten. The smallest suspect is 196, and the reverse and add process is sometimes known as the 196 algorithm. It has been tested out to 10^9 digits.

Your team's job is to write a program to explore how many iterations of the reverse and add process it takes to get to a palindrome, given a starting number and a number base to be used. If a number is already a palindrome in the specified base, the iteration count is zero.

The input is a series of lines terminated by end-of-file. Each line has a test case, as two positive integers in base ten separated by spaces. The first integer is the base to be used, between 2 and 256 inclusive. The second integer is the starting number. It will be greater than zero, have fewer than 19 digits, and have no leading zeroes.

For each test case your program is to print a line with the number of iterations to reach a palindrome, one space, and the length of the palindrome. No leading or trailing whitespace or leading zeroes are to be printed on the line. If the iteration count is greater than 500, print a line containing only the string "`>500`".

Sample Input

```
10 56
10 4
10 89
10 10911
10 1997
12 58972
12 105789413
12 100100
2 196
2 22
13 2048
13 209847
127 86060307891903
128 999999999999999999
123 999999999999999999
```

Problem 1
Lychrel Search (continued)

Output for the Sample Input

```
1 3
0 1
24 13
55 28
>500
7 7
>500
9 8
1 8
>500
2 4
3 6
6 9
>500
2 9
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 2
Morse Redux

A Morse-like code is an assignment of sequences of dots and dashes to alphabet characters. You are to create a Morse-like code that yields the shortest total length to a given message, and return that total length.

A dot has length one. A dash has length three. The gap between dots and dashes within a character has length one. The gap between characters has length three. Spaces, punctuation, and alphabetic case are ignored, so the text:

`The quick brown dog jumps over the lazy fox.`

is encoded as though it were:

`THEQUICKBROWNDOGJUMPSOVERTHELAZYFOX`

For instance, for the input `ICPC`, one can encode the `C`'s with a single dot, the `I` with a dash, and the `P` with two dots, which has a total length of $3+3+1+3+1+1+1+3+1$ or 17. See Figure 1 below.

Input is a single line consisting of uppercase or lower case letters, spaces, commas, periods, exclamation points, and question marks. The line will have a maximum length of 32,000 characters and will contain at least one letter. Everything but the letters should be ignored.

Your program is to print a line containing only a single integer giving the length of the encoded string when an optimal Morse-like code is used.

Sample Input

```
ICPC
```

Output for the Sample Input

```
17
```

	I			*(letter gap)*			C	*(letter gap)*			P			*(letter gap)*			C	
	-						•				•		•				•	
	1	2	3	1	2	3	1	1	2	3	1	1	1	1	2	3	1	
Length	3			3			1	3			3			3			1	17

Figure 1. Chart showing letter lengths and spacing corresponding to the sample input.

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 3
Stairs

When building stairs in living spaces, local, national, and international building codes usually require all residential stairs to follow the same basic rules. This commonality allows pedestrians to ascend and descend stairs safely, regardless of setting. Your program will assist builders to bound the number and dimensions of stairs.

A single stair has two parts: the *riser* and the *tread*. See Figure 1 below. The vertical dimension of the riser is called the *rise*, r. The horizontal dimension of the tread is called the *depth*, d. The total vertical height is called *Total Rise*, and the cumulative length of all the treads is called the *Total Run*. In the United States, residential stair building codes recommend the following relationships for r and d.

$$r \leq 7.75 \text{ inches}$$

$$10.00 \text{ inches} \leq d$$

$$24.00 \text{ inches} \leq 2r + d \leq 25.00 \text{ inches}$$

When only one step up is required, the depth, d, and any constraints involving d do not apply.

Your program must read values of Total Rise, expressed in decimal inches, one value per line. Values for Total Rise may have up to two digits after the decimal point. From Total Rise, choose the minimum number of stairs, with equal rise for all steps and equal depth for all treads, that satisfy the above relationships.

For each value of Total Rise, print three numbers on a line, separated from each other by single spaces. The first number is an integer, S, denoting the minimum number of stairs required. The second number is the minimum Total Run, and the third is the maximum Total Run, both based upon S. Express Total Run values as inches rounded to and printed with two decimal places. When no stairs are required, print "0 0.00 0.00".

No leading or trailing whitespace should appear on an output line.

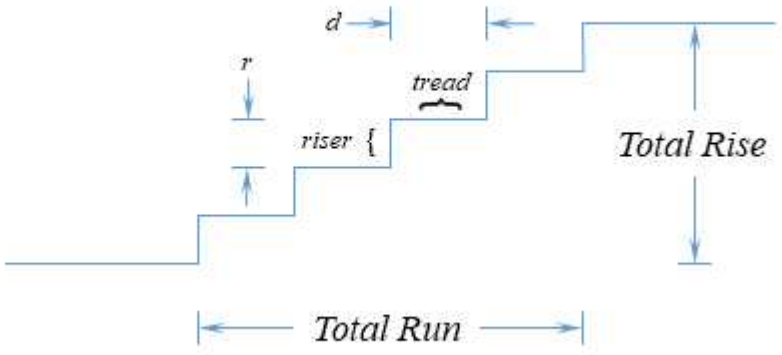

Figure 1. Although it takes 5 *steps* to traverse Total Rise, only 4 *stairs* are required.

Problem 3
Stairs (continued)

Sample Input

```
20
27.75
```

Output for the Sample Input

```
2 21.33 23.33
3 30.38 33.38
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 4
Computer Cache

Your computer has a cache consisting of n different addresses, indexed from 1 to n. Each address can contain a single byte. The ith byte is denoted as a_i. Initially all cache bytes start off with the value zero. Formally, the cache can be modeled by a byte array of length n that is initially all zeros. You have m different pieces of data you want to store. The ith piece of data is a byte array x_i of length s_i.

You are going to do q different operations on your computer. There are three types of operations:

- `1` i p: Load data i starting at position p in the cache. Formally, this means set $a_p = x_{i,1}, a_{p+1} = x_{i,2}, ..., a_{p+s_{i-1}} = x_{i,s_i}$, where $x_{i,k}$ represents the kth byte of the array x_i. This overwrites any previously stored value in the cache. It is guaranteed that this is a valid operation (e.g. $s_i + p - 1 \leq n$, and $1 \leq i \leq m$). It is possible for multiple versions of some data to be loaded in multiple positions at once.
- `2` p: Print the byte that is stored in address p.
- `3` i l r: Increment the lth through rth bytes in the ith piece of data, modulo 256. Formally, this means to set $x_{i,k} = (x_{i,k} + 1) \bmod 256$ for $l \leq k \leq r$. This does not affect values that are already loaded in the cache and only affects future loads.

Input to your program is as follows:

- The first line of input consists of three numbers n, m, and q, separated by whitespace.
- The following m lines define each $x_i, 1 \leq i \leq m$, as referenced in load and increment operations. The lines appear in order, with the first line representing x_1. Each of these lines starts with the byte count followed by the byte values, separated by whitespace.
- The following q lines consist of cache operations, one per line. The operations are as described above.

It is guaranteed there is at least one type 2 print query operation in the input. Additionally:

$$1 \leq n, m, q \leq 5 \times 10^5$$

$$\sum_i s_i \leq 5 \times 10^5$$

$$s_i \geq 1$$

$$0 \leq x_{i,j} \leq 255$$

Your program must output the results for each type 2 operation, one integer value per line.

Problem 4
Computer Cache (continued)

Sample Input

```
5 2 10
3 255 0 15
4 1 2 1 3
2 1
1 2 2
1 1 1
2 1
2 4
3 1 1 2
2 1
1 1 2
2 2
2 5
```

Output for the Sample Input

```
0
255
1
255
0
3
```

Explanation of the Sample Input

`2 1`	Nothing has been put into the cache, so print 0
`1 2 2`	The cache is now [0, 1, 2, 1, 3]
`1 1 1`	The cache is now [255, 0, 15, 1, 3]
`2 1`	Print the first value of the cache which is 255
`2 4`	Print the fourth value of the cache which is 1
`3 1 1 2`	The first piece of data becomes [0, 1, 15]. The cache is still [255, 0, 15, 1, 3]
`2 1`	Print the first value of the cache which is 255.
`1 1 2`	The cache becomes [255, 0, 1, 15, 3].
`2 2`	Print the second value of the cache which is 0.
`2 5`	Print the fifth value of the cache which is 3.

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 5
Swap Madness

You are given W, a set of N words that are anagrams of each other. There are no duplicate letters in any word. A set of words $S \subseteq W$ is called "swap-free" if there is no way to turn a word $x \in S$ into another word $y \in S$ by swapping only a single pair of (not necessarily adjacent) letters in x. Your team is to write a program that finds the size of the largest swap-free set S chosen from the given set W.

The first line of input contains an integer N ($1 \leq N \leq 500$). Following that are N lines, each with a single word. Every word contains only lowercase English letters and no duplicate letters. All N words are unique, have at least one letter, and every word is an anagram of every other word.

Your program is to print a line containing only the size of the largest swap-free set.

Sample Input 1

```
6
abc
acb
cab
cba
bac
bca
```

Output for Sample Input 1

```
3
```

Sample Input 2

```
11
alerts
alters
artels
estral
laster
ratels
salter
slater
staler
stelar
talers
```

Output for Sample Input 2

```
8
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 6
Maze Connect

Given an orthogonal maze rotated 45 degrees and drawn with forward and backward slash characters (see below), determine the minimum number of walls that need to be removed to ensure it is possible to escape outside of the maze from every region of the (possibly disconnected) maze.

The maze depicted below has only a single region fully enclosed. Removing any wall will connect it to the outside.

```
/\
\/
```

The maze below has two fully enclosed regions. Two walls need to be removed to connect all regions to the outside.

```
/\..
\.\.
.\/\
..\/
```

Input to your program consists of a series of lines. The first line has two numbers, R and C, giving the number of rows and columns in the maze's input description, separated by whitespace. Following this will be R lines each with C characters, consisting only of the characters '`/`', '`\`', and '`.`'. Slashes represent walls, and dots are placeholders to represent the absence of a wall. Both R and C are in the range $1 \ldots 1000$.

Define an odd (even) input location as one where the sum of the x and y coordinates is odd (even). Either all forward slashes will be in the odd input locations and all backslashes in the even input locations, or vice-versa.

Output on a single line an integer indicating how many walls need to be removed so escape is possible from every region in the maze.

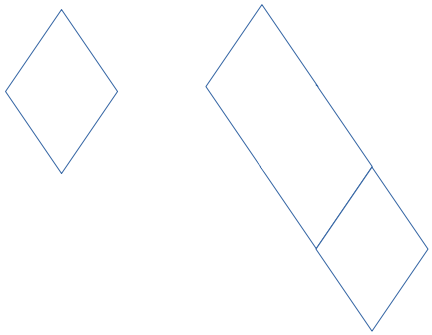

Figure 1. Drawings showing maze regions represented in the sample input.

Problem 6
Maze Connect (continued)

Sample Input 1

```
2 2
/\
\/
```

Output for Sample Input 1

```
1
```

Sample Input 2

```
4 4
/\..
\.\.
.\/\
..\/
```

Output for Sample Input 2

```
2
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 7
Flaw Removal

An artist who wanted to create an installation where his works appeared to be floating in mid-air has cast a large cube of clear acrylic to serve as a base. Unfortunately, during the casting, some small flecks of dirt got into the mix, and now appear as a cluster of pinpoint flaws in the otherwise clear cube.

He wants to drill out the portion of the cube containing the flaws so that he can plug the removed volume with new, clear acrylic. He would prefer to do this in one drilling step. For stability's sake, the drill must enter the cube perpendicular to one of its faces.

Given the (x, y, z) positions of the flaws, and treating the size of the flaws as negligible, what is the smallest diameter drill bit that can be used to remove the flaws in one operation? The drill may enter any one of the cube faces, but must be positioned orthogonally to the face.

The first line of input contains an integer N, $3 \leq N \leq 5000$, denoting the number of flaws. This is followed by N lines of input, each containing three real numbers in the range $-1000.0 \ldots 1000.0$, denoting the (x, y, z) coordinates of a single flaw. Each number will contain at most six digits following a decimal point. The decimal point may be omitted if all succeeding digits are zero.

Print a line with the diameter of the smallest drill bit that would remove all the flaws. The answer is considered correct if the absolute or relative error is less than 10^{-4}.

Sample Input 1

```
3
1.0 0.0 1.4
-1.0 0.0 -1.4
0.0 1.0 -0.2
```

Output for Sample Input 1

```
2.0000000000
```

Sample Input 2

```
5
1.4 1.0 0.0
-0.4 -1.0 0.0
-0.1 -0.25 -0.5
-1.2 0.0 0.9
0.2 0.5 0.5
```

Output for Sample Input 2

```
2.0000000000
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 8
Permits in Kafkatown

Getting a business permit in Kafkatown requires a trip to City Hall. There you are given a permit form that must be signed by K city clerks whose names are printed at the bottom of the form.

Entering the clerks' room, you find a long line of people working their way down a narrow aisle along the clerks' desks. The aisle is so narrow that the line is forced to shuffle forward, single file, past each clerks' desk in turn. Once in the line you cannot leave, back up, or change positions with other people. The desks are numbered sequentially.

As you present your permit for a signature, you are told that no clerk will sign unless all of the signatures above his or her name on the permit form have already been filled in. To your dismay, the clerks' desks are not arranged in the same order as the names on your form.

How many times will you need to pass through the line until you can get your permit? Your team is to write a program to determine this.

For example, assume you need signatures from five clerks, at desks number 1, 23, 18, 13, and 99. You will have to go through the line three times: the first time to get signatures from clerks at desks 1 and 23, the second time to get a signature from the clerk at desk 18, and the third time to get signatures from clerks at desks 13 and 99.

The first line of input contains an integer K, the number of signatures you need to collect, in the range 1 to 100 inclusive. This is followed by K lines of input, each containing an integer in the range $1 \ldots 100$, indicating the desk numbers of each of the clerks whose signature you need, in the order that they appear on your form. (Clerks whose signatures are not needed on your form are omitted from this list.) No desk number will appear more than once.

Your program is to print a single line containing only an integer denoting the number of passes you will need to make through the line in order to collect all of the signatures that you need. No leading or trailing whitespace, or leading signs or zeroes, are to be printed on the line.

Sample Input

```
5
1
23
18
13
99
```

Output for the Sample Input

```
3
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 9
Visibility

Given a terrain, that is, a map of points with their elevations, an interesting question is: From which point can you see the most other points or the fewest other points?

Let's start with a simple case in two dimensions, not three. Begin with a set of points in a plane, with p_i each at (x_i, y_i) ordered by x_i and each p_i connected by straight lines to p_{i-1} and p_{i+1} (if they exist). From which p_j can you draw the most straight lines to the other points without crossing one of the lines between the points? Alternately, from which point p_k can you draw the fewest?

The set of points and lines looks like a ridge line. See Figure 1.

One point can see another if the sight line to the other point is not blocked by terrain, that is, some line between points. A sight line is not necessarily blocked by a point though. For example, in Figure 1, p_0 can see p_1 and p_2, and p_6 can see p_8 and p_9. However while p_2 can see p_4, it cannot see p_8 because of the combination of the lines though p_3, p_4, and p_5.

A point can always see the adjacent point or points. A point cannot see itself.

Input to your program consists of lines with integers separated by spaces, terminated by end of file. The first integer is y_0; x_0 is implied, and is equal to 0. The next value is y_1 and is at $x_1 = 1$, and so on with each $x_i = i$. There will be at least 2 and at most 1,000 points and the value of each y will be between 0 and 10,000 inclusive.

Your program is to print two lines, the first is for the point that can see the most points, the second for the fewest. Each line has two integers separated by a single space and each integer has no leading spaces or zeroes and no trailing spaces. The first integer is the x coordinate of the point, the second is the number of points seen from it.

In case of ties, use the point with the lowest x coordinate.

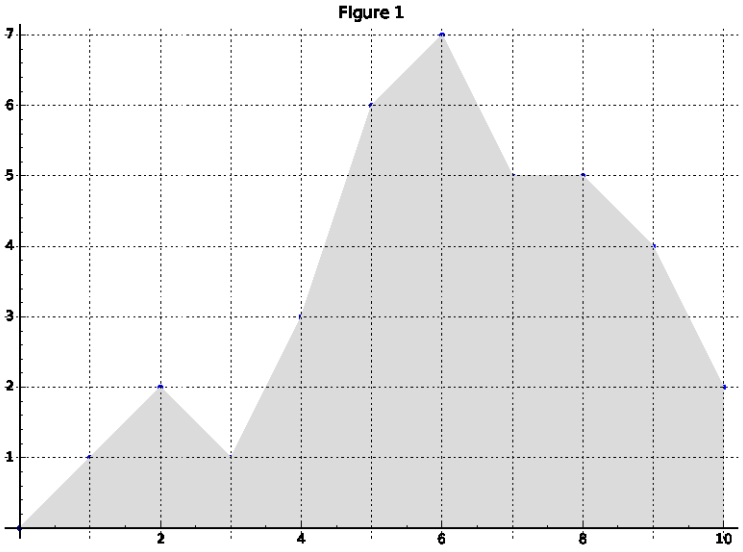

Figure 1. Diagram corresponding to the sample input.

Problem 9
Visibility (continued)

Sample Input

```
0 1 2 1 3 6 7
 5 5 4 2
```

Output for the Sample Input

```
5 6
10 1
```

2019/2020 SOUTHERN CALIFORNIA REGIONAL INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 10
Pixel Activation

An LCD panel is composed of a grid of pixels, spaced one alu ("arbitrary length unit") apart both horizontally and vertically. Wires run along each row and each column, intersecting at the pixels. Wires are numbered beginning with 1 and proceeding up to a panel-dependent maximum. The vertical wire numbered 1 lies along the left edge of the panel, and the horizontal wire numbered 1 lies along the bottom edge of the panel.

A pixel will activate, turning dark, when a current is present along both the vertical and horizontal wire passing through that pixel.

For a period of time, we will send pulses of current down selected wires. The current flows down the wires at a speed of one alu per atu ("arbitrary time unit"). The pulses themselves have a length measured in atus. A pixel activates when current is passing through both intersecting wires at the same time. If the leading edge of a pulse on one wire reaches the intersection at the exact same time that the trailing edge of a pulse on the other wire leaves that intersection, the pixel is not activated.

All pulses in vertical wires start from the bottom of the grid. All pulses in horizontal wires start from the left of the grid. At most one pulse will travel along any one wire.

Given the schedule of pulses to be sent through the wires, determine how many pixels will have been activated by the time all pulses have exited the top and right of the grid.

Input to your program is a series of lines. The first line contains n, the number of current pulses, with $1 \leq n \leq 200{,}000$.

Following this are n lines, each describing a single pulse. Each such line contains four elements, separated from one another by a single space:

- A single character that is either "`h`" or "`v`", indicating the horizontal/vertical direction of the pulse.
- An integer t, $1 \leq t \leq 200{,}000$, denoting the starting time of the pulse. The starting time is considered to be the moment when the leading edge of a vertical [horizontal] pulse crosses horizontal [vertical] wire #1.
- An integer m, $1 \leq m \leq 200{,}000$, denoting the length of the pulse.
- An integer a, $1 \leq a \leq 100{,}000$, denoting the wire number (horizontal or vertical) along which the pulse will travel.

Your program is to print on a single line the number of pixels that will have activated by the time the last pulse of current has left the grid.

Sample Input 1

```
4
h 1 4 1
v 2 4 2
h 10 2 2
v 11 2 3
```

Output for Sample Input 1

```
2
```

Problem 10
Pixel Activation (continued)

Sample Input 2

```
4
h 1 10 1
h 5 10 2
v 1 10 1
v 5 10 3
```

Output for Sample Input 2

```
4
```