Armenian Open Programming Contest

In memory of Vladimir Yeghiazaryan

May 23, 2021

Problems[1]

We thank our partners

РОССИЙСКО-АРМЯНСКИЙ УНИВЕРСИТЕТ

ONEMARKETDATA

ՁԵՌՆԱՐԿՈՒԹՅՈՒՆՆԵՐԻ ԻՆԿՈՒԲԱՏՈՐ ՀԻՄՆԱԴՐԱՄ ENTERPRISE INCUBATOR FOUNDATION

CODESIGNAL

[1] See the last page for general information about the problems.

Problem A

Remove a digit

You are given a 7-digit number. Delete one digit from it in order to get the maximum possible 6-digit number.

Input

The only line of the input contains one 7-digit integer N ($1000000 \leq N \leq 9999999$)

Output

The only line of the output must contain one 6-digit number without leading zeros.

Input	**Output**
9884726	988726
7878989	878989

Problem B

Are you lucky?

Armen bought a new electronic device and set a passcode on it, which is a 3 digit code. But after a month he, unfortunately, forgot the passcode. And the only detail that he remembers is that it consists only of digits ‘0’ and ‘1’.

Then Armen decided to find out how lucky he is and how fast he will find out the password. Thus, he unsuccessfully tried 001, 011 and after that the 3rd attempt was correct.

Now it’s your turn to find out how lucky you are? Can you find out Armen’s passcode?

Input

There is no input for this task.

Output

You should simply upload or paste Armen’s passcode in the online judge, on a single line, without spaces. Below is a (wrong) answer in the correct formatting:

```
010
```

Problem C

Prime chain

One day when traveling in an airplane(traveling was quite popular before 2020) Eduard and Hayk were playing an interesting game with numbers. Eduard was starting from some 1-digit prime number and at each step, one of them must add a new digit at the end of the current number in a way that the resulting number will also be a prime number. E.g. Eduard can start from number 2 then Hayk can add a digit 3 and get 23 which is still a prime number and a correct move from Hayk. Then Eduard can add 9 and get 239 which is again a prime number…. The player who can't make a valid move (can't add a new digit at the end in a way to get a prime number) will lose.

Hayk asked you to help him find a way to always win a game against Eduard.

Input/Output

This is an interactive problem and you will play for Hayk, and Jury's program will play instead of Eduard. Thus, Jury will start.

The first line of input will contain a 1-digit prime number (2, 3, 5, or 7), the first move. Then your program will need to print 2 digit number in the standard output in a way that it will be a prime number and will start with the digit given in the input.

After that Jury's program will print either -1, which means Jury can't make a move and you win, or it will print a 3-digit prime number after adding a digit at the end of the number that you printed and you will need to generate a 4-digit number, and so on.

If your program can't make a move, because it can't find a valid move, please print -1.

If jury outputs -1 and you win, please finish the program and don't read anything else from the input.

Do not forget to flush after each output. See the last page for more details about interactive problems.

`Jury's output`	`Your output`
`2`	
	`23`
`239`	
	`2393`
`-1`	

There is no way to add a digit at the end of 2393 to get a prime number.

Problem D

Media in Yerevan

You are the head of a media agency in Yerevan where major events happen all the time. But you need to get information about every event as soon as possible. So you have a big team of reporters distributed in the city and always ready on call.

The roads in Yerevan can be modeled as a connected graph with n intersections, and m bidirectional roads between intersections. There are reporters in exactly k of the intersections. During a month r events happen, and being the head of the media agency you want to find out for each event which reporter is the closest to the event location, so you can send them there as fast as possible.

For each of the events 1, 2, …, r, find which reporter is closest to the event location, and how far away they are. If multiple reporters are at the same distance, output the one with the lowest index.

It is guaranteed that you can get between any pair of intersections in Yerevan using the roads. It is also guaranteed that there is at most one reporter in each intersection.

Input

In the first line, there are four integers $n \leq 10^5$, $m \leq 10^5$, $1 \leq k \leq 10000$, and $1 \leq r \leq 10^5$.

Then follow m lines, each containing two integers $0 \leq a < n$ and $0 \leq b < n$, indicating that there is a road between intersection i and j.

Then follow k more lines, each containing an integer $p_i < n$ indicating that reporter number i is in intersection p_i.

Then follow r more lines, each containing an integer $b_i < n$ indicating that the i^{th} event happens in intersection b_i.

Output

Output r lines, each containing two integers d and b, the distance from the event to reporter's location which is the closest, and the index of that reporter. If multiple reporters are the same distance from the event, you should output the one with the lowest index.

Input	Output

3 2 1 3
0 1
1 2
0
0
1
2
	0 0
	1 0
	2 0

6 6 2 6
0 1
1 2
2 3
3 4
4 1
3 5
0
5
0
1
2
3
4
5
	0 0
	1 0
	2 0
	1 1
	2 0
	0 1

Problem E

Variable names

Sergey is a newly hired junior developer in OMD. In his first big project, he created n variables with long names of lengths between 1 and 300. Since the names are very long, senior developer Armen suggested changing the names. However, the requirement from Armen is that the new names must be non-empty subsequences of the initial name, and must also be pair-wise distinct (however the old names are not necessarily distinct). Your task is to choose names for them, such that the longest new name is as short as possible.

A subsequence of a string $s=s_1s_2...s_k$ is obtained by deleting some (but not all and possibly none) of the characters of the string. Formally, it's any string of the form $s_{m1}s_{m2}...s_{ml}$ where $1 \leq m_1 < m_2 < ... < m_l \leq k$ is an increasing sequence of integers in $[1, k]$.

Input

The first line consists of an integer $2 \leq n \leq 300$, the number of variables.

Then follows n lines containing one string of between 1 and 300 each, denoting the name of i^{th} variable. Each name consists only of lower-case English letters.

Output

If there is no solution, output -1.

Otherwise, output $n + 1$ line, with the first line containing an integer m, denoting the length of the longest new name, and the $(i+1)^{th}$ line containing the new name of the i^{th} variable.

If there are multiple solutions, you may output any one of them.

Input	**Output**
4 aw we shu le	1 a w u l

Problem F

Building bridges

There are n islands in Budai. And the Budai government wants to build some new bridges to make all islands connected. Each island has an official name A_i, and an unofficial name B_i. The beauty of a bridge between islands i and j is $w=|LCP(A_i,B_j)|+|LCP(B_i,A_j)|$, where LCP denotes the longest common prefix of two strings, and $|S|$ denotes the length of a string S. The overall beauty of a plan is the sum of beauty of all the bridges in the plan. The government wants to know the highest possible overall beauty of Budai if it's connected using the least possible number of bridges.

It is guaranteed that $\sum|A_i|+|B_i| \leq 10^5$. It is also guaranteed that A_i and B_i only contain lowercase English letters.

Input

In the first line, there is an integer $n \leq 10^5$.

Then follow n lines, the i^{th} of which contains the string A_i.

Then follow n more lines, the i^{th} of which contains the string B_i.

Output

Output a single line containing an integer, the maximum overall beauty.

Input	Output
3 aaa bbc bb bbb ac bbc	6

Problem G

Math quiz

You are given a list of statements, which are either true or false. You need to find out which is correct which is incorrect and as output send the file with true/false-s. The N-th line of the output file should be true if the N-th statement is correct and false otherwise.

Input

There is only one input file for this task, with the list of statements. Alternative link

Output

You should simply upload or paste the *output file*. Your output should contain 14604 lines. Below is the beginning of the answer file:

```
true
true
false
true
...
```

Problem H

Shortest click-path

In computing, a *hyperlink*, or simply a *link*, is a reference to data that the user can follow by clicking or tapping. If you are reading this task, then probably you have already got to this page by opening ejudge.rau.am website, then clicking on the link for the contest page then clicking *login,* and then clicking the link to the problem statements. So you probably did 3-4 clicks in order to get to this page.

When surfing the World Wide Web you do a lot of clicks going from one page to the other. Sometimes it's possible to minimize the number of clicks if you know the shortest path from the page you are to your target page. But that isn't always trivial.

For example, if you go to this start_sample page, and suppose you want to get to the page target_sample.html. You will need to **click** the first link (first.html), then in the opened page **click** the second link (second.html), and then **click** on the first link (target_sample.html). So totally you will need to do 3 clicks. But there is another shorter path as well. You can **click** the second link (second.html) when you are on the first page, and then **clicking** on the first link will get you to the targeted page, and you will do just 2 clicks.

In the scope of this task, you need to find all pages that are accessible from the given page by clicking the links and also find out the length of the shortest click-path from the start page to each of these pages.

Input

All pages are hosted in ejudge.rau.am and initially you are on the start.html page.

Alternative link

Output

You should simply upload or paste the *output file*. The first line of the output file should contain the number of accessible pages from start.html (including *start.html* itself). Each of the next lines must contain the full path of the page and the minimum number of clicks that one needs to do to get from start.html to that page.

All pages should be listed in lexicographic order. Below is the correct answer for the start_sample.html.

```
5
https://ejudge.rau.am/first.html 1
https://ejudge.rau.am/hello.html 2
https://ejudge.rau.am/second.html 1
https://ejudge.rau.am/start_sample.html 0
https://ejudge.rau.am/target_sample.html 2
```

Problem I

Count the logos

As you know today's competition wouldn't be possible without our partners, and we would like to thank them once more for helping us organize Yeghiazaryan Cup annually.

In order to thank them, we decided to prepare a page with their logos. As you know we have 4 partners - RAU, EIF, OMD, CodeSignal. And we prepared the following page, which is a very simple game. There are many buttons on the page, each of which has some hidden company logo in it. You need to click the button and the logo will open. Try clicking some of the buttons, and you will see the beauty :)

But you probably already noticed that there are too many buttons and it's a little bit hard to click all of them. But you better try to do that, because in the scope of this task you need to count the number of logos of each company.

Input

There is no input for this task. The only input is this game page.

Alternative link

Output

You should simply upload or paste the *output file*. Below is a (wrong) answer in the correct formatting. The correct output should also contain 4 lines in this format (and in this order), just numbers should be different.

```
OMD: 3
EIF: 4
RAU: 7
CodeSignal: 7
```

Problem J

Trees for the tent

In Yerevan, there is an interesting park with n trees, no three of which are collinear. The Mayor of the city wants to organize an event for the city day in the park. For the event space, they want to find 3 trees in the park in a way that they will form a triangle, but there won't be any other tree inside. After that, they will use these 3 trees as a base for a large tent and organize an event inside the tent.

Help mayor to choose such three trees that the triangle formed by these trees don't have any other trees in it!

Input

In the first line, there is an integer $3 \le n \le 10^5$.

Then follow n lines, each containing two integers $0 \le x \le 10^9$ and $0 \le y \le 10^9$, indicating the position of the i^{th} tree. It's guaranteed that no three trees are collinear. Also, there is at most one tree in a point.

Output

Output three lines, each containing two integers x and y, indicating the position of a tree which the mayor should use for his tent. If there are multiple solutions you can output any of them.

Input	Output

4
0 0
0 5
5 0
1 2
	0 0
	0 5
	1 2

Problem K

Transposition

Given an integer N. Find a transposition a of the first N positive integers such as for any $i > 1$ sum $a_1 + a_2 + . . . + a_{i-1}$ is equal to 0 modulo a_i .

Input

The input file contains one integer n $(2 \le n \le 100)$ number of elements in transposition.

Output

Print n space-separated pairwise distinct integers between 1 and n - a requested transposition. If there exists more than one correct solution, print any of them.

Input	Output
5	4 1 5 2 3

Problem L

Car museum

A car museum is one of the few places where car fans can see supercars and legendary cars. Such museums often have many cars. Sometimes some of the cars need to be moved (either to other museums or for restoration). And these cars must be moved very carefully.

Employees only wish to move cars if they really have to. So, given a map of a museum including its walls, doors and where the cars are located, and the coordinates of the car to move, how many cars must be moved in total?

It's possible to rotate cars on the spot, but they can only be moved through completely empty space and not diagonally. All the doors are wide enough to move a car through.

Input

The first line contains two integers R, C ($3 \leq R, C \leq 400$), the size of the museum in rows and columns.

- R lines follow, each containing a string of C characters with the following meaning:
  - `’#’: a wall;`
  - `’c’: a car;`
  - `’D’: a door in a wall.`

  The first and last lines must be either walls or doors. The first and last characters in a row must be walls or doors as well.
- The next line will contain two integers r ($1 < r < R$), and c ($1 < c < C$), the co-ordinates of the car to move. 1, 1 is the top-left corner.

Output

Output one line containing one integer: the smallest number of cars that need to be moved (including the car we are moving) to allow our desired car to leave the museum.

Input	**Output**

`4 5`
`#####`
`#cDc#`
`#c#cD`
`#####`
`3 2`
	`4`

10 10 ########## #cc#ccccc# #cc#cccccD #cccccccc# ########c# #cccccccc# ###cccccc# #c#cccccc# #cccccccc# ########## 2 2	11

Problem M

Mini judge

If you have participated in competitive programming contests in the past you are probably familiar with some online judge systems as pcms or pc^2. Or at least you are familiar with ejudge, the platform that is used for this competition.

Developing such a powerful system isn't a quick development task. But in the scope of this task, you are asked to develop a limited version of the online judge system.

You are free to use any technology you will consider useful because the requirement of this task is to find out which of the given submissions solved the given task (got OK), which don't.

More formally - you are given a ***submissions.zip*** file that contains all submissions that were done during the contest. Each submission file name has the following format

`submissionid-taskid.language`

`Where`

- `submissionid` - is a unique id of the submission, which is a 6 digit number possibly with trailing 0s.
- `taskid` - is the id of a task that this submission is trying to solve. It's a letter from the range ['A' - 'Z']
- `language` - is the shortcode of the programming language used for the submission. Supported languages are
    - cpp - stands for C++, and should be compiled using g++ compiler with C++11 flag turned on
    - py - stands for python, and should be executed using python3

Also, you are given a ***problems.zip*** file that contains information about the tasks. It has folders named `A, B, C, …., Z` Tasks can be any subset of `{A, B, … Z}.` E.g. in the case of 4 tasks, there can be folders names A, C, D, G.

Each folder contains input and output test cases for that task in one of the following two formats

```
001.in                    001
001.out                   001.a
002.in                    002
002.out                   002.a
...
```

Where `.in` files (or without extension) are input tests for the task, that you need to pass to stdin for the submissions, and the corresponding `.out (or .a)` file is the correct output for that test.

Submission will be considered ***correct*** if it will execute on each of the given tests successfully(compiles successfully, finishes in 5 seconds, uses less than 512MB memory, and returns a success code - 0) and print corresponding correct output in stdout. Otherwise, the submission ***fails***. When comparing user output with the expected output, please ignore the trailing spaces at the end of the lines and trailing newlines at the end of the output files.

Let's look into the following sample

[sample_submissions.zip]

[sample_problems.zip]

You can see 2 submissions here `000001-A.cpp` and `000002-A.cpp` and there is only one task folder `A`, with only one input test `001.in` and `001.out` files.

As you can see the first submission compiles and after execution, it prints the sum of 2 numbers given in the input, which is correct `(1 + 2 = 3) and 001.out` contains number 3, so submission is **correct.** While second submission prints multiplication of the numbers `(1 * 2 = 2)` which isn't equal to the `001.out` file content, so the second test fails.

Output for this sample test will be

```
000001: OK
000002: FAIL
```

Some key points from the above and some more details:

- Use g++ 7.3 compiler (or higher) with C++11 enabled. When compiling using Windows, please make sure that you use the same compiler, otherwise, your verdict may differ (e.g. it will fail compilation but using g++ it won't fail, or vice versa)
- Use python 3.6 or higher when executing python solutions.
- Submissions read input from `stdin` and print output into `stdout` stream.
- When comparing user's output with correct output ignore trailing spaces and trailing newlines.
- If the correct output (expected output) is a double number then you need to compare numbers with 3 digits after decimal point
- The time limit for all tasks is 5 seconds. If submission runs longer than 5 seconds it should be considered as failed `(time limit exceeded verdict)`.
- If submission uses more than 512MB memory it should be considered as `failed (memory limit excited verdict)`.
- If submission returns non zero code it should be considered as `failed (run time error verdict)`

There are 2 versions of this task - M1 and M2. The first one is the easy version because it has additional restrictions.

Problem M1

This task has the following additional restrictions

1. `problems.zip` contains only one task A, with only one test case `001.in and 001.out`
2. `submissions.zip contains only 30 submissions all in c++`

Input

There is no input for this task. Inputs are 2 zip files:

[submissions_m1.zip]
[problems_m1.zip]

Output

You should simply upload or paste the *output file*. Below is a (wrong) answer in the correct formatting. Each line of the output file should contain submission id (in increasing order), colon, space, and either word OK or FAIL. Here is the example of the output for the sample test described above.

```
000001: OK
000002: FAIL
```

Problem M2

This version contains more than 1 task in problems.zip and submissions are done not only using c++ but also using python.

Input

There is no input for this task. Inputs are 2 zip files:

submissions_m2.zip
problems_m2.zip

Output

You should simply upload or paste the *output file*. Below is a (wrong) answer in the correct formatting. Each line of the output file should contain submission id (in increasing order), colon, space, and either word OK or FAIL. Here is the example of the output for the sample test described above.

```
000001: OK
000002: FAIL
```

General information about interactive and stdin/stdout problems

- The time limit for all interactive or stdin/stdout problems is 2 seconds and the memory limit is 128MB.
- All input/output should be done from standard input and output. You can find examples of standard input/output in common languages here.

General remarks about interactive problems

1. You must print a new line after each interaction;
2. You must flush the output stream after each interaction:
   - `In C or C++: fflush(stdout);`
   - `In Java: System.out.flush();`
   - `In Python: sys.stdout.flush()`
   - `In C#: Console.Out.Flush();`
3. If your program receives EOF (end-of-file) condition on the standard input, it must exit immediately with exit code 0. Failure to comply with this requirement may result in Time Limit Exceeded error instead of other verdicts AC/WA/PE.
4. Typical issues with interactive problems are
   - *Wrong Answer* – usually means that your program followed the interaction protocol but the answer or the intermediate steps are wrong.
   - *(Wall) Time Limit Exceeded* – this means that due to your program's execution, the interaction is not progressing. This can happen
     1. if your program is expecting input from the jury's program by mistake. Most commonly this happens when the jury's program has detected an error and quit, but your program is waiting for input. In this case, you will get the TLE verdict, instead of the actual WA/PE error. See point (3) above;
     2. if your program has not provided the necessary output for the jury's program to respond. Most often this is because you have not flushed the output stream. See point (2) above.
   - *Presentation Error* – usually means that your program did not follow the interaction protocol correctly and the jury's interacting protocol is not able to test it.
   - *Runtime error* – usually a mistake in your program that makes your program crash during the execution.