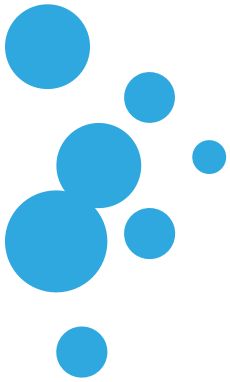
# STEM Games 2019



Technology

# SOLVING COMMUNICATION

designed by

Roko Gudić, Marko Ivić, Vanessa Keranović, Zvonimir Kučiš, Andro Merćep, Andro Mikulić, Filip Paradžik, Igor Rinkovec, Filip Šuste, Ana Škaro, Alen Štruklec, Robert Vaser, Lovro Vrček

# Day 1

## Cipher

For a given ciphertext $m$, decrypt it in order to find its plain text. The input consists of one line representing the ciphertext $m$. The output should be the decrypted plain text. **Difficulty**: Entry.

### Examples

### Input

kszqcas hc ghsa uoasg hsqvbczcum ofsbo

### Output

welcome to stem games technology arena

## Parachute

A parachute jumper jumps from a plane starting at coordinates $(0, 0)$. The wind randomly changes directions in a way that he can land anywhere within $r$ unit radius. In those $r$ unit radius you are given radii of $m$ islands. Islands are circles and all of the islands are within the $r$ unit radius where the jumper can land. Calculate the probability that the parachute jumper will land on water. First line of input represents the radius $r$ and number of islands $m$ separated by space. Second line contains the $m$ radii of circles that represent islands, separated by space. The output should be one value rounded to two decimal places, representing the probability. **Difficulty**: Easy.

### Examples

### Input

4.20 2
0.53
0.89

### Output

0.94

### Input

2.72 5
0.34
1.53

0.29
0.11
0.23

**Output**

0.65

**Input**

69.69 13
64.09
2.88
0.75
1.02
5.14
0.42
3.19
4.50
0.54
2.14
1.18
0.41
1.68

**Output**

0.14

## Keys

For a given key, represented with a string, find the most similar key in a list of $m$ different keys. The first line of input is the value $m$ and the query key separated by a space. Each succeeding line represents one of the $m$ target keys. The output should be one value which denotes the minimal zero-based index of the most similar target key. **Difficulty**: Easy.

**Examples**

**Input**

```
2 [{
[/
[\
```

**Output**

0

**Input**

```
3 ]%(
*[*
]]%%(
%+$
```

**Output**

1

**Input**

```
12 -}&{&*+#%
}}&#%%
-%
&{&+%
&&&+
}}&{&*+##%
&{&*+#%
-}{&%
(&**+
--&{{&**+#
&&{{&**
#%%
--&&%%
```

**Output**

4

## Boxes

For given $m$ boxes a robot fills or empties them in a predefined pattern. The robot moves from the first box to the last and then backwards, each of which represents an iteration. In the i-th iteration, the robot stops at every i-th box and inspects its contents. If the box is empty, it places an object in it, otherwise it removes the object from it. All boxes are empty at the beginning. Find the number of boxes containing an object after $n$ iterations. The input represents values $m$ and $n$ separated by a space. The output should be one value representing the number of boxes that contain an object. **Difficulty**: Easy.

**Examples**

**Input**

1 2

**Output**

1

**Input**

10 4

**Output**

6

**Input**

500 125

**Output**

193

## Cities

For a given list of $n$ roads spread between $m$ cities, find the number of trade routes. Trade routes are groups of cities with the following property: each city of a trade route can be reached via roads from any other city in the same route. Afterwards, in every of $k$ time units one of the roads gets destroyed and the number of trade routes has to be recalculated. Find the average number of trade routes, starting from the initial setting up to the last road removal. The first line of input contains values $m$, $n$ and $k$ separated by a space. Succeeding $n$ lines consists of two integer values separated with a space representing one road connecting two cities. Last $k$ lines contain one integer which denotes the zero-based road index that is destroyed. The output should be one value rounded to two decimal places, representing the average number of trade routes.
**Difficulty**: Normal.

**Examples**

**Input**

2 1 1
0 1
0

**Output**

1.50

**Input**

5 2 1
0 1
0 2
1

**Output**

3.50

**Input**

17 33 22
2 8
2 14
10 11
2 13
9 15
13 16
3 11
2 5
1 3
15 16
5 8
7 13
0 8
8 14
5 13
3 9
7 16
4 13
1 2
10 15
0 7
5 9
1 9
6 13
11 15
0 6
4 7
14 15
6 15

1 11
4 16
6 9
1 5
3
32
30
7
31
19
18
21
0
24
12
17
22
16
8
14
1
10
26
2
29
15

**Output**

2.91

## Land ho

In a given $mxn$ matrix containing positive integers that represent terrain heights, and a threshold value $s$ for the sea level, find the number of islands and seas. Every area greater than the threshold value represents islands. 4-neighbor and 8-neighbor rules apply for islands and seas, respectively. The first line of input represents values $m$, $n$ and $s$ separated by a space. The second line contains $m * n$ space separated values of the terrain matrix. The output should consists of two values separated by a single space denoting the total number of islands and seas. **Difficulty**: Normal.

**Examples**

**Input**

2 1 0
0 1

**Output**

1 1

**Input**

3 3 3
3 3 3 6 6 6 1 1 1

**Output**

1 2

**Input**

5 7 42
100 0 0 100 100 0 100 100 100 100 0 0 100 100 100 0 0 100 0 0 0 0 0 0 0 0 0 0
100 0 0 0 0 0 100

**Output**

6 1

## Modulo

For a given number $m$ find the greatest number that its modulo of 8 equals 5. That number can only be obtained by permutation of the digits of number $m$. The input represents value $m$. The output should be one value representing the number described above. If such number does not exist, return -1. **Difficulty**: Normal.

**Examples**

**Input**

31

**Output**

13

**Input**

1111

**Output**

-1

**Input**

361542

**Output**

654213

## Prime

For a given number $m$, find the Lucas number modulo 1000000000 of the first prime number greater than $m$. The input represents value $m$. The output should be one value representing the number described above. **Difficulty**: Normal.

**Examples**

**Input**

3

**Output**

11

**Input**

54

**Output**

295485799

**Input**

4269

**Output**

55882399

## Fence

Squirrel is looking for nuts. It has found the farm filled with trees. But this farm is surrounded by an electric fence. However, some nuts have fallen to the ground and squirrel can pick them up without harm. Problem is that the squirrel is so hypnotized by the sudden discovery of food that it forgot about the dangers of the fence. Luckily, you can help it. Tell the squirrel, how many times does it need to make a turn, while running around the fence, in order not to accidentaly run into it. Constraints are:

- all turns are by 90 degrees.
- the first input line contains an integer $n$: number of straight segments in the field.
- following $(n + 1)$ lines contain pairs of integers $(x_i, y_i)$: representing the vertices of the fence. The i-th straight segment of the field begins at the point $(x_i, y_i)$ and ends at the point $(x_{i+1}, y_{i+1})$.
- the first straight line segment is always directed to the north.
- the most southern point is the starting point (in case of tie: the most western among them).
- no pair of straight segments of the field has shared points (except for the neighboring points that share exactly one!).
- points are never the same (except for the first and last one).
- adjacent segments can never be directed in the same direction or in the oposite directions.

The task is to print a single integer the number of turns squirrel has to make in order not to run into the electric fence. **Difficulty**: Normal.

### Examples

#### Input

```
6
0 0
0 1
1 1
1 2
2 2
2 0
0 0
```

#### Output

```
1
```

**Input**

16
1 1
1 5
3 5
3 7
2 7
2 9
6 9
6 7
5 7
5 3
4 3
4 4
3 4
3 2
5 2
5 1
1 1

**Output**

6

## Tables

Your friend is organizing the pub quiz. Much to their surprise, people have applied, and your friend asked you to help with the tables organization. Members of one team want to sit together. However, there are so many people, and some tables can fit more people than there are in some teams. That is why you decided to seat multiple teams at the same table as long as there is place left. Tables in the pub are labeled with numbers in a such way that better tables come first. Your friend asked you to prioritize, so teams that applied first get better tables. For each team that applied you know how many members it has. After you assign the table to the team, available seats number at that table decreases. First line of input contains $n$ and $m$ (integers): number of tables in the pub and number of groups applied. The second line contains $n$ integers $t_1$, $t_2$,..., $t_n$, which represent the number of free seats on each table. The third line contains $m$ integers $g_1$, $g_2$,..., $g_m$. These are numbers of members in each team. Print the number of the assigned table for each team. If they cannot fit to any table, 0 should be printed. **Difficulty**: Normal.

**Examples**

**Input**

3 5
3 2 5
2 1 4 2 6

**Output**

1 1 3 2 0

**Input**

9 6
3 2 4 1 5 5 2 6
4 4 7 1 1 5

**Output**

3 5 0 1 1 6

# Day 2

## More boxes

For given $m$ boxes, a robot circulating over the boxes disintegrates every $n$-th box until only one box is left. Find the box which is intact at the end. After a box is disintegrated, the process starts from the next intact box. The input represents values $m$ and $n$ separated by a space. The output should be one value representing the zero-based index of the intact box in regards to the starting point. **Difficulty**: Normal.

### Examples

**Input**

3 2

**Output**

2

**Input**

7 4

**Output**

1

**Input**

7070 420

**Output**

5691

## More keys

For two given key molds, represented with strings, find and merge the largest series of pieces present in both molds. The first line of input is the first key mold, while the second line represents the second mold. The output should be one string which represents the concatenated largest series of equal pieces between input key molds. If there is a tie, pick the alphabetically smallest series. **Difficulty**: Normal.

**Examples**

**Input**

```
$%)}{
##%({}
```

**Output**

```
%{
```

**Input**

```
$)&&[}[
}})$({%
```

**Output**

```
$
```

**Input**

```
}({)[%%#{&[}[(]
{{%{#&#]%#(}}&]&)#&($){
```

**Output**

```
{%#&(]
```

## Lava

You find yourself in a lava pit. You want to get out. In this environment everything happens fast - new rocks are emerging from the cooling lava, and you can almost see a way out. However, just as you make one step toward the exit, the rocks that marked your path are melted away and some new ones are created. Interestingly, there is a pattern to this behavior. Can you memorize the pattern and calculate how many times do you need to jump to get to the exit, without the rock floor disappearing just underneath you? Or maybe, there is no way out? The rules are:

- start position is in the top left corner.
- end (goal) position is in the bottom right corner. You are safe once you reach this position.
- each step you can choose to stay at current position, or move up, down, left or right.
- each step the lava pit layout changes, following a circular pattern.

- first line of input are three integers $m$, $n$ and $p$, separated by a space character.
- following lines represent lava pit layout, where . marks the rock (safe) and | marks the lava (not safe). There are no spaces between those characters.
- input can consist of multiple cases, one after another.
- line where $m = n = p = 0$ marks the end of input and should not be considered as a case.
- output should be printed as one line, containing case number, then :, followed by the number (integer) annotating how many steps are required for you to get out of the lava pit, or - character if there is no way out. Cases are separated by ;. **Difficulty**: Normal.

**Examples**

**Input**

```
3 3 3
...
||.
...
..|
|..
...
|.|
||.
...
3 3 1
...
...
...
2 2 1
..
..
2 2 2
.|
|.
..
|.
0 0 0
```

**Output**

```
1:5;2:4;3:2;4:2;
```

**Input**

5 5 1
.....
.....
.....
.....
.....
5 5 2
.....
.....
.....
.....
.....
.|||.
.|||.
|||||
.|||.
.|||.
0 0 0

**Output**

1:8;2:-;

## Brackets

You are given $n_1$ pairs of round brackets (), $n_2$ pairs of square brackets [], and $n_3$ curly brackets {}. Your task is to find all valid mathematical expressions created by those brackets. For example, {[]} is a valid expression and expression {[}] is not. However, to make things simpler, you dont need to think about "strength" of the bracket types, e.g. both {()} and ({}) are valid. **Difficulty**: Normal.

**Examples**

**Input**

3 0 0

**Output**

5

**Input**

2 1 0

**Output**

15

**Input**

2 2 2

**Output**

11880

## Sphynx

As you wander through the maze you run into a sphynx. As we all know sphynxes love riddles, and so it will let you pass only if you find answers to all of its questions (*Otherwise, you shall not pass!*). However, this one has developed a peculiar love for mathematics. Therefore all questions are posed as mathematical equations of the following form: $x_n = a*x_{n-1}+b*x_{n-2}+c*x_{n-3}$. For each question you have to find the $n$-th term $x_n$ in the aforementioned equation and sum them all up. First line of input are terms $x_0$, $x_1$ and $x_2$ (in that order), which are constant for each question. Second line of input is the number of questions $m$. Next $m$ lines of input consist of 4 numbers: parameters $a$, $b$, $c$, and a number $n$ for the term you seek. Calculate the sum of those terms and output the sum modulo 1000000009. **Difficulty**: Normal.

**Examples**

**Input**

1 1 1
1
1 2 3 3

**Output**

6

**Input**

1 0 1
2
1 2 3 4
0 0 1 3

**Output**

7

**Input**

1 4 2
5
2 -3 1 1
-1 4 -4 6
8 3 5 22
23 54 18 7
23 41 24 30

**Output**

320143810

## All the boxes

A robot found himself in a warehouse with infinite number of shelfs that are infinitely long. His job is to stack boxes onto that shelfs from the first shelf up and from the begining of each shelf, stacking boxes one after the other, in a predefined pattern. Each box arriving at the warehouse has an identifier which equals the one-based index in the infinite shipment. The robot places each box at the end of the first shelf in which the identifier of the last box placed on that shelf summed with the identifier of the current box makes a perfect square. If no such shelf with already placed boxes exists, the new box is placed on the first empty shelf, occupying the first position on that shelf. For given $m$-th shelf and $n$-th position on that shelf, find the box identifier occupying that position. The input represents values $m$ and $n$ separated by a space. The output should be one value representing the one-based index of the box placed on the $m$-th shelf and $n$-th position on that shelf. **Difficulty**: Hard.

**Examples**

**Input**

1 1

**Output**

1

**Input**

1 3

**Output**

6

### Input

4 1

### Output

8

## All the keys

For a given key meld, represented with a large string, find the most frequent key part of length $m$. The first line of input is the value $m$ (not greater than 29) and the key meld separated by a space. The key meld consists of only four characters, which are #, $, % and &. The output should be one string which represents the most frequent key part. If there is a tie, pick the alphabetically smallest. **Difficulty**: Hard.

### Examples

### Input

```
1 %&$#
```

### Output

```
#
```

### Input

```
3 %#$$#%#$#%$%###%##
```

### Output

```
#%#
```

### Input

```
6 &%$##&$&$%#&%%$&%%#$##&%#$##&$&#&###$$&#&$$#%&%##&$&$#$#$#&%%#%&$&$#%
%&%$$$$%#%%$$&&$%#&%$#&#$$##$%#%$#&$$&$&%&$%%&$#$&#%#$$%%$#&&&$%&%#
%&&%$&$&$#$%$#$$#$&%%$%&$&$&#%#%&##%$#&#%$###&#&%%#&&&#%#&%%$$%&&&&$$$
$$%&##%%$#$&&&&%%%#%%###$%%&%##$&&$&$%&&&&#$&&&$%%&#&%#$$#&%&$$###$#$
&#&#$$$$#$&&&$%&$%%#&$&##$%##&%$#$&$#$$&&$&$&$#$#%$%%%%&&&&$%%%#%#$
#%#&##&%%&%%%$&&%#$$%&%%#%###&&%&&#&&$#$%%$#%$$&#%$##$%#$%#&%&#$$&%%$
%&$#$$$#%%&#%&#$&%&&
```

**Output**

```
&$&$#$
```

## Optimus

Let $n$ be a positive integer and $S$ a subset of distinct positive integers all greater than 1 and less than $n$, and none of them share a common divisor except 1. Find the maximum sum the set $S$ can have. The input represents value $n$. The output should be one value representing the maximum sum the set $S$ can have. **Difficulty**: Hard.

**Examples**

**Input**

6

**Output**

12

**Input**

24

**Output**

125

**Input**

76

**Output**

906

# Day 3

User experience is paramount these days. For mobile phone users this often means getting and receiving their messages seamlessly and not getting unsolicited and unwanted messages from third-parties. Those unsolicited messages can often incorporate phishing attacks or other form of illegal activity, which makes detection of these messages hugely important in domain of security, as well as user experience. Solution to this problem is development of spam filters. While spam detection in e-mail traffic is a well documented and mature endeavour, SMS spam detection poses different kinds of problems. As you may know, SMS messages are restricted by the number of characters, they lack structured fields and their text is rife with abbreviations and idioms. Typical spam detection techniques that work on large texts usually found in e-mails fall short when applied to SMS trafffic.

Furthermore, SMS traffic can be divided into P2P and A2P traffic. Usually, from the user's perspective, P2P (Person to Person) traffic is considered non-spam (or ham, as we call it) and A2P (Application to Person) traffic is more prone to be classified as spam. However, when you are in a B2B (Business to Business) company, every SMS is A2P.

Infobip is one of the world's largest A2P traffic providers and can be viewed as a bridge between businesses and MNOs (Mobile Network Operators). Two thirds of Earth's population interacted with Infobip's platform, which enables businesses around the Globe to send traffic to their clients. This makes spam detection even more tricky. In this task you'll get a dataset that looks something like this:

| Text | Label |
|---|---|
| `Sample_text_1` | `spam` |
| `Sample_text_2` | `ham` |
| `Sample_text_3` | `ham` |
| `Sample_text_4` | `spam` |

Your task will be to build a binary classifier to detect whether an SMS is spam or ham. You will be provided with a training dataset on which you'll build your classifier. Your model will be evaluated with F1 score, since this metric is best suited for inbalanced sets (negative class is over-represented).

**Note**: Most URLs are masked and replaced with generic url.com placeholder beacuse of GDPR. Majority of the names and numbers are randomized and anonymized for the same reason.

**IMPORTANT NOTE**: This dataset is private property of Infobip, Ltd and is provided solely for puropses of this competition. Any use, distribution or any kind of activity on this dataset which falls outside of scope of this competition is prohibited and can be opened to legal prosecution if terms are breached.