Irish Collegiate Programming Competition 2017

Problem Set

University College Cork ACM Student Chapter

March 25, 2017

Instructions

Rules

- All mobile phones, laptops, and other electronic devices must be powered off and stowed away for the duration of the contest.
- The only networked resource that teams are permitted to access is the submission system.
- If a team discovers an ambiguity or error in a problem statement, they should tell an organiser. If the organisers agree that an ambiguity or error exists, a clarification will be issued to all teams.
- No multi-threading is allowed, and no sub-processes.
- No file input/output is allowed. All input to your program will be provided via standard input (stdin) and all output should be printed to standard output (stdout). Examples of how to do this in each of the languages is provided in the resources section of the submission site.

Submission Instructions

- Your password will be provided by the organisers. Please notify an organiser if you are unable to log in.
- Submissions should consist of a single source file, **not a compiled executable**.
- To submit, click the "Submit a Solution" link, complete the submission form, upload your source file, and click "Save".
- Your submission should now be listed in your submission queue.
- Java solutions should be a single source file and should not include the package header. The testing script will also be renaming your file to `Pn.java` (where n is the problem number) and as such the main class for problem 1 should be defined as: `public class P1 {...}` and will be called with: `java P1 < input-file`

- C and C++ submissions will be compiled with the flags `-lm -O2`.
- `C#` submissions will be run using mono since the test environment is running on Linux.
- Haskell submissions will be compiled with the flag `-O2`.

Testing and Scoring

- Solutions should be submitted in the form of source code, which will be compiled on the test environment. A solution will be accepted if it compiles on the test environment and produces the correct output for the sample inputs given in the question.
- The output from your program should be terminated by a new line and should not contain any additional whitespace at the beginning or end of lines.
- A solution will be tested against 10 separate test cases which are designed to test the limits and corner cases of the inputs. One point is given for each correct output.
- Programs are limited to one second of CPU time and 256MB of RAM.
- If a solution is submitted while an earlier solution to the same problem is still in the queue, the earlier submission will not be tested. The earlier submission will be marked with the message: "Old, not going to be tested".
- In the event of a tie, the winning team will be the one whose last scoring submission was submitted earliest.

1 Corcaigh-xit

The day has arrived for the people of the Republic of Cork to declare their independence. Politicians have decided that the border of the county should be protected from the rest of Ireland. Because neighbouring counties stubbornly refuse to pay for the construction of a wall, a decision has been made to plant mines along the county border. To implement this, local geographers have marked rectangular zones of height N and width M in which they defined cells of unit size. Each of these $N \times M$ cells may contain a landmine. They carefully recorded whether or not cells contained a mine and the total number of mines in the zone.

Although the location of the mines must remain secret, local authorities want to create maps from which their location can be inferred. Dónal, a UCC student, was asked to produce these maps so that each cell contains the number of mines buried in the vicinity of that cell or the character 'x' if the cell itself is mined. Each cell has between 3 and 8 neighbours, depending on its location in the grid.

Not Cork

o	x	o	x	o
o	o	o	x	x
o	o	o	o	o

Cork

Not Cork

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

Cork

Original maps on the left. On the right, the expected completed map.

Can you help Dónal to produce these maps ?

Input The first line contains two integers N and M, $1 \leq N, M \leq 1000$, corresponding to the the height and width of the map respecitvely. The next N lines contain M space-separated characters. Each character is either an *x* if this cell contains a mine, or *o* to represent an empty cell.

Output The output consists of N lines of M space-separated characters. Each character either encodes for the number of adjacent cells containing mines, or the character 'x' if the cell itself contains a mine.

Sample Input 1

```
3 5
o x o x o
o o o x x
o o o o o
```

Sample Output 1

```
1 x 3 x 3
1 1 3 x x
0 0 1 2 2
```

Sample Input 2

```
5 5
o o o o o
o o x o o
o o x o o
o o x o o
o o o o o
```

Sample Output 2

```
0 1 1 1 0
0 2 x 2 0
0 3 x 3 0
0 2 x 2 0
0 1 1 1 0
```

2 Make Europe Great Again

In an attempt to provide the best to European citizens, the parliament wishes to reintroduce the use of Roman numerals throughout the Union. A lot of people all over Europe feel that Roman numerals are far superior to the currently used Arabic number system. Unfortunately, most have forgotten the art of counting with Roman numerals. Your task is to develop a Arabic to Roman numeral batch convertor.

Roman Numeral	`I`	`V`	`X`	`L`	`C`	`D`	`M`
Arabic Numeral	1	5	10	50	100	500	1000

Roman numerals and their corresponding Arabic numbers.

The following simple rules must be applied to convert an Arabic number to a Roman numeral.

- Each digit in the Arabic number must be considered as a separate item and as such should be converted to its Roman counterpart. Symbols are placed from left to right in order of their values. For instance, $2017 = 2000 + 10 + 7$ should be converted as `MMXVII` = `MM` + `X` + `VII`.
- In order to avoid the repetition of four characters, a smaller symbol should be placed in front of a bigger one with the meaning that the smaller one is subtracted to the bigger one. There are three cases where this should be applied:
  - `I` is placed in front of `V` or `X`, to create 4 or 9 respectively.
  - `X` is placed in front of `L` or `C`, to create 40 or 90 respectively.
  - `C` is placed in front of `D` or `M`, to create 400 or 900 respectively.

  At most one smaller symbol can be placed in front of a bigger symbol for subtraction. For example, `IIV` would not mean 3.

Input The input consists of two lines. The first contains an integer $1 \leq N \leq 10$ representing the number of Arabic numbers to be converted. The second line contains N integer numbers ranging from 1 to 3999.

Output The output of your program should consist of a single line containing N space-separated string conversions of the Arabic numbers to Roman numerals.

Sample Input 1

```
1
42
```

Sample Output 1

```
XLII
```

Sample Input 2

```
2
2017 99
```

Sample Output 2

```
MMXVII XCIX
```

3 Gauging Goldbach's Gnarly Numbers

Goldbach's conjecture, formulated in 1742, is a long-standing problem in number theory that remains unproven. The conjecture asserts that every even integer greater than 2 can be written as the sum of two prime numbers. For example, the first few even numbers can be written as: $4 = 2 + 2$, $6 = 3 + 3$, $8 = 3 + 5$. Note that the two primes need not be distinct.

In some cases, there are multiple ways to write an even integer as the sum of two primes, for example:

$$10 = 3 + 7 = 5 + 5$$
$$22 = 3 + 19 = 5 + 17 = 11 + 11$$

Note that $3 + 7$ and $7 + 3$ are not considered distinct ways to add two primes up to 10. The first six even numbers that can be written in exactly two ways as the sum of two primes are: 10, 14, 16, 18, 20, and 28.

Since proving or disproving Goldbach's conjecture is a bit much for a Saturday[1], we instead wish to chase the n^{th} even number that can be written in exactly w ways as the sum of two prime numbers.

Input Your program should read a single line containing two positive integers, $1 \leq n \leq 500$ and $1 \leq w \leq 250$.

Output The output should consist of a single line containing the n^{th} even integer that can be written in exactly w ways as the sum of two primes, or 0 if there is no such solution less than or equal to $100,000$.

Sample Input 1

```
5 2
```

Sample Output 1

```
20
```

Sample Input 2

```
1 3
```

Sample Output 2

```
22
```

[1]Although, a bonus point will be offered if you are able to provide a proof.

4 Pete's Partial Password Pilferring

Advanced Computer Management Inc. is an organization with some peculiar security policies. Their email servers only support a very limited selection of passwords, so each user must choose their password from a list of preapproved passwords. There is no requirement for passwords to be unique, so it is possible for multiple users to have the same password. Also, users are dicouraged from changing their passwords regularly, since that is a real hassle.

Unfortunately, Advanced Computer Management Inc. has suffered a security breach. Hackers have stolen the list of approved passwords. Additionally, they have obtained some of the users' login information. In most cases it was only possible for hackers to steal part of a user's password consisting of one or more of the initial characters. From this they can narrow down the possible passwords for each user.

With knowledge of what the hackers have stolen, it is your job to calculate how many possible passwords the hackers will have to try in order to gain access to the compromised user accounts.

Input The first line contains two space separated integers N, M, where $1 \leq N \leq 15000$ and $1 \leq M \leq 15000$. N is the number of forgotten passwords, and M is the size of the list of potential passwords. Each of the following N lines contains a partial password. The next M lines contain the list of approved passwords. Each of these passwords are unique.

Passwords consist of a string of lower case letters (a-z) of length no longer than 50 characters.

Output The output should consist of N lines, where the i^{th} line contains a single integer representing the number of possible passwords that begin with the i^{th} partial password.

Sample Input 1

```
2 6
pa
pro
proof
part
pit
pro
prop
pan
```

Sample Output 1

```
2
3
```

Sample Input 2

```
3 10
hack
sp
le
leet
hack
shack
spies
hacker
hackers
late
leak
hacks
spot
```

Sample Output 2

```
4
2
2
```

5 Reordering Recommended Reading References

As part of your Advanced Joke-Language Programming coursework, you have been asked to read a big pile of technical books; revered tomes of wisdom such as *Effective Ked* and *A Trillion Things a Tremendous TrumpScript Programmer Should Know*.

Each book consists of a series of short sections, which are numbered sequentially from 0. Sections of a book typically contain cross-references to other sections of the same book; for instance, Section 2 may reference Sections 1 and 3, and Section 1 may reference Section 0. However, it is guaranteed that no circular references will occur; for instance, Section 0 cannot both reference and be referenced by Section 1. Note: books are completely independent; they do not reference one another.

For each book, you want to order the sections such that reading the sections in that order would not cause you, at any stage, to read a section referencing other sections that you have not already read. At least one such ordering is guaranteed to exist, but there may be more than one; in such a scenario, ties are broken lexicographically.

Input The first line contains an integer n, such that $1 \leq n \leq 10$, which corresponds to the number of books. Then follow n blocks of lines, each block corresponding to one book.

The first line of each block contains two space-separated integers: s_i and r_i, corresponding to the total number of sections in the book and the total number of cross-references in the book, respectively. Note that $1 \leq s_i \leq 10000$ and $0 \leq r_i \leq s_i(s_i - 1)/2$.

The remainder of the block consists of r_i lines. Each one of those lines defines a single cross-reference within the book, in the form of two space-separated integers: the index of the section that contains the reference, followed by the index of the section that is being referenced.

Output The output should consist of n lines; the ith line should contain s_i space-separated integers corresponding to the indices of the ith book's sections in the desired order.

Sample Input 1

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

Sample Output 1

```
0 1 2
```

Sample Input 2

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

Sample Output 2

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

6 Repairing Ropey Rhetoric

A pathetic politician's pitiful pronouncements are poisoning his popularity. Glaring grammatical gaffs, simple spelling slip-ups, and multiple major mistakes polluting his presidential proclamations.

You have been hired to correct these numerous errors in his speech texts to turn his troubled tirade into a tremendous talk. Luckily for you, you charge for each correction, and there are plenty of corrections to make. Every character added, removed, or changed increases your fee. However, as an honest person, you intend to make the minimal number of changes.

Your task is to compare a number of corrected texts with their originals, and count how many changes you had to make for each of them.

A change can be: i) inserting a character, ii) removing a character, or iii) replacing a character with another character.

Input The first line of input contains a single integer n, where $1 \leq n \leq 10$, representing the number of texts that have been changed.

For each of these texts, the input will contain three lines. The first contains two space-separated integers x and y, such that $1 \leq x \leq 500$ and $1 \leq y \leq 500$. x is the number of characters in the original text, and y is the number of characters in the corrected text, including spaces, numbers, and punctuation. The second line contains x characters representing the original text, and the third line contains y characters corresponding to the corrected text.

The text may contain: i) uppercase letters (A-Z), ii) lowercase letters (a-z), iii) digits (0-9), iv) punctuation (full stop, commas, hyphens, or spaces).

Output Output must consist of n lines, where the i^{th} line contains the number of changes needed to transform the i^{th} original text into the i^{th} corrected text.

Sample Input 1

```
1
7 6
sitting
kitten
```

Sample Output 1

```
3
```

Sample Input 2

```
2
14 16
sunday, monday
saturday, friday
18 19
tuesday, wednesday
thursday wednesday.
```

Sample Output 2

```
6
4
```