Program Design (I)

Lec02

Division of Two Integers

Description

Given two integers x and y, your program should calculate the answer of x**/**y and print out the result according to the **Output Format**.

Note : -10^3 < x, y < 10^3

Input

Two integers split by a space.

Output

Two values split by space: 1. One digit after the decimal point. 2. Two digits after the decimal point. All the values should be rounded.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5 9
```

Output

```
0.6 0.56
```

Sample2

Input

```
0 999
```

Output

```
0.0 0.00
```

Make a Circle

Description

Suppose you are in an art class now. The teacher wants you to make a circle, made up of cloth, with string glued on the edge. Given the radius of the circle, you should calculate the area of the cloth and the length of the string that you need in order to make this circle.

You should use macro to define PI as 3.14f.

Input

The radius of the circle. It is an integer x where 1 <= x <= 100.

Output

Two floating-point numbers (round to nearest 100th) separated by a space. The first one represents the area of the cloth, and the second one represents the length of the string.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1
```

Output

```
3.14 6.28
```

Sample2

Input

```
3
```

Output

```
28.26 18.84
```

Say Hello to C

Description

Welcome to program design (I). Let's say hello to our best friend C.

Input

No input

Output

Output a single line "Hello C!"

Loader Code

Your code will be judge using this program:

Sample1

Input

Output

```
Hello C!
```

Swap Two Variables

Description

Read two integers and store them in variables a and b.
Swap the values in a and b and print them out .

Input

Two integers split by a space.

Output

Two integers split by a space.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
20 21
```

Output

```
21 20
```

Sample2

Input

```
2147483647 -2147483648
```

Output

```
-2147483648 2147483647
```

Lec03

Addition in Column Form

Description

Your cousin is a second grade elementary school student. He learned the addition that written in column form in his today's class. However, he still has lots of questions about it. Your task is to write a small program to demonstrate the answer to the addition of $x$and $y$written in column form.

Input

Given 2 integers split by a space. 0 <= x, y <= 9999

Output

The column form will be performed in 4 lines. First line will start in 2 space and augend(被加數) padding with zeros until 4 digits. Second line will start in "+)" and addend (加數) padding with zeros until 4 digits. Third line is "------". Last line is the result padding with zeros until 6 digits.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
0 0
```

Output

```
  0000
+)0000
------
000000
```

Sample2

Input

```
9999 9999
```

Output

```
  9999
+)9999
------
019998
```

Sample3

Input

```
105 9651
```

Output

```
  0105
+)9651
------
009756
```

Sample4

Input

```
50 985
```

Output

```
  0050
+)0985
------
001035
```

Cirno's Perfect Math Class 琪露諾的完美算術教室

Description

琪露諾的算術教室要開始了～以天才為目標努力吧！

琪露諾除了算術教室以外也經營幻想鄉客運（車上附冷氣）

某天她想計算經過每站後車上分別會有幾位乘客

但眾所皆知，琪露諾是個『⑨. 笨蛋』

所以請各位幫忙計算人數

問題：

巴士從紅魔館 (`X`站) 出發　一開始有 `A` 人上車

在白玉樓 (`Y` 站) 有 `B` 人下車　另外有 `C` 個人上車了

在八雲家 (`Z` 站) 有 `D` 人下車　請問每次過站後車上分別有幾位乘客?

為了讓琪露諾看得懂，請在每個人數前用`0`補滿四位數

Everybody! Cirno's math class is about to begin! Do your best to become a genius!

Not only the math class, Cirno also operates a bus company "Gensokyo Bus", One day she wants to know how many passengers is onboard when the bus passes each station,

but everybody knows cirno is a "⑨. Baka" (a.k.a. an idiot),

so please help her to calculate this.

Problem:

A bus left the Scarlet Devil Mansion (say, `X` station); `A` people boarded at the start

At Hakugyokurou (say, `Y` station), `B` left and `C` people boarded

At Yakumo-san's house (say, `Z` station), `D` people left; so how many passengers is on board after each station?

To help cirno understand the number, please left-padding each number to 4 digits with `0`

https://www.youtube.com/watch?v=xFdDNrd6W9s

Input

One line contain 4 integers A B C D separated by space (0 <= A, B, C, D <= 9999)

Output

One line contains 3 integers x, y, z padding with zeros until 4 digits separated by space (0 <= x, y, z <= 9999)

Loader Code

Your code will be judge using this program:

Sample1

Input

```
6381 5852 3862 4329
```

Output

```
6381 4391 0062
```

Sample2

Input

```
8982 8444 1532 571
```

Output

```
8982 2070 1499
```

Sample3

Input

```
3791 2054 6299 7230
```

Output

```
3791 8036 0806
```

Sample4

Input

```
6641 787 1731 5883
```

Output

```
6641 7585 1702
```

Scanf Format Problem

Description

There is a sequence A/B**C/)D , where A,B,C,D are integer.

Please output the $A+B+C+D$.

Input

There is one line sequence A/B**C/)D. (1 < A,B,C,D < 1000000 and A,B,C,D are integers)

Output

Please output the A+B+C+D. No need to print the newline.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
123/456**789/)222
```

Output

```
1590
```

Sample2

Input

```
234322/45226**723489/)22222
```

Output

```
1025259
```

Sample3

Input

```
2/43226**712119/)522
```

Output

```
755869
```

Sample4

Input

```
684643/930498**150845/)464300
```

Output

```
2230286
```

俗俗的賣大拍賣 Discount Price Tag Printing

Description

2021/10/21 21:50 更新 / Update:

新增提示，請至下方 Hint 查看

Added hint below

資訊系阿超正在協助可憐的銷售員設計即將到來的周年慶看板。

但是因為人工處理起來真的太費時了，

請幫她設計一個程式可以自動化的處理完這件任務吧!

別擔心，阿超知道要如何將輸出結果從螢幕終端顯示器中帶到現實的!

A-chao, from CSIE, is helping a poor supermarket salesperson designing the price tag for the upcoming anniversary event. However, there is way too much to do it by hand. Please help him design a program to print out the price tag format. Don't worry, he knows how to make things in the terminal become physical.

The following are the rules for each output lines:

- Line #0 / #4: Filly Packed with the character `"`
- Line #1: Start with a backslash `\`, four spaces, **the discount percentage which takes 3 character lengths**, the string `%off!`, four more spaces, then ends with another backslash.
- Line #2: Start with a backslash `\`, one space, the string `Before` , two more spaces, a dollar sign `$`, **the original cost without a decimal point** using a **field width of five characters (justified to the right)**, and ends with another space and a backslash.
- Line #3: Start with a backslash `\`, one space, the string `After`, three more spaces, a dollar sign `$`, **the discounted cost till the second decimal place** using a **field width of five characters (justified to the right)**, and ends with another space and a backslash.
- **No leading zero** should be added to any of the printed numbers.
- You can checkout the testing cases for full examples.

Input

Two integers a, b separated by a space. a represents "discount percentage". b represents "original cost". 0 <= a <= 100 1 <= b <= 99

Output

Consists of 5 lines, forming a 18*5 rectangle. Please refer to the description section for details of each line.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
50 8
```

Output

```
""""""""""""""""""
\     50%off!    \
\ Before  $    8 \
\ After   $ 4.00 \
""""""""""""""""""
```

Sample2

Input

```
25 16
```

Output

```
""""""""""""""""""
\     25%off!    \
\ Before  $   16 \
\ After   $12.00 \
""""""""""""""""""
```

Sample3

Input

```
15 80
```

Output

```
""""""""""""""""""
\     15%off!    \
\ Before  $   80 \
\ After   $68.00 \
""""""""""""""""""
```

Sample4

Input

```
100 1
```

Output

```
""""""""""""""""""
\    100%off!    \
\ Before  $    1 \
\ After   $ 0.00 \
""""""""""""""""""
```

Lec04

EAN-13

Description

In Taiwan, manufacturers of goods put a barcode on each product. This code identifies both the manufacturer and the product. Among a variety of different formats of the barcodes, one of these is known as EAN-13, which represents a thirteen-digit number. For example: 4 710367 663270.

The EAN-13 is composed of the following four components:

1. GS1 prefix: The first three digits usually identify the GS1 Member Organization which the manufacturer has joined.
2. Manufacturer code: This is a unique code (variable-length) assigned to each manufacturer by the numbering authority indicated by the GS1 Prefix.

3. Product code: The product code, which immediately follows manufacturer code, is assigned by the manufacturer.
4. Check digit: This is an additional digit, used to verify that a barcode has been scanned correctly.

To calculate the check digit, follow the three steps below:

1. Numbering the positions from the right (exclude check digit), the odd positions in the twelve-digit number are weight of 3 and the even positions are weight of 1. Multiply each data digit with its corresponding weight.

   For example:

   > data: 4 7 1 0 3 6 7 6 6 3 2 7
   >
   > weight: 1 3 1 3 1 3 1 3 1 3 1 3

2. Calculate the sum of the results from step 1.
3. Subtract the sum from **the nearest multiple of 10 that is equal to or higher than the sum (see hint)**. The result is the check digit.

Write a program that calculate the check digit.

Reference: https://en.wikipedia.org/wiki/International_Article_Number

Input

Three integers (1 digit, 6 digits and 5 digits) which form the first twelve-digit number of the EAN-13.

Output

Output the check digit that your program calculate.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
4 710367 66327
```

Output

```
0
```

Sample2

Input

```
4 006381 33393
```

Output

```
1
```

Lena's pudding plan

Description

Lena loves to eat pudding, one day, she receives several boxes of puddings from her friend Anetta. The puddings are handmade and can only leave for a week. Lena does not want any pudding to become spoiled, and she wants to have the same number of puddings each day. Please help her to deal with a plan to take these puddings in the coming week.

There are three boxes of puddings, you have to find out how many puddings that Lena can take in each day of a week (**7 days**), moreover, how many puddings will be left if she wishes to have the same number of puddings each day.

Input

Three integers in a single line that indicates the number of puddings in each box. There are not more than 10000 puddings in each box.

Output

Two integers x and y in a single line, where x indicates the number of puddings that Lena can take for each day, and y means the number of puddings left.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
9112 9213 1709
```

Output

```
2862 0
```

Sample2

Input

```
4972 1970 979
```

Output

```
1131 4
```

Sample3

Input

```
1287 5609 393
```

Output

```
1041 2
```

Sample4

Input

```
2551 229 8342
```

Output

```
1588 6
```

Polynomial Calculation

Description

Read a floating-point number for `x` and calculate the value of the following polynomial:

> $\mathbf{7x^4 - 8x^3 - x^2 + 6x - 22}$

Input

A floating-point number to read for x.

Output

The result should be displayed with one digit after the decimal point.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5.5
```

Output

```
5055.2
```

Sample2

Input

```
0
```

Output

```
-22.0
```

Temperature conversion

Description

Give you a floating-point number that represent Celsius. Please convert it to Fahrenheit. Fahrenheit = Celsius *9/5 + 32

Input

A floating-point number that represent Celsius. (-273.15 <= t <= 10^4)

Output

A floating-point number with two digits after the decimal point that represents Fahrenhelt.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
4794.71
```

Output

```
8662.48
```

Sample2

Input

```
5970.84
```

Output

```
10779.51
```

Lec05

Arabic to Roman

Description

Roman numerals are a numeral system that originated in ancient Rome and are represented by seven different symbols:

> Symbol: I, V, X, L, C, D, M
>
> Value; 1, 5, 10, 50, 100, 500, 1000

For example, 6 is written as VI in Roman numerals, just 5 and 1 added together.

However, the numerals for 4 (IV) and 9 (IX) are written using "*subtractive notation*", where the first symbol (I) is subtracted from the larger one (V, or X), thus avoiding the clumsier (IIII, and VIIII). Subtractive notation is also used for 40 (XL) and 90 (XC), as well as 400 (CD) and 900 (CM). These are the only subtractive forms in standard use.

Given an integer, convert it to a Roman numeral.

Input

An integer which is guaranteed to be within the range from 1 to 3999.

Output

A Roman numeral representing the input integer.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1764
```

Output

```
MDCCLXIV
```

Sample2

Input

```
2439
```

Output

```
MMCDXXXIX
```

Pokémon GO

Description

Ariel likes to play Pokémon GO! Everyday, she will earn `n` poké balls and want to catch one pokémon. According to the Pokémon's combat ability `cp`, health points `hp`, and speed `s` when the pokémon runs away, Ariel needs to use some poké balls to catch the pokémon. Please tell her whether she can catch the pokémon successfully. If yes, how many poké balls will remain?

1. If the pokémon's combat ability is less than 100, Ariel only needs to use one poké ball.
2. If the pokémon's combat ability is between 100 and 600 (including 100 and 600), she needs to use 5 poké balls. Furthermore, she has to use another 3 poké balls if the pokémon's health points are more than 100.
3. If the pokémon's combat ability is more than 600, she needs to use 15 poké balls. Furthermore, she has to use another 10 poké balls if the pokémon's health points are more than 200.
4. If the pokémon's speed is more than 4, the pokémon runs away after Ariel has used 6 poké balls. It means that she can't catch the pokémon anymore!

Ariel 喜歡上玩 Pokémon GO！每天都會得到 `n` 個寶貝球，也都會捕捉一隻寶可夢，捕捉寶可夢所需要的寶貝球數目會根據寶可夢的戰鬥能力 `cp`、血量`hp` 以及逃跑速度 `s` 而有所不同。

1. 如果戰鬥能力小於 100，Ariel 只需要使用一個寶貝球就可以抓到寶可夢

2. 如果戰鬥能力介於 100 到 600 之間 (包括 100 與 600)，則需消耗 5 個寶貝球，而且若血量超過 100 ，要額外使用 3 個寶貝球
3. 如果戰鬥能力超過 600，則需要消耗 15 個寶貝球，而且若血量超過 200，要額外使用 10 個寶貝球
4. 如果逃跑速度大於 4，寶可夢會在 Ariel 使用了 6 個寶貝球之後逃跑(即第六個寶貝球還是沒有抓到寶可夢)，逃跑的寶可夢是想抓也抓不到了!\

請告訴 Ariel 這隻寶可夢能不能抓的到，如果可以抓的到，那 Ariel 會剩下幾個寶貝球?\

Input

An integer n represents the number of Poké Balls.(1<=n<=50) Two integer cp, hp represents the Pokémon's combat power and health points, respectively.(1<=cp<=1000, 1<=hp<=1000) An integer s represents the Pokémon's speed.(1<=s<=10)

Output

Output "NO" if Ariel can't catch Pokémon. Otherwise, Output "YES" and remaining Poké Balls.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
11
344 101
5
```

Output

```
NO
```

Sample2

Input

```
40
600 100
2
```

Output

```
YES
35
```

Split the number

Description

Read an integer x, and output its digits split by space.

Note: 0 <= x <= 999

Input

Single integer

Output

Three digits split by spaces. If the number of digits is less than three, pad the output with zeros to make it three digits.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
100
```

Output

```
1 0 0
```

Sample2

Input

```
5
```

Output

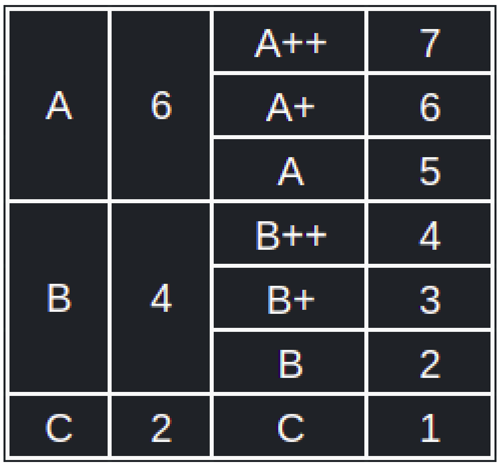
```
0 0 5
```

國中教育會考成績 CAP grade

Description

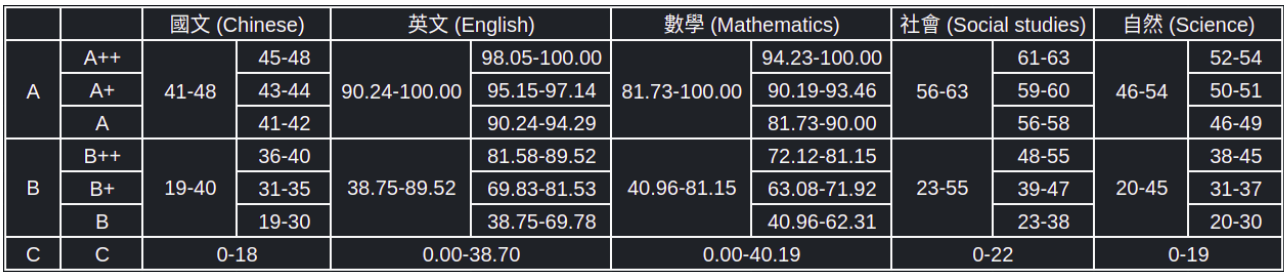
The **Comprehensive Assessment Program for Junior High School Students(CAP)**, is one of the current Senior high-school entrance program in Taiwan. The exam regulations and grading rules are as following:

- The CAP consists of 5 subjects: **Chinese, English, Mathematics, Social Studies, and Science**.
- Base on the result of each subject, each are divided into **"Excellent" (Grade A), "Fair" (Grade B), and "Improvement Needed" (Grade C)** three ranks. Within grade A and B, each will be further divided into three smaller levels **"A++/B++", "A+/B+", and "A/B"**.
- Each rank corresponds to different amount of **score**, and each level corresponds to a different amount of **point**. Both are summed up to get the total score / total points. The score / point translation table is as following:



A	6	A++	7
		A+	6
		A	5
B	4	B++	4
		B+	3
		B	2
C	2	C	1

- The subjects Chinese, Social Study and Science are graded by the number of correct answers. The subjects Mathematics and English, are graded by the weighting system. The translation table is as following:



		國文 (Chinese)		英文 (English)		數學 (Mathematics)		社會 (Social studies)		自然 (Science)	
A	A++	41-48	45-48	90.24-100.00	98.05-100.00	81.73-100.00	94.23-100.00	56-63	61-63	46-54	52-54
	A+		43-44		95.15-97.14		90.19-93.46		59-60		50-51
	A		41-42		90.24-94.29		81.73-90.00		56-58		46-49
B	B++	19-40	36-40	38.75-89.52	81.58-89.52	40.96-81.15	72.12-81.15	23-55	48-55	20-45	38-45
	B+		31-35		69.83-81.53		63.08-71.92		39-47		31-37
	B		19-30		38.75-69.78		40.96-62.31		23-38		20-30
C	C	0-18		0.00-38.70		0.00-40.19		0-22		0-19	

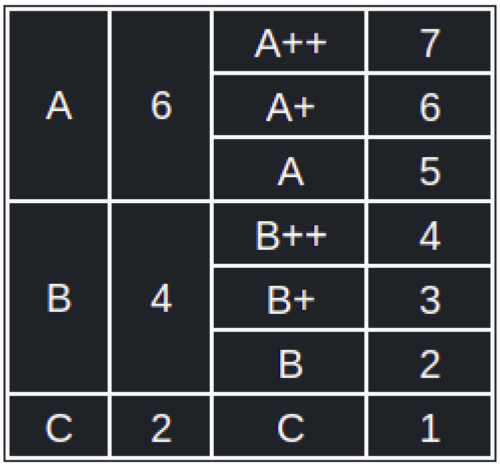
- The final result can be represented as "**score/points(ranks)**".

Example: Steven's grade of each subject is: **CH=43, EN=92.5, MA=79.8, SO=61, SC=41**. We can know the rank of each subjects is: **CH=A+, EN=A, MA=B++, SO=A++, SC=B++**. So the total score is 6 + 6 + 4 + 6 + 4 = 26, and the total points 6 + 5 + 4 + 7 + 4 = 26. The final CAP grading result is `26/26(3A2B)`.

Given a student's grade of each subject, please calculate the final CAP grading result.

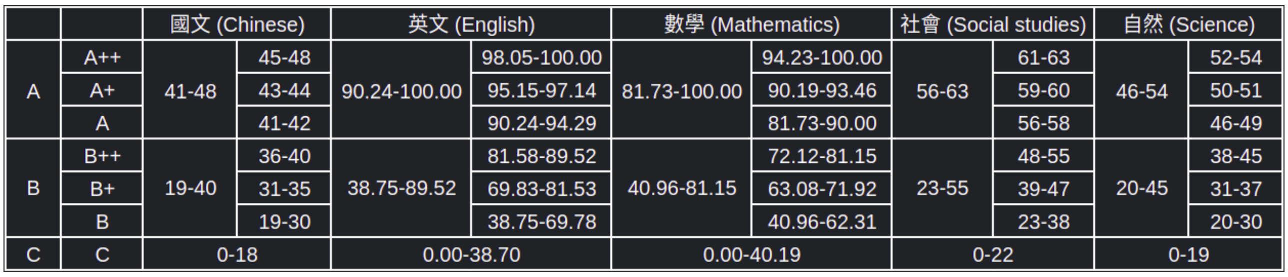
國中教育會考，是台灣現行的高中入學管道之一。會考的考試規則與成績計算方式如下：

- 考科總共有**國文**、**英文**、**數學**、**自然**、**社會**五科。
- 每一考科的成績依高低被區分為**精熟 (A)**、**基礎 (B)**、**待加強 (C)**三個等級，而其中精熟與基礎又可再分別細分為**「A++ / B++」、「A+ / B+」、「A / B」**三個分級。
- 不同的等級會對應到不同**積分**，不同分級對應到不同**積點**。將各科等級加總會得到總積分，各科分級加總得到總積點。其換算表如下表所示：



A	6	A++	7
		A+	6
		A	5
B	4	B++	4
		B+	3
		B	2
C	2	C	1

- 國文、社會、自然三科以答對題數進行評分；數學、英文因為分別有手寫、聽力部份，另以加權分數進行評分。108 學年度的各科量尺分數換算表如下表所示：



		國文 (Chinese)		英文 (English)		數學 (Mathematics)		社會 (Social studies)		自然 (Science)	
A	A++	41-48	45-48	90.24-100.00	98.05-100.00	81.73-100.00	94.23-100.00	56-63	61-63	46-54	52-54
	A+		43-44		95.15-97.14		90.19-93.46		59-60		50-51
	A		41-42		90.24-94.29		81.73-90.00		56-58		46-49
B	B++	19-40	36-40	38.75-89.52	81.58-89.52	40.96-81.15	72.12-81.15	23-55	48-55	20-45	38-45
	B+		31-35		69.83-81.53		63.08-71.92		39-47		31-37
	B		19-30		38.75-69.78		40.96-62.31		23-38		20-30
C	C	0-18		0.00-38.70		0.00-40.19		0-22		0-19	

- 學生成績可以被表示為「**積分/積點（等級）**」。

例：小明的五科分數分別為：**國 43**、**英 92.5**、**數 79.8**、**社 61**、**自 41**則可以得知其各科分級為：**國 A+**、**英 A**、**數 B++**、**社 A++**、**自 B++**，總積分為 6 + 6 + 4 + 6 + 4 = 26，總積點為 6 + 5 + 4 + 7 + 4 = 26，其會考成績可表示為`26/26(3A2B)`。

給定某生的各科分數，請你幫忙計算該生的會考成績。

Input

Five number CH, EN, MA, SO, SC, which represent the original grade of Chinese, English, Mathematics, Social studies and Science, respectively. The data type of CH, SO, SC is integer. The data type of EN, MA is float.

Output

The final grade of this student, in the format `score/points(ranks)`. Please refer to description above.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
43 92.5 79.8 61 41
```

Output

```
26/26(3A2B)
```

Sample2

Input

```
47 98.5 98.2 63 54
```

Output

```
30/35(5A)
```

Sample3

Input

```
38 35.2 28.1 45 48
```

Output

```
18/14(1A2B2C)
```

Lec06

Collatz conjecture (3n + 1 Problem)

Description

Collatz conjecture, also known as **3n+1 problem**, which is a conjecture in mathematics that is about a sequence defined as follows: Let `S1` be a positive integer which is the start of the sequence, and

- if `Sn` is **even**, then `Sn+1 = Sn / 2`
- if `Sn` is **odd**, then `Sn+1 = 3 * Sn + 1`

The conjecture is that for any given positive integer `S1` (<= 10^5), the sequence will reach 1.

e.g. Given `S1 = 22`, the sequence will be: `22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1`

As students of NCKU, we are interested in this conjecture, please write a program to help us do some research in this conjecture.

Input

A positive integer S1, (1 <= S1 <= 10^5)

Output

Two integer L, M within a line, where L represents the length of the sequence and M is the maximum number of the sequence.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
22
```

Output

```
16 52
```

Sample2

Input

```
999
```

Output

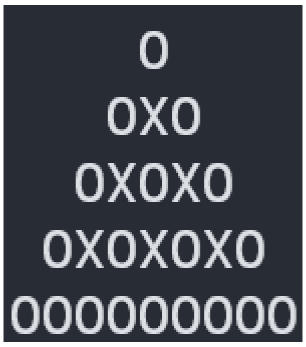
```
50 11392
```

Congruent OOXX Triangle

Description

Given a number h, print a **congruent triangle** with height h that consists of OOXX (the last line must be only consist of O).

For example, h=5:



```
    O
   OXO
  OXOXO
 OXOXOXO
OOOOOOOOO
```

Input

An integer h. (1 <= h <= 150)

Output

A congruent triangle consisted of OX, and the bottom of the triangle must only contains O.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1
```

Output

```
O
```

Sample2

Input

```
5
```

Output

```
    0
   0X0
  0X0X0
 0X0X0X0
000000000
```

Pokémon Go II - Pokémon Gym

Description

Ariel likes to play Pokémon Go! She had many battles in pokémon gym. She needs a tool to record the battle's result.
She had n battles. Each battle is three-on-three and each pokémon has **combat power(CP)**. If the combat power of Ariel's pokémon is more than competitor's, then Ariel's pokémon wins. If Ariel has more wins than competitor, she wins the battle. Please record each battle is "**Win**", "**Tie**", or "**Lose**" and calculate the **win rates**.

For example,

Ariel had a battle. Her three pokémons' combat power are 200, 300 and 400, and the competitor's three pokémons' combat power are 100,350 and 400. Then,

200>100 Win

300<350 Lose

400=400 Tie

Because Ariel and competitor had the same number of wins, this battle is "**Tie**". Ariel had one battle and the number of wins is 0, then the win rates is **0/1 = 0.00%**

Ariel 最喜歡玩 Pokémon Go! 她參加了很多場道館賽，所以她需要有個小幫手幫忙記錄!
她參加了 n 場道館對戰，每一場中，對方派出 3 隻寶可夢，Ariel 也同樣派出 3 隻寶可夢應戰，每隻寶可夢都有 **CP 值**，CP 值較高的寶可夢獲勝，CP 值相同的寶可夢對戰則為平手，獲勝的寶可夢數量最多的那一方獲勝，若雙方獲勝的寶可夢數量相同時，則這一場為平手，請幫忙紀錄每一場比賽 Ariel 是 **獲勝**、**平手**或是**輸**，並算出**勝場率**。

舉例:

Ariel 參加了一場道館賽，派出了三隻寶可夢，其CP值分別為 200、300、400，而對方派出的寶可夢其 CP 值為 100、350、400
所以

200>100 Win

300<350 Lose

400=400 Tie

雙方各有一隻寶可夢獲勝，所以這場比賽比賽為平手(**Tie**)，而Ariel總共參加了一場比賽，勝場數為 0 ，所以**勝場率**為 **0/1 = 0.00%**

Input

First line, an integer n represents the number of battles. Then, there are 3*n lines after the first line. (1<=n<=50) Each line has two integer a, b represent combat power(CP) of Ariel's and competitor's pokémon, respectively.(0<=a,b<=1000)

Output

You have output n battle's results and the win rates. If Ariel wins the battle, output "Win". If Ariel loses the battle, output "Lose". if the battle is tie, output "Tie". The win rates is a floating-point number with two digits after the decimal point.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
3
1000 200
500 100
400 399
100 200
222 122
200 200
133 500
122 444
700 900
```

Output

```
Win
Tie
Lose
33.33%
```

Sample2

Input

```
1
200 500
100 222
500 900
```

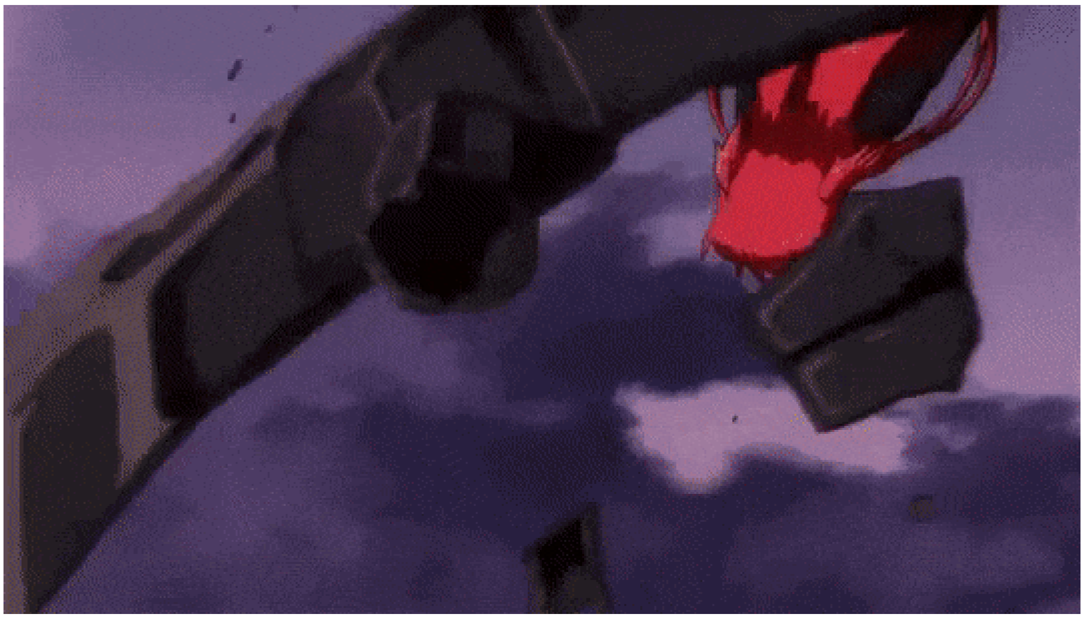
Output

```
Lose
0.00%
```

The happiest girl in the world

Description

On the way back to the Skyland in an airship, crews face a huge attack from monsters. Willem falls out of the airship after several waves of attack, Chtholly notices that and wants to save him, please help her calculate how many meters remained before Willem reaches the ground. You may use the following equation to calculate the distance Willem has traveled through: $x = \frac{1}{2}gt^2, g = 9.8m/s^2$



Input

An integer h that indicates the height of the airship, 1000 <= h <= 10000

Output

The remained distance, in floating-point number, for each second until Willem reaches the ground, each occupies a line and is rounded to the first digit after the decimal point.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1000
```

Output

```
1000.0
995.1
980.4
955.9
921.6
877.5
823.6
759.9
686.4
603.1
```

```
510.0
407.1
294.4
171.9
39.6
```

Lec07

Add Up Two Hexadecimal

Description

There are two **unsigned 16 bits** hexadecimal integer `A B`,

Please add up those number and store the answer as an **unsigned 16 bits decimal integer**.

If the answer is overflow, print `OVERFLOW!`, otherwise, print the the result of `A + B`, the type of which is unsigned 16 bits decimal integer.

Input

Two unsigned 16 bits hexadecimal integer.

Output

If the result of adding up two input is overflow, print "OVERFLOW!", otherwise, print the result of A + B (unsigned 16 bits decimal integer).

Loader Code

Your code will be judge using this program:

Sample1

Input

```
0x0001 0x0001
```

Output

```
2
```

Sample2

Input

```
0xFFFF 0x0001
```

Output

```
OVERFLOW!
```

Caesar Cipher Encoder

Description

The Caesar cipher (a cipher is a method of encrypting data) is one of the simplest and most widely known encryption techniques. For English text, the Caesar cipher would work by taking each letter in the plaintext message and substituting the letter that is k letters latter (allowing wraparound; that is, having the letter **z** followed by the letter **a**) in the alphabet.

For example, if $k = 3$, then the letter **a** in plaintext becomes **d** in ciphertext, **B** in plaintext becomes **E** in ciphertext, and the plaintext message **"bob, i love you. Alice"** becomes **"ere, l oryh brx. Dolfh"** in ciphertext. Note that k may be negative. It means that you should substitute the letter in the plaintext message that is $|k|$ letters earlier.

Write a program that prints out the Caesar cipher for the given input (described in Input Format).

Note: Transform only English alphabet (case-preserving) and preserve other characters such as Arabic numerals or punctuation.

Input

The input contains two lines. Both lines are followed by a newline character. The first line contains an integer for k (-10000 <= k <= 10000), and the second line is a plaintext message.

Output

Output the ciphertext of the plaintext message in the input using the Caesar cipher. There is no newline character at the end of the output.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
7
TAs of the PD1 course are so handsome !!!
```

Output

```
AHz vm aol WK1 jvbyzl hyl zv ohukzvtl !!!
```

Sample2

Input

```
-105
a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z.
```

Output

```
z,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y.
```

Combination

Description

The combination is defined as `C(m, n) = m! / (n! * (m - n)!)`, and `n!` is n's factorial. Use the definition above to calculate the value of `C(m, n)`.

Input

Two integers m, n separated by a space. 1 <= n <= m < 30.

Output

An integer represents the value of C(m, n).

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5 2
```

Output

```
10
```

Sample2

Input

```
10 7
```

Output

```
120
```

進階版快閃心算 - Advanced Flashing Number Mental Arithmetic

Description

快閃心算（フラッシュ暗算）是源自日本珠算界的一個數學競技活動。計算者會在螢幕上看到快速出現消失的數個數字，在顯示完成之後必須在約5秒內回答出所有出現數字的總和。覺得聽起來很簡單嗎？參考 https://www.youtube.com/watch?v=Q7Jd3Mbzup0 試試吧！

資訊系的阿超覺得『沒錯，就是太簡單了』。於是他設計了進階版的規則：

1. 螢幕上顯示的數字可能是 **6 進位(SEN)、8 進位(OCT)、10 進位(DEC)，以及16 進位(HEX)。**
2. 每次**出現的數字 K 介於 0 和 999,999,999 之間。**
3. 由於最後的數字可能很大，**回答時必須用 1,000,000,007 取餘數。**
    - 舉例來說，假設總和是 1,000,000,010，由於 3 ≡ 1,000,000,010 mod 1,000,000,007**，必須回答『3』。**

同時，阿超也出了一些題目，但是太簡單了他不屑自己算。請幫他設計一個程式，可以幫他計算這些題目的答案！

※題目中沒有加註進位制的數字皆為十進制！※

Flashing Number Mental Arithmetic（フラッシュ暗算）is a competitive game originated from Japanese zhusuan circle. The calculator (a human, that is) will see several numbers rapidly appear and disappear on the screen, after it, he/she have to answer the sum of all appeared numbers in a few seconds. Sounds simple? Try it out at https://www.youtube.com/watch?v=Q7Jd3Mbzup0！

Chao from CSIE thinks that, "Yup, its way too easy". So, he designed the rules for a "advanced" version:

1. The numbers appear on the monitor might be in **Base-6(SEN), Base-8(OCT), Base-10(DEC), or Base-16(HEX)**.
2. Value of every number will be in **between 0 and 999,999,999**.
3. Since the final number might be extremely large, **you have to answer in modular 1,000,000,007.**
   - E.g., if the sum is 1,000,000,010, since that 3 ≡ 1,000,000,010 mod 1,000,000,007, **you have to answer "3"**.

At the same time, Chao also created some problems, but they are waaaaay too easy for him to solve. Help him design a program that calculates the correct answer.

※ All number in this problem description, unless specified, are in Base-10(Decimal) ! ※

Input

First line consist of a integer N, represents how many number to sum. 1 <= N <= 1,000 Followed by N lines, each consists of 2 integers, S and K. S denotes the base of K. S ∈ {6, 8, 10, 16} , 0 <= K <= DEC999,999,999 If K is Base-6, or Base-10, its written normally. If K is Base-8, it will have prefix "0". (e.g. 0123, 0777) If K is Base 16, it will have prefix "0x". (e.g. 0x41414141, 0xDEADBEAF)

Output

output a integer, that is the sum of all input K, mod 1,000,000,007.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
4
6 20
8 040
10 100
16 0xFF
```

Output

```
399
```

Sample2

Input

```
8
6 0
8 00
10 0
16 0x0
6 243121245343
8 07346544777
10 999999999
16 0X3B9AC9FF
```

Output

```
999999975
```

Lec08

Beer Can Wall

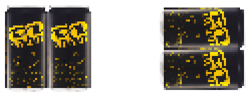
Description

Aqua loves beer. She wants to stack a wall using beer cans. The wall will be only 20cm tall. Assume that the size of each beer can is 20cm by 10cm, and she can put it horizontally or vertically. Given the width of the wall `W`(always multiples of 10cm), please help her calculate the combinations of the placement.

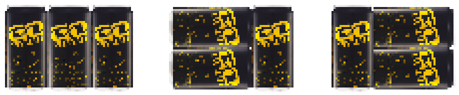
For example:

1. There is 1 combination for 10cm width wall.

2. There are 2 combinations for 20cm width wall.



3. There are 3 combinations for 30cm width wall.



Input

An integer W, where 0 <= W <= 500 and w % 10 == 0.

Output

The number of combinations.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
20
```

Output

```
2
```

Sample2

Input

```
30
```

Output

```
3
```

Sample3

Input

```
40
```

Output

```
5
```

Pokémon Go III - Team Rocket

Description

火箭隊想要搶走小智的皮卡丘! 他們總共有 26 種搶走皮卡丘的計畫，為 A ~ Z 計畫，他們把計畫放進不同編號的箱子中，但卻忘了做紀錄，現在火箭隊完全不知道計畫被放進那些箱子中，只好翻箱到櫃，他們總共找了 n 個箱子，每個箱子都擁有一個編號 x, 箱子內會有一個計畫 y, 請列出火箭隊找到的計畫被放進哪些箱子中，以及沒有找到的計畫有哪些。

ps. 計畫編號有可能不是 A ~ Z

Input

Integer n represents the number of boxes. Then, there are n lines. In each line, an integer x(1<=x<=10^9) and a character y(could be any ASCII character) represent the box's number and the plan's name respectively.

Output

From plan A to Z, output the plan is found or not. For example, output "Plan A is in box 12." if the plan A is found in box 12. Otherwise, output "Not found plan B." if the plan B is not found.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5
12 B
20 D
50 C
9 G
1000 E
```

Output

```
Not found plan A.
Plan B is in box 12.
Plan C is in box 50.
Plan D is in box 20.
Plan E is in box 1000.
Not found plan F.
Plan G is in box 9.
Not found plan H.
Not found plan I.
Not found plan J.
Not found plan K.
Not found plan L.
Not found plan M.
Not found plan N.
Not found plan O.
Not found plan P.
Not found plan Q.
Not found plan R.
Not found plan S.
Not found plan T.
Not found plan U.
Not found plan V.
Not found plan W.
Not found plan X.
Not found plan Y.
Not found plan Z.
```

Sample2

Input

```
8
99 A
12344 W
221 p
5434 q
8888 ?
9898 $
12233 Y
933 0
```

Output

```
Plan A is in box 99.
Not found plan B.
Not found plan C.
Not found plan D.
Not found plan E.
Not found plan F.
Not found plan G.
Not found plan H.
Not found plan I.
Not found plan J.
Not found plan K.
Not found plan L.
Not found plan M.
Not found plan N.
Not found plan O.
Not found plan P.
Not found plan Q.
Not found plan R.
Not found plan S.
Not found plan T.
Not found plan U.
Not found plan V.
Plan W is in box 12344.
Not found plan X.
Plan Y is in box 12233.
Not found plan Z.
```

sister's noise

Description

"sister`s noise　捜し続ける"

Misaka has many sisters (totally 20000 of them), they take turns going abroad, others who stay live in 10 houses on the same street.

They always chat with each other, but neighbors often complain about their noise

You can calculate the noise level made by `M` sisters in house `A` and `N` sisters in house `B` with this formula:

`Noise(A, B) = (M + N) x |A - B|`

Now you know how many of them in each house, please calculate max noise level they can made.

御坂有很多妹妹們（總共有2萬個），她們常輪流出國，剩下的則住在同一條街上相鄰的10戶

她們很喜歡互相聊天交換情報，但人多嘴雜常常被鄰居抱怨太吵

住在第`A`戶的`M`個妹妹與第`B`戶的`N`個妹妹們互相聊天製造的噪音音量計算方式如下：

`Noise(A, B) = (M + N) x |A - B|`

現在你知道這10戶分別住了多少人

請你找出這些妹妹們會製造出的最大噪音音量

https://youtu.be/1Nv-vPBA0fI

P.S. 鄰居最後使用向量反射裝置反射音波，解決了噪音問題

Input

One line contains 10 integers (S1 - S10), separated by spaces 1 <= S1, S2, ... , S10 <= 20000

Output

One integer (the maximum noise)

Loader Code

Your code will be judge using this program:

Sample1

Input

```
2035 1092 15 325 2305 2098 54 1593 1827 813
```

Output

```
30896
```

Sample2

Input

```
1465 546 349 1804 2467 602 1296 1615 2040 1157
```

Output

```
28040
```

循環房間 - Looping Rooms

Description

背景 / Story

阿超做了一個夢。在夢裡，有個熟悉的聲音引導它到了許多間房子，每間裡面都有一間大廳和許多房間。
每間房間都恰有兩扇門，一扇只能進，一扇只能出，並連接到其他房間；
阿超走進其中一間房間之後持續往前，發現只要它走過了數間房間一定會回到最一開始進去的地方，如此形成一個循環後，他便會被傳送回到大廳，可以選擇一間新的房間進去。

起床之後，阿超覺的這個夢很有趣，便憑借他過人的記憶力紀錄下了每間房間的編號以及他們通往的房間。
請幫它幫阿超撰寫一個程式，找出每棟房子的房間循環架構。

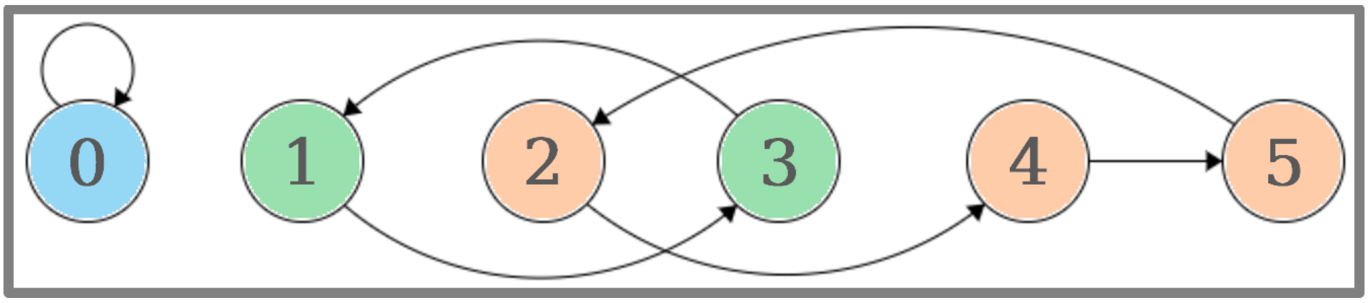
David dreamed a dream. (Yes, his name is David from now on. "Chao" is just waaaay too weird.)
In the dream, a familiar voice guided him to a series of houses. Every of them have a main hall and numerous rooms.

**Every single room has exactly two doors - one can only be used to enter the room, and one can only be used to leave, leading to another room.
**David found that after he entered a room from the hall, once he continue traveling into the next room, he will definitely go back the the first entered room, then he'll be allowed to go back to the main hall, available to choose a new room to enter.

After David woke up, he thought that the dream was interesting, so he exerted his memorizing gift and recorded the rooms and which room they leads to, for all the houses.\

輸入輸出說明 / Input-Output Description

Input 會描述房間的連接狀況。**第一行代表房間數量（ 編號 0 ~ N-1 ）， 接下來 N 行依序代表第 i 個房間的出口連接到哪個房間**。請參考右方的 Sample 1，第一行表示有6個房間，接下來描述 6 個房間的出口：第 0 個房間的出口是第 0 個房間，第 1 個房間的出口是第 3 個房間...以此類推。本範例中房間的連接情況如下圖所示：\



其中**圓圈代表房間編號，箭頭代表出口⟶入口的連接。**
各個顏色標出了房間形成環的組合，共三個，因此輸出第一行為 "3 rings"。

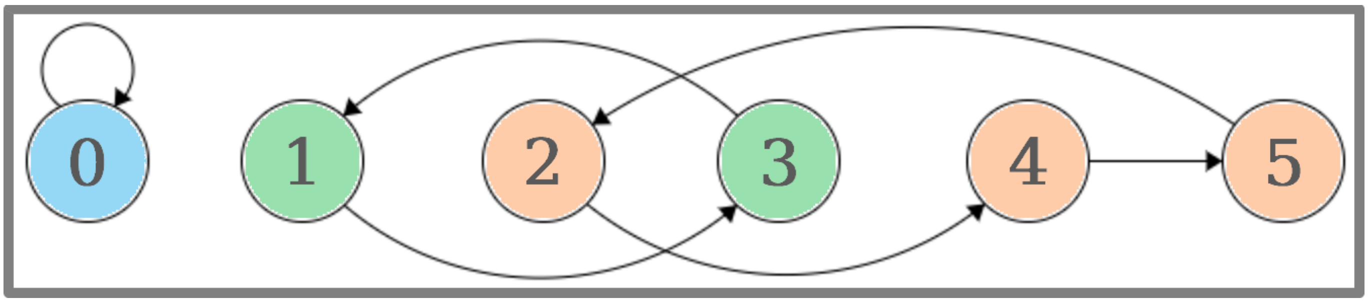
接著根據以下規則印出各個循環：

- 各個循環**之中**以**最小的房間編號為起點**，以`A -> B`的方式連接，直到回到該房間。
    - 也就是說，橘色循環須印出 `2 -> 4 -> 5 -> 2`。
- 各個循環**之間**相比，由**比較小的起始房間編號的優先**印出。
    - 也就是說，本題順序為藍色(0)、綠色(1)、橘色(2)

因此，正確輸出為：

```
3 rings
0 -> 0
1 -> 3 -> 1
2 -> 4 -> 5 -> 2
```

The input describes the room connections. **The first line tells how many rooms are there (indexed 0 ~ N-1)**, then **the following N lines tells which room the exiting door leads to for room i**. Take Sample 1 as example, first line states that there are 6 rooms, and following N lines state the next room for each room, e.g. 0th room leads to 0th room, 1st room leads to 3rd room. The topology of the rooms in Sample 1 is illustrated as below:



The vertices(circles) represents the room and the index, and the edges(arrows) represents the exit⟶enter connections.
As you can see, each color denotes each "rings" that were formed, 3 in total, so the first line of the output should be "3 ring(s)".

Then, follow these rules to print out the rings:

- When printing **each rings**, **Start from the room with the smallest index.** The connections should be formatted as
  `A -> B`and ends at where you started.
  - That is, the orange ring should be printed as `2 -> 4 -> 5 -> 2`.
- **Among the rings**, those having the **smaller starting-room index should be printed first**.
  - That is, the order of sample 1 should be blue(0), then green(1), then orange(2).

Hence, the correct output is:

```
3 rings
0 -> 0
1 -> 3 -> 1
2 -> 4 -> 5 -> 2
```

Input

The first line contains a integer N, $1 \le N \le 500$ Following N lines, each consists an integer $K_0 \sim K_{n-1}$, $0 \le K \le N-1$ $s \ne t \Rightarrow K_s \ne K_t$

Output

The first line should print out how many loops are there, as "N rings" then, print the loops, one in each line, as "K_0-> K_1 -> ... -> K_n -> K_0"

Loader Code

Your code will be judge using this program:

Sample1

Input

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

Output

```
3 rings
0 -> 0
1 -> 3 -> 1
2 -> 4 -> 5 -> 2
```

Sample2

Input

```
15
13
3
4
14
5
10
8
6
1
7
2
11
9
0
12
```

Output

```
4 rings
0 -> 13 -> 0
1 -> 3 -> 14 -> 12 -> 9 -> 7 -> 6 -> 8 -> 1
2 -> 4 -> 5 -> 10 -> 2
11 -> 11
```

Lec09

Alpha compositing

Description

在計算機圖形學領域，Alpha 合成（英語：alpha compositing）是一種將圖像與背景結合的過程，結合後可以產生部分透明或全透明的視覺效果。 (By Wikipedia)

在電腦當中，圖片就是一個含有一堆像素的二維陣列，每一個像素都是一個顏色，該顏色可以用數字來表示。因此，一張 m x n 大小的圖片，在電腦中可以用一個 m x n 大小的矩陣來表示。

要進行 Alpha 合成，必須先準備兩張圖片：一張前景 A，一張背景 B。兩張圖當中的每個像素分別以`Ca`, `Cb` 表示，而兩張圖的透明度則分別以 `Aa`, `Ab` 表示。透明度會介於 0 到 1 之間，0 代表完全透明，1 代表完全不透明。

有了以上參數，我們想要計算兩張圖片合成起來之後的圖片 O，這張合成後的圖片中的每個像素以 `Co` 表示，則 `Co` 可以透過以下算式得知：

`Co = (Ca * Aa + Cb * Ab * (1 - Aa)) / (Aa + Ab * (1 - Aa))`

Alpha 合成的用處非常廣泛，例如：在渲染網頁畫面時，如果有某些畫面元件希望顯示成半透明的，可以使用 CSS 中的 opacity 濾鏡來指定元件的透明度，這個 opacity 濾鏡就是使用了Alpha合成的演算法進行畫面渲染的。

現在，給你一張照片 `image` 、這張照片的長寬`width`, `height`、這張照片的透明度`alpha`，和一個不透明純色背景的顏色 `background_color`，請你將這張照片跟純色背景進行 Alpha 合成。

Input

The first line contains foreground's alpha value (floating number range in [0, 1]), and background color (integer number range in [0, 255]). The second line contains width and height of the foreground image, both are integer number range in [0, 255]. The following lines contain each pixel of the foreground image.

Output

The composited image.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define SIZE 256

void alpha_composite(unsigned int image[][SIZE], unsigned int width, unsigned int
height, float alpha, int background_color);
```

```
int main()
{
    unsigned int width, height, image[SIZE][SIZE] = {0}, backgounrd_color;
    float alpha;

    scanf("%f %u", &alpha, &backgounrd_color);
    scanf("%u %u", &width, &height);
    for (int i = 0; i < height; ++i)
    {
        for (int j = 0; j < width; ++j)
        {
            scanf("%u", &image[i][j]);
        }
    }

    alpha_composite(image, width, height, alpha, backgounrd_color);

    for (int i = 0; i < height; ++i)
    {
        for (int j = 0; j < width; ++j)
        {
            printf("%u ", image[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Sample1

Input

```
0.5 251
10 5
234 156 247 130 188 52 72 133 70 205
237 232 116 122 79 226 162 138 119 2
225 243 179 164 169 9 20 171 192 108
141 115 234 250 222 208 24 216 141 22
213 96 198 205 122 35 170 151 68 162
```

Output

```
242 203 249 190 219 151 161 192 160 228
244 241 183 186 165 238 206 194 185 126
238 247 215 207 210 130 135 211 221 179
```

```
196 183 242 250 236 229 137 233 196 136
232 173 224 228 186 143 210 201 159 206
```

Sample2

Input

```
1 125
16 16
42 161 251 230 134 110 133 47 21 8 215 104 221 73 248 255
190 250 248 252 247 38 101 143 15 74 214 111 221 27 109 143
17 142 81 251 152 232 66 22 128 94 239 172 242 245 142 197
9 41 139 87 107 78 151 20 144 110 222 16 14 26 91 38
48 233 39 99 54 49 253 45 48 188 181 246 84 6 100 83
171 152 121 58 62 173 3 170 34 74 22 6 18 203 152 40
100 164 135 204 11 115 15 107 192 60 123 150 207 197 165 91
63 116 175 136 0 122 111 10 25 100 114 181 198 33 66 86
161 151 138 254 95 59 204 77 24 222 242 202 109 180 129 62
216 142 229 233 108 29 86 199 135 190 188 52 68 114 190 82
39 162 244 70 8 121 91 82 146 51 255 79 100 89 72 17
27 217 196 19 21 129 139 180 71 1 210 215 22 207 130 158
220 20 48 227 212 134 17 222 213 5 64 202 99 50 230 168
90 23 11 60 20 229 86 33 44 63 179 227 46 109 53 87
99 252 35 136 228 170 155 51 225 224 104 97 8 106 220 68
135 162 169 35 90 177 219 253 76 70 223 219 255 240 241 233
```

Output

```
42 161 251 230 134 110 133 47 21 8 215 104 221 73 248 255
190 250 248 252 247 38 101 143 15 74 214 111 221 27 109 143
17 142 81 251 152 232 66 22 128 94 239 172 242 245 142 197
9 41 139 87 107 78 151 20 144 110 222 16 14 26 91 38
48 233 39 99 54 49 253 45 48 188 181 246 84 6 100 83
171 152 121 58 62 173 3 170 34 74 22 6 18 203 152 40
100 164 135 204 11 115 15 107 192 60 123 150 207 197 165 91
63 116 175 136 0 122 111 10 25 100 114 181 198 33 66 86
161 151 138 254 95 59 204 77 24 222 242 202 109 180 129 62
216 142 229 233 108 29 86 199 135 190 188 52 68 114 190 82
39 162 244 70 8 121 91 82 146 51 255 79 100 89 72 17
27 217 196 19 21 129 139 180 71 1 210 215 22 207 130 158
220 20 48 227 212 134 17 222 213 5 64 202 99 50 230 168
90 23 11 60 20 229 86 33 44 63 179 227 46 109 53 87
99 252 35 136 228 170 155 51 225 224 104 97 8 106 220 68
135 162 169 35 90 177 219 253 76 70 223 219 255 240 241 233
```

Sample3

Input

```
0 139
32 32
71 14 124 73 209 208 203 242 86 53 150 109 149 19 134 72 102 182 193 21 3 15 233
161 24 134 67 112 41 3 204 0
87 219 33 125 141 184 68 250 158 231 176 94 14 24 42 49 212 28 163 111 156 10 129
240 143 48 77 11 52 77 180 109
192 153 158 187 57 16 73 52 14 222 107 107 157 10 187 13 252 90 196 44 27 136 126
9 53 252 225 70 92 240 113 125
134 111 205 132 193 139 171 97 218 120 37 253 5 135 233 223 105 28 151 161 79 151
168 30 4 22 206 107 89 86 186 209
30 248 40 175 144 39 129 83 221 254 240 44 199 36 39 22 86 90 18 94 255 5 38 27 15
14 89 243 20 29 123 207
188 57 33 141 123 151 111 57 135 117 100 151 82 80 238 166 18 143 128 71 142 197
195 171 177 112 185 246 159 16 205 232
126 22 114 225 145 170 145 228 30 81 21 158 195 9 12 222 131 50 252 57 2 75 130 96
0 77 156 50 125 236 182 12
167 206 166 85 181 38 33 17 108 39 112 169 186 0 56 207 65 68 18 11 235 169 3 64
195 170 173 118 12 170 52 63
182 192 113 123 225 76 62 139 15 202 63 178 118 105 103 187 177 89 103 41 242 52
237 27 110 149 219 43 108 144 73 7
250 229 123 179 88 226 48 255 191 200 249 53 217 252 217 136 54 245 7 73 76 54 167
2 152 34 210 221 55 175 116 67
237 249 166 3 230 128 232 74 3 191 190 225 166 64 142 244 60 234 53 31 34 144 200
182 220 153 121 43 163 12 212 64
44 197 180 4 245 131 29 132 252 246 204 245 25 175 205 177 147 44 119 22 171 211
138 209 101 81 98 180 80 164 105 116
201 209 0 179 236 169 251 96 184 209 183 163 5 238 208 74 114 31 77 31 194 121 40
171 12 73 44 183 237 163 251 163
228 252 126 205 81 17 164 46 197 184 216 57 42 193 241 104 46 187 27 211 115 1 87
245 207 177 15 39 53 220 247 106
61 172 108 160 254 172 51 151 200 7 49 59 87 218 152 74 67 173 236 41 244 18 77 4
34 186 66 95 76 179 220 201
153 61 93 152 42 240 12 54 223 4 82 184 24 199 55 9 205 81 53 102 58 16 123 147 81
24 142 62 155 202 144 90
203 220 4 55 188 242 108 48 112 14 237 102 232 234 36 99 227 142 162 116 214 108
252 71 237 185 14 58 113 166 29 193
207 119 163 102 29 106 197 220 151 249 109 93 49 57 241 56 201 213 157 98 54 37 35
110 146 233 98 90 8 126 29 123
226 225 91 42 44 107 110 223 89 239 44 26 233 175 27 127 208 170 53 152 27 179 148
195 103 163 84 38 44 149 87 73
62 47 248 207 234 90 51 83 80 227 149 198 40 223 165 96 224 181 204 42 39 168 116
96 249 226 121 177 24 180 241 213
79 96 209 53 11 192 96 135 205 184 104 163 188 19 63 242 106 68 3 246 195 93 193
40 54 64 172 207 115 214 63 241
58 80 113 141 114 47 246 104 220 127 135 238 10 122 78 69 154 232 180 190 63 93
189 153 2 216 174 49 40 144 42 72
```

```
60 59 164 170 28 106 201 52 184 10 147 54 182 79 43 214 137 69 18 93 222 226 236
178 51 1 178 133 24 191 198 38
0 238 160 116 6 204 9 12 92 96 146 158 231 139 49 127 133 95 43 240 149 111 191
130 129 209 39 162 240 215 55 175
86 210 204 185 249 150 221 252 137 66 98 113 147 205 35 199 190 114 94 181 198 26
134 195 191 82 227 15 40 143 216 164
248 172 238 13 222 190 18 197 236 199 201 236 86 229 202 48 135 143 178 96 248 161
243 18 129 250 221 226 178 217 194 178
233 144 90 11 147 123 178 249 124 90 65 109 132 180 236 222 33 198 253 93 99 14
231 198 130 116 111 170 100 156 84 51
137 236 51 147 157 254 223 208 91 14 30 103 37 113 238 40 35 168 13 90 110 94 7
219 6 82 225 66 209 51 110 120
114 145 158 127 114 239 155 166 90 107 203 116 146 39 251 111 169 227 197 28 29 9
98 56 156 164 23 172 155 190 65 39
201 220 255 205 12 200 110 185 113 54 39 39 219 178 133 139 177 11 142 126 53 139
170 134 132 186 187 144 102 114 222 42
175 206 159 47 173 66 59 146 188 102 48 23 169 209 18 173 244 118 191 176 35 127
212 141 60 178 48 151 240 49 36 46
185 204 72 221 136 184 144 103 101 213 46 93 94 194 134 147 210 21 51 122 48 225
33 111 196 4 85 133 199 101 209 176
```

Output

```
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
```

```
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139 139
139 139 139 139 139 139 139 139 139 139 139 139
```

Combination Again (Oh come on)

Description

Although you guys have done Combination in lecture 7 before (right?), you all notice that a factorial of a large number may cause overflow even with type `unsigned long`(what a nightmare. We all have learnt that there is another definition of combination:

C(n, m) =

1. `1`, if m == 0 or m == n

2. `C(n - 1, m - 1) + C(n - 1, m)`, otherwise

Please try to calculate the combination again with this definition.

Input

A line with two integers represent n and m respectively. (1 <= m <= n <= 30)

Output

An integer which represents C(n, m).

Loader Code

Your code will be judge using this program:

Sample1

Input

```
25 8
```

Output

```
1081575
```

Sample2

Input

```
28 25
```

Output

```
3276
```

Passcode of Treasure Box

Description

In Henry's dream, there are 10000 treasure boxes. A mysterious sound says: past paper of PD1 is in the nth treasure box.

Each treasure box has its own passcode. The rule of the passcode is as following:

The passcode of the **1st** treasure box is `1`
For other boxes, if **n is an even number**, the passcode is the same as the`(n/2)th treasure box`.
If **n is an odd number**, the passcode is the sum of the passcode of `(n + 1)th treasure box` and that of `(n - 1)th treasure box`.

Please help Henry to find the passcode.

After Henry has found the passcode and opened the treasure box, there is a letter:
Here are all of the past paper, but the questions in this midterm are all newly set by the TAs. XD

小明做了一個夢，夢裡有 10000 個寶箱，奇妙的聲音告訴他，程設一期中考考古在第 n 個寶箱，想要的話可以全部給你，去找吧...

所有寶箱都有密碼，密碼規則是這樣的：

第一個寶箱，密碼是 1
其餘的寶箱，若 n 是偶數，密碼和`第 n / 2 個寶箱`相同
如果 n 是奇數，密碼是`第 n + 1 個寶箱` 與 `第 n - 1 個寶箱` 密碼的總和

請問你能幫小明猜中密碼嗎？

後續...
小明猜中了！
打開寶箱一看，發現裡面只有一張字條，
想要考古通通都給你，但是這次期中考題目都是助教新出的，加油XD

Input

1 <= n <= 10000

Output

An integer number of passcode

Loader Code

Your code will be judge using this program:

Sample1

Input

```
7
```

Output

```
3
```

Sample2

Input

```
3
```

Output

```
2
```

Secret Value

Description

~~2021/11/05 Updated Description! (Incorrect, Removed)~~

[2021/11/09 **Added information about calculating GCD(a, 0).**]{.underline}

David is trying to calculate the "secret value" of numbers.
The secret value of integer N is calculated as:

1. **If N is a one-digit integer, the secret value is itself.**
2. Split the number into upper part and lower part.
   If there are an odd number of digits, the middle digit belongs to the lower part.
   - The integer 121088 gets divided into 121 and 088.
   - The integer 12345 gets divided into 12 and 345
3. Find the two parts' **greatest common divisor(GCD)**. The result is called k.
   - Please keep in mind that **GCD(a, 0) = GCD(0, a) = |a|.**
   - The result of GCD(a,b) is the greatest non-negative integer k satisfices that a = ck, b = dk. Read This to know more.
4. Sum up N **itself** and the **secret value of** k.

David hate huge numbers, so you MUST answer in modular 10007.
Your work will be judged by [**running the loader section**]{.underline}**, so please implement the functions provided!**

Example (Sample Input 1)

For N = 2424, we can split it into 24 and 24, whose GCD is also 24, so the result k of 2424 is 24.
Now we need to know the secret number of 24 to sum up with N.

For N= 24, we can split it into 2 and 4, whose GCD is 2, so the result k of 24 is 2.

Since 2 has only one digit, the secret value of 2 is 2

Hence, we can get the secret value of [2424 = 2424 + secret value of 24 = 2424 + 24 + secret value of 2 = 2450]{.underline}

Input

A single integer N 1 ≦ N ≦ 2,147,483,647

Output

The secret value of N, mod 10007

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define MODULUS 10007
int getSecret(int n);
int gcd(int a, int b);

int main(){
    int n;
    scanf("%d", &n);
    printf("%d", getSecret(n));
    return 0;
}
```

Sample1

Input

```
2424
```

Output

```
2450
```

Sample2

Input

```
99
```

Output

```
108
```

Sample3

Input

```
121088
```

Output

```
1016
```

Lec09-2

Complete Binary Tree Traversal

Description

Tree is a widely used data structure.

A **tree** looks like:

```
Tree:
         1
       /   \
     2       3
   /   \   /   \
  4     5 6     7
```

can be stored in an 8 elements integer array

```
 [no_use, 1, 2, 3, 4, 5, 6, 7]
```

Child node:

```
The lower nodes 2、3 linked to 1 are called "the node 1's children".
```

Binary tree is define as :

```
For each node, it has at most two children.
```

Complete binary tree can be seen as :

```
The tree node is "from top to down, left to right" full.

Is a complete binary tree:
         1
       /   \
     2       3
   /   \   /
  4     5 6

Not a complete binary tree:
         1
       /   \
     2       3
       \
        5
```

There are 3 kind of simple traversal can be implemented on tree :

Pre-order traversal :

visit current node -> visit left child -> visit right child

In-order traversal :

visit left child -> visit current node -> visit right child

Post-order traversal :

visit left child -> visit right child -> visit current node

Please implement the Complete Binary Tree Traversal.

Input

The first line is what kind traversal should be implemented. '0' for pre-order traversal '1' for in-order traversal '2' for post-order traversal The second line is the number of nodes in this tree. The third line is a integer list represent the tree.Each integer seperated by a space. the number of tree node n is ranged in: 0 <= n <= 5000

Output

A sequential integer list show the traversal result. Each visited node is seperated by a space.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
0
10
1 2 3 4 5 6 7 8 9 10
```

Output

```
1 2 4 8 9 5 10 3 6 7
```

Sample2

Input

```
1
10
1 2 3 4 5 6 7 8 9 10
```

Output

```
8 4 9 2 10 5 1 6 3 7
```

Sample3

Input

```
2
10
1 2 3 4 5 6 7 8 9 10
```

Output

```
8 9 4 10 5 2 6 7 3 1
```

Quicksort

Description

Updated at 11/28: One more sample revealed. Please refer to sample 3.

Quicksort is a divide-and-conquer sorting algorithm. You may have learnt it in PD1 class, but if you want to understand it better, the best way is to implement it yourself!

The algorithm of quicksort could be describe as following:

1. You have an unsorted array, the index of first element is called "low", and the index of the last one is called "high".

2. Arbitrarily pick a number from the given array as "pivot". The "pivot" is used to "divide" the array into two sub-arrays, according to whether they are less than or greater than the pivot. In this question, **we require you to always pick the last number in array as "pivot"**.

3. Iterate through the array (except the "pivot") to find the index of "pivot". **In this iteration, we keep moving the numbers that are smaller than "pivot" or equal to "pivot" to the front of the array, and keep track of the index of the last number of them**. After we finishing iterating, the index of "pivot" is next to the last number that is smaller than it.

4. **Move "pivot" to the position we found in step 2**. At this point, you have "divide" the original array into two sub-arrays, but they both are still unsorted.

5. Repeat step 1 - 4 on two sub-arrays recursively, **until you find the length of divided sub-array is 1 or 0**, which is trivially sorted.

We can illustrate this process with array `8 5 1 9 10 7 3 2 4 6`:

```
   l                   h
1. 8 5 1 9 10 7 3 2 4 6 // you have an unsorted array, the first index is low, the
last is high

   l                   h
2. 8 5 1 9 10 7 3 2 4 6 // pick the last one as "pivot", in this case, it is 6
```

```
                      p

   l                  h
3. 8 5 1 9 10 7 3 2 4 6 // move those numbers that are smaller than 6 to the front
   i                  p
   8 5 1 9 10 7 3 2 4 6 // 8 is bigger than 6, no action
 s i                  p
   5 8 1 9 10 7 3 2 4 6 // 5 is smaller than 6, move to the front
   s i                p
   5 1 8 9 10 7 3 2 4 6 // 1 is smaller than 6, move to the front
     s i              p
   5 1 8 9 10 7 3 2 4 6 // 9 is bigger than 6, no action
     s   i            p
   5 1 8 9 10 7 3 2 4 6 // 10 is bigger than 6, no action
     s      i         p
   5 1 8 9 10 7 3 2 4 6 // 7 is bigger than 6, no action
     s        i       p
   5 1 3 9 10 7 8 2 4 6 // 3 is smaller than 6, move to the front
       s        i     p
   5 1 3 2 10 7 8 9 4 6 // 2 is smaller than 6, move to the front
         s        i   p
   5 1 3 2 4 7 8 9 10 6 // 4 is smaller than 6, move to the front
           s        i p

   l                  h
4. 5 1 3 2 4 6 8 9 10 7 // move "pivot" to the center of two sub-arrays
             p

   l       h
5. 5 1 3 2 4 6 8 9 10 7 // repeat it on sub-arrays recursively
           p
-> 1 3 2 4 5 6 8 9 10 7
         p

...

5. 1 2 3 4 5 6 7 8 9 10 // Sorted!
```

It is better that we can trace the sorting process with our eye than with our mind. So besides implementing the sorting algorithm above, **we need you to print out the result of step 4 everytime you "conquer" the sub problem**. You need to use bracket `[ ]` to show the range of sub-array, and put asterisk `*` in front of the pivot in that sub-array. Please refer to the sample output.

Hope you can totally understand this cool algorithm after this exercise 😄

Input

The first line is an integer n (1 <= n <= 500), indicating the number of numbers in the given array. The following line is an unsorted array with n integer numbers. Notice that there may be duplicated value in the array.

Output

The process of quick sort, line by line. And the final line is the sorted array. Please refer to samples.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define N 500
#define swap(x, y) {int tmp = x; x = y; y = tmp;}

void quicksort(int a[], int low, int high);
int partition(int a[], int low, int high);

int size;

int main(void)
{
    int array[N];
    scanf("%d", &size);

    for (int i = 0; i < size; i++)
    {
        scanf("%d", &array[i]);
    }

    quicksort(array, 0, size - 1);

    for (int i = 0; i < size; i++)
    {
        printf("%d ", array[i]);
    }
    return 0;
}
```

Sample1

Input

```
10
8 5 1 9 10 7 3 2 4 6
```

Output

```
[ 5 1 3 2 4 *6 8 9 10 7 ]
[ 1 3 2 *4 5 ] 6 8 9 10 7
[ 1 *2 3 ] 4 5 6 8 9 10 7
1 2 3 4 5 6 [ *7 9 10 8 ]
1 2 3 4 5 6 7 [ *8 10 9 ]
1 2 3 4 5 6 7 8 [ *9 10 ]
1 2 3 4 5 6 7 8 9 10
```

Sample2

Input

```
1
1
```

Output

```
1
```

Sample3

Input

```
10
2 7 10 1 9 6 3 2 5 6
```

Output

```
[ 2 1 6 3 2 5 *6 9 10 7 ]
[ 2 1 3 2 *5 6 ] 6 9 10 7
[ 2 1 *2 3 ] 5 6 6 9 10 7
[ *1 2 ] 2 3 5 6 6 9 10 7
1 2 2 3 5 6 6 [ *7 10 9 ]
1 2 2 3 5 6 6 7 [ *9 10 ]
1 2 2 3 5 6 6 7 9 10
```

Sample4

Input

```
10
10 11 12 13 14 15 16 17 18 19
```

Output

```
[ 10 11 12 13 14 15 16 17 18 *19 ]
[ 10 11 12 13 14 15 16 17 *18 ] 19
[ 10 11 12 13 14 15 16 *17 ] 18 19
[ 10 11 12 13 14 15 *16 ] 17 18 19
[ 10 11 12 13 14 *15 ] 16 17 18 19
[ 10 11 12 13 *14 ] 15 16 17 18 19
[ 10 11 12 *13 ] 14 15 16 17 18 19
[ 10 11 *12 ] 13 14 15 16 17 18 19
[ 10 *11 ] 12 13 14 15 16 17 18 19
10 11 12 13 14 15 16 17 18 19
```

Sample5

Input

```
10
13 12 11 10 9 8 7 6 5 4
```

Output

```
[ *4 12 11 10 9 8 7 6 5 13 ]
4 [ 12 11 10 9 8 7 6 5 *13 ]
4 [ *5 11 10 9 8 7 6 12 ] 13
4 5 [ 11 10 9 8 7 6 *12 ] 13
4 5 [ *6 10 9 8 7 11 ] 12 13
4 5 6 [ 10 9 8 7 *11 ] 12 13
4 5 6 [ *7 9 8 10 ] 11 12 13
4 5 6 7 [ 9 8 *10 ] 11 12 13
4 5 6 7 [ *8 9 ] 10 11 12 13
4 5 6 7 8 9 10 11 12 13
```

Rat in a Maze

Description

There is a $5*5$ maze. A rat wants to walk through the maze and it moves in a vertical or a horizontal direction. Please help the rat find a route from the upper-left block to the bottom-right block.

upper-left block: `(0,0)`

bottom-right block: `(4,4)`

Note: There is at most one route.

Hint: You can use recursion to solve this problem.

Input

A maze is a 5*5 matrix. 'w' and 'r' represent wall and road respectively.

Output

Print out a route in a 5*5 matrix.(1: a route, 0: not a route) If there isn't any route, print out "Can't find the exit!".

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdbool.h>
#define n 5

bool visit(char [][n], int [][n], int, int);
int main(void) {
    char maze[n][n];
    int route[n][n];
    // initialize maze and route matrices
    for (int i=0; i<n; i++) {
        for (int j=0; j<n; j++) {
            route[i][j]=0;
            scanf("%c", &maze[i][j]);
            getchar();
        }
    }
    if (visit(maze, route, 0, 0)) { // (0,0) is a starting point
        for (int i=0; i<n; i++) {
            for (int j=0; j<n; j++)
                printf("%d ", route[i][j]);
            printf("\n");
        }
    } else {
        printf("Can't find the exit!");
    }
    return 0;
}
```

Sample1

Input

```
r w r r r
r w r w w
r r r r r
r w r w w
r w r r r
```

Output

```
1 0 0 0 0
1 0 0 0 0
1 1 1 0 0
0 0 1 0 0
0 0 1 1 1
```

Sample2

Input

```
r w r r r
r w r w w
r r r w r
r w r w w
r w r w r
```

Output

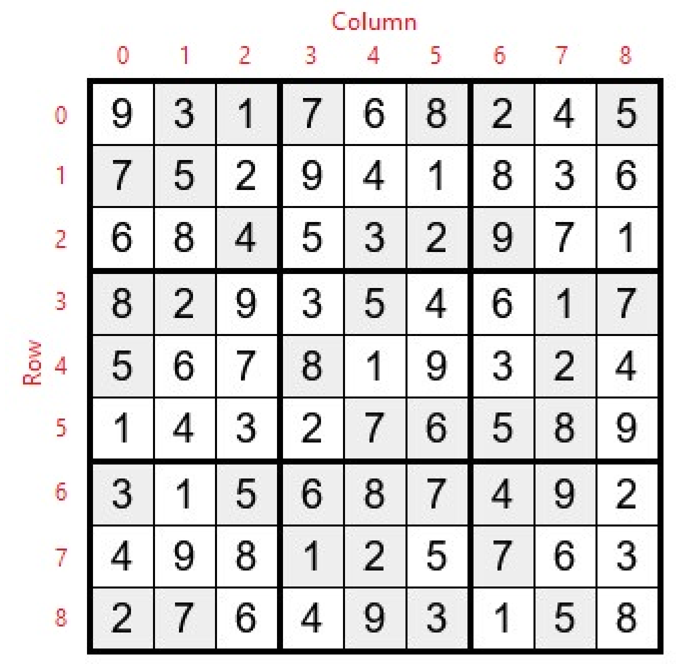
```
Can't find the exit!
```

Sudoku Checker

Description

Sudoku is a logical based, combinatorial number-placement puzzle. The objective is to fill a **9×9** grid with digits so that **each column, each row, and each of the nine 3×3 subgrids that compose the grid contain all of the digits from 1 to 9**.

For example:



	0	1	2	3	4	5	6	7	8
0	9	3	1	7	6	8	2	4	5
1	7	5	2	9	4	1	8	3	6
2	6	8	4	5	3	2	9	7	1
3	8	2	9	3	5	4	6	1	7
4	5	6	7	8	1	9	3	2	4
5	1	4	3	2	7	6	5	8	9
6	3	1	5	6	8	7	4	9	2
7	4	9	8	1	2	5	7	6	3
8	2	7	6	4	9	3	1	5	8

Write a program which checks the input sudoku puzzle and prints out the row number and column number of the points that do not satisfy sudoku's requirements.

Note: You only need to implement `check_sudoku` function.

Input

The parameter of the check_sudoku function is a pointer that points to a two-dimentional integer array, which represents the sudoku puzzle.

Output

Print out the row number and column number of the points (from left to right then top to bottom) that do not satisfy soduku's requirements. Each point printed out is followed by a newline character, and is in the format (row,column).

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define NUM 9

void check_sudoku(int grid_p[][NUM]);

int main(void){
    int grid[NUM][NUM]; // sudoku puzzle
    for(int i = 0; i < NUM; ++i){
        for(int j = 0; j < NUM; ++j){
            scanf("%d", &grid[i][j]);
        }
    }
    check_sudoku(grid);
    return 0;
}
```

Sample1

Input

```
9 3 1 7 1 8 2 4 5
7 5 2 9 4 1 8 3 6
6 8 4 5 3 2 9 7 1
8 2 9 3 5 4 6 1 7
5 6 7 8 1 9 3 2 4
1 4 3 2 7 6 5 8 9
3 1 5 6 8 7 4 9 2
4 9 8 1 2 5 7 6 3
2 7 6 4 9 3 1 5 8
```

Output

```
(0,2)
(0,4)
(1,5)
(4,4)
```

Sample2

Input

```
9 3 1 7 6 8 2 4 5
2 5 2 9 4 1 8 3 6
6 8 4 5 3 2 9 7 1
8 2 9 3 5 4 6 1 7
5 6 7 8 1 9 3 2 4
1 4 3 2 7 6 5 8 9
3 1 5 6 8 7 4 9 2
4 9 8 1 2 5 7 6 3
2 7 6 4 9 3 1 5 8
```

Output

```
(1,0)
(1,2)
(8,0)
```

The coordinate to the past

Description

Matsumoto is a brilliant AI that is built and sent to the past to prevent a war between humans and AI. When on his way back to the past, he will receive a set of encoded coordinates indicating where to go.

To decode the coordinates, he has to follow the following steps:

1. split the received line of characters into two groups: characters at odd positions and characters at even positions (index starts by 0 and from left)

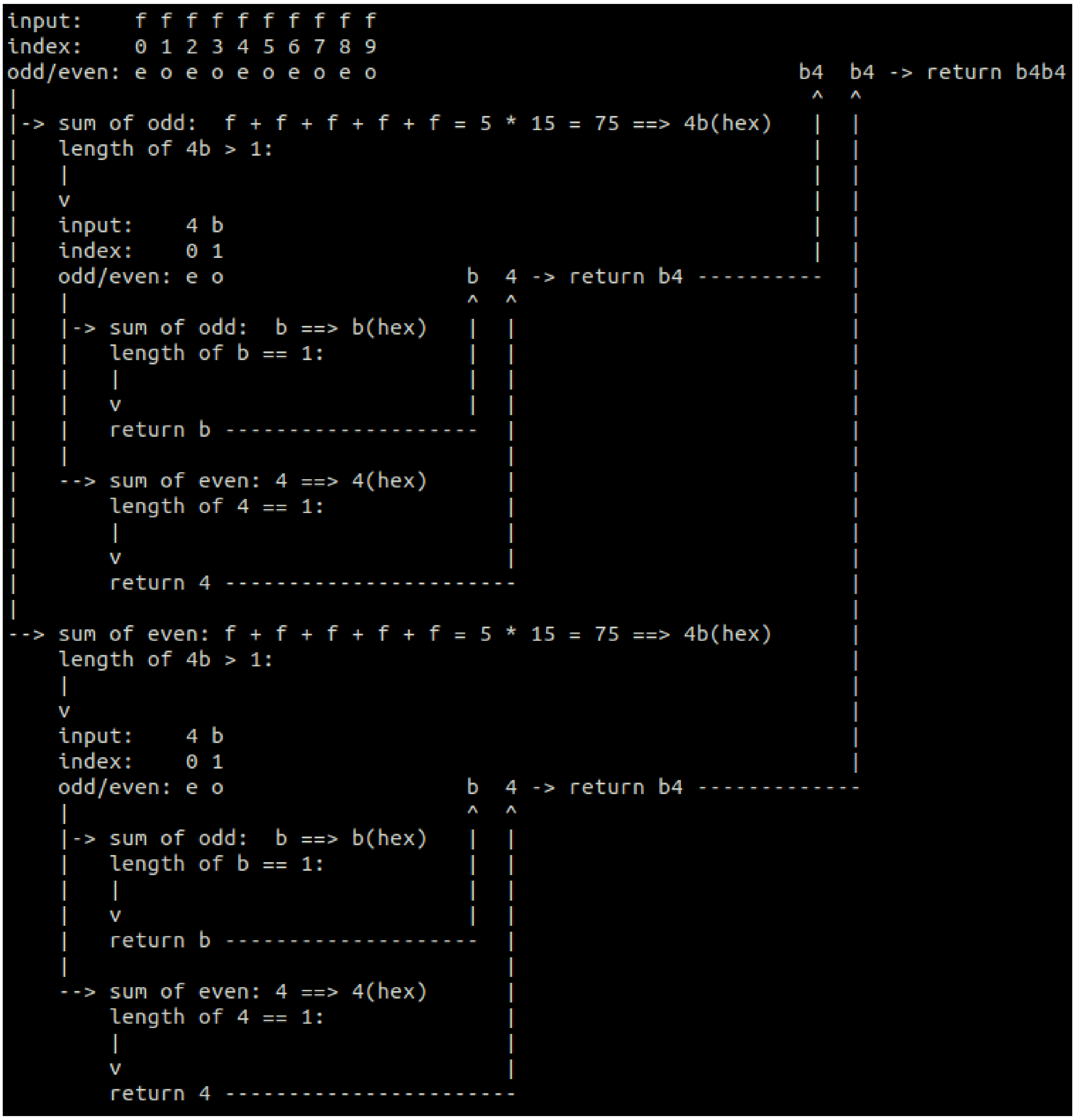
2. sum up the numbers in each group

3. write these two numbers into hexadecimal

4. process the two numbers from step 1 if the length of the number is greater than 1

5. the coordinate is the concatenate of the result of the two numbers (the odd one on the left)

The following figure presents an example of the previously shown steps:



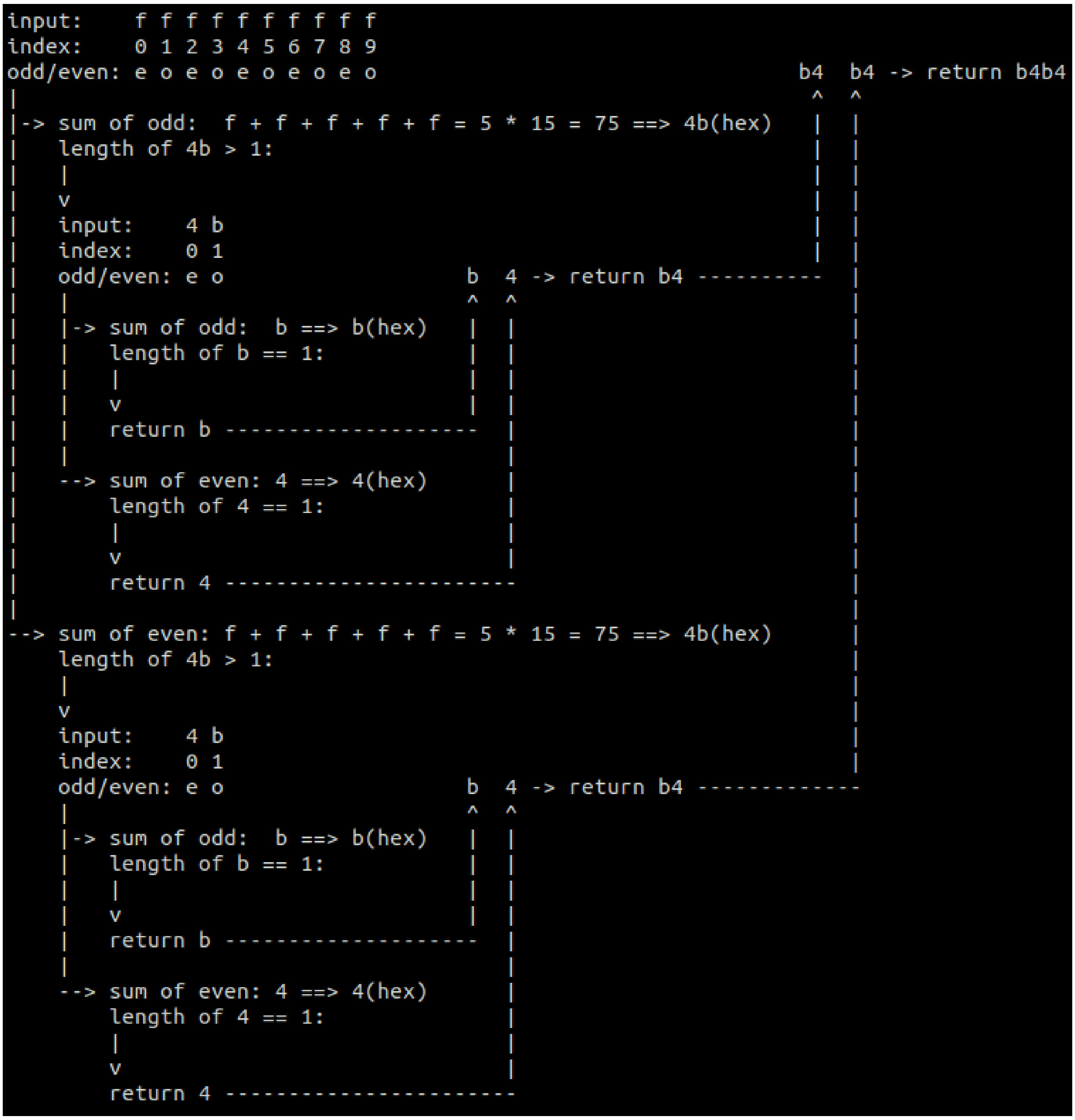
```
input:    f f f f f f f f f f
index:    0 1 2 3 4 5 6 7 8 9
odd/even: e o e o e o e o e o                                 b4  b4 -> return b4b4
|                                                              ^  ^
|-> sum of odd:  f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)   |  |
|   length of 4b > 1:                                          |  |
|   |                                                          |  |
|   v                                                          |  |
|   input:     4 b                                             |  |
|   index:     0 1                                             |  |
|   odd/even: e o                   b  4 -> return b4 ----------  |
|   |                               ^  ^                          |
|   |-> sum of odd:  b ==> b(hex)   |  |                          |
|   |   length of b == 1:           |  |                          |
|   |   |                           |  |                          |
|   |   v                           |  |                          |
|   |   return b --------------------  |                          |
|   |                                  |                          |
|   --> sum of even: 4 ==> 4(hex)      |                          |
|       length of 4 == 1:              |                          |
|       |                              |                          |
|       v                              |                          |
|       return 4 -----------------------                          |
|                                                                 |
--> sum of even: f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)      |
    length of 4b > 1:                                             |
    |                                                             |
    v                                                             |
    input:     4 b                                                |
    index:     0 1                                                |
    odd/even: e o                   b  4 -> return b4 -------------
    |                               ^  ^
    |-> sum of odd:  b ==> b(hex)   |  |
    |   length of b == 1:           |  |
    |   |                           |  |
    |   v                           |  |
    |   return b --------------------  |
    |                                  |
    --> sum of even: 4 ==> 4(hex)      |
        length of 4 == 1:              |
        |                              |
        v                              |
        return 4 -----------------------
```

Please help him to decode the coordinate.

松本是個聰明的人工智慧，創造他的人想要將他送回過去阻止一場人類與人工智慧的戰爭。在他穿越的過程中，他會收到一份經過編碼的目的地座標，想要將座標解碼需要透過以下的步驟：

1. 將收到的所有字元分成奇數位與偶數位兩組（從最左邊的開始編號，並且從0開始）
2. 分別計算兩組數字的和
3. 將兩個相加的結果以16進位表示
4. 若有和的位數大於1的，則重複上述步驟
5. 將兩組數字計算的結果串接起來（奇數組的結果在左邊，偶數組的在右邊）

下面是範例二的示意圖：



```
input:    f f f f f f f f f f
index:    0 1 2 3 4 5 6 7 8 9
odd/even: e o e o e o e o e o                                 b4  b4 -> return b4b4
|                                                              ^  ^
|-> sum of odd:  f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)   |  |
|   length of 4b > 1:                                          |  |
|   |                                                          |  |
|   v                                                          |  |
|   input:    4 b                                              |  |
|   index:    0 1                                              |  |
|   odd/even: e o                   b  4 -> return b4 ----------  |
|   |                               ^  ^                          |
|   |-> sum of odd:  b ==> b(hex)   |  |                          |
|   |   length of b == 1:           |  |                          |
|   |   |                           |  |                          |
|   |   v                           |  |                          |
|   |   return b --------------------  |                          |
|   |                                  |                          |
|   --> sum of even: 4 ==> 4(hex)      |                          |
|       length of 4 == 1:              |                          |
|       |                              |                          |
|       v                              |                          |
|       return 4 -----------------------                          |
|                                                                 |
--> sum of even: f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)      |
    length of 4b > 1:                                             |
    |                                                             |
    v                                                             |
    input:    4 b                                                 |
    index:    0 1                                                 |
    odd/even: e o                   b  4 -> return b4 -------------
    |                               ^  ^
    |-> sum of odd:  b ==> b(hex)   |  |
    |   length of b == 1:           |  |
    |   |                           |  |
    |   v                           |  |
    |   return b --------------------  |
    |                                  |
    --> sum of even: 4 ==> 4(hex)      |
        length of 4 == 1:              |
        |                              |
        v                              |
        return 4 -----------------------
```

請幫助松本解碼他收到的座標。

Input

A newline character terminated string indicates the encoded coordinate. The length of the string will be greater than 1 and less than 10001. Each character represents a hexadecimal, the upper case and the lower case are both possible. 一行以換行字元結尾的字串，代表經過編碼的座標。 該字串的長度介於1到10001間（不包含10001）。字串中每個字元代表一個16進位數字，且大小寫都有可能。

Output

A single line that indicates the decoded coordinate, the letter of the coordinate should be shown in lower case (if any). 解碼後的字串，若字串中有出現英文字母一律以小寫表示。

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1111111111
```

Output

```
55
```

Sample2

Input

```
ffffffffff
```

Output

```
b4b4
```

那個孤單的數字 - The Lonely Number

Description

這個世界是上時時刻刻都在發生著排擠的狀況，就連數字的世界也是。真可憐。
給定 `N` 個的數字，依據以下規則將其全部分成三個數字一組的小組，請你找到無法被分組的那個可憐蟲。

Every second, someone is getting excluded from his group, and so does it happens in the world of numbers. Given N numbers, group them into triads according to the following rules. Please find the lonely pitiful number.

- 每個數組中的三個數字 a, b, c，必須可以滿足 a + b = c。
  The 3 numbers a, b, c in each groups must satisfy the formula a + b = c.
- **提供的數字可能重複，但各個成員只能被使用一次。**
  Each numbers might be given many times, but each occurrence can only be used once.
  - E.g., `input = [1, 2, 3, 1, 2, 3, 8]`, `group = [[1, 2, 3], [1, 2, 3]]`, `output = [8]`
- `N` = 3k + 1, `k` ∈ ℕ，必定可以組成 `k` 個數組。
  `N` = 3k + 1, `k` ∈ ℕ. It is guaranteed that `k` different triads can be made.
- 每組輸入皆只有一個正確的孤單數字，但有**可能有不同的分組方法。**
  Each set of inputs are guaranteed to have only 1 correct lonely number, but **might have multiple ways to group the numbers**.

Sample Description

**nums = [1, 2, 3, 3]
**可以組成`[[1, 2, 3]]`而多餘的 3 會被留下來，因此輸出為 3。
Can be grouped into`[[1, 2, 3]]`, with the redundant 3 getting excluded, so the output should be 3.

**nums = [1, 3, 4, 2, 2, 4, 3, 5, 8, 3]
**可以被分成`[[1, 2, 3], [2, 3, 5], [4, 4, 8]]`，或`[[1, 3, 4], [2, 2, 4], [3, 5, 8]]`，皆會留下一個孤單的 3，因此輸出為 3。
Can be grouped into`[[1, 2, 3], [2, 3, 5], [4, 4, 8]]`or`[[1, 3, 4], [2, 2, 4], [3, 5, 8]]`, both excluding the redundant 3, so the output is also 3.\

Input

First line consists a integer, tells how many numbers will be given. $1 \leq N \leq 50$, $N = 3k + 1$, $k \in \mathbb{N}$ Following is N numbers each separated with a space. $1 \leq k \leq 100$

Output

The single lonely number. You don't need to mind what are the groups.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
4
1 2 3 3
```

Output

```
3
```

Sample2

Input

```
10
1 3 4 2 2 4 3 5 8 3
```

Output

```
3
```

Sample3

Input

```
13
2 8 4 2 2 5 8 3 5 8 3 4 2
```

Output

```
8
```

Lec10

Beverage Shop

Description

Cody has a beverage shop. The following drinks are what he offered:

- Green tea **G** (500mL): 20 NTD
- Black tea **B** (500mL): 20 NTD
- Tieguanyin tea **T** (500mL): 30 NTD
- White gourd tea **W** (500mL): 15 NTD
- Tieguanyin green tea **GT** (Tieguanyin tea 250mL + green tea 250mL): 45 NTD
- White gourd Tieguanyin tea **WT** (Tieguanyin tea 350mL + white gourd tea 150mL): 40 NTD
- Black tea Latte **BM** (black tea 200mL + milk 300mL): 35 NTD
- Tieguanyin Latte **TM** (Tieguanyin tea 200mL + milk 300mL): 45 NTD
- White gourd Latte **WM** (White gourd tea 200mL + milk 300mL): 30 NTD

The sugar and ice level of each drink are adjustable. Assume that there are 5 levels for both sugar and ice:

- Sugar:
    1. No sugar: 0g
    2. Low sugar: 3g
    3. Half sugar: 5g
    4. Less sugar: 7g
    5. Standard: 10g
- Ice:
    1. No ice: 0g
    2. Low ice: 30g
    3. Less ice: 50g
    4. Normal: 70g
    5. Extra ice: 100g

Given the ingredient of each drink and list of sales, please help Cody calculate how much ingredient remained and how much he earned.

Input

The input consists of several lines. The first 7 numbers are the amount of each ingredient. The orders are black tea (L)/ White gourd (L)/Tieguanyin tea (L)/green tea (L)/milk (L)/sugar (kg)/ice (kg). The 2nd line is the number of sales n. The last n lines are the content of each sale, which consists of the type of the beverage, sugar level and ice level.

Output

The first line contains the remained amount of each ingredient. The second line is how much Cody earned.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
2 3 3 1 2 2 2
8
B 5/3
T 3/3
G 2/5
TM 2/4
W 1/3
WT 3/4
WT 3/4
WT 3/5
```

Output

```
1500 2050 1250 500 1700 1964 1440
250
```

Sample2

Input

```
2 1 2 1 3 1 1
4
TM 2/5
WM 4/4
TM 1/3
TM 5/4
```

Output

```
2000 800 1400 1000 1800 980 710
165
```

Sample3

Input

```
2 2 2 1 2 1 2
7
WM 2/4
B 2/1
TM 1/3
WT 1/2
B 4/1
W 1/2
T 4/5
```

Output

```
1000 1150 950 1000 1400 980 1720
200
```

Encryption

Description

加密演算法通常都是不同替換 (substitution) 與重排 (transposition) 的排列組合。
替換與重排的定義分別為：

- **替換 (substitution)**
  以取代的方式將文字作位移，當文字變成另一個文字時就保護了真正的明文意義。像是 Caesar Cipher 與期中考題的 Trithemius Cipher 就是經典的替換加密的範例。

- **重排 (transposition)**
  將原本的字串重新排列順序，舉例來說將字串的順序完全顛倒：`PD1 is good` 變成 `doog si 1DP` 就算是其中一種重排。

在這次的作業裡面，我們要利用下面這幾種方法的排列組合來進行加密：

1. **交換 (swap)**
   將整個文字拆成兩份，前後交換，現在以 `1 2 3 4 5 6` 來舉例：

   改動前：1 2 3|4 5 6

   改動後：4 5 6|1 2 3

2. **位移 (rotate)**
   根據指定的左/右及總位移格數移動文字，現在以向 **左** 位移 **2** 位舉例

   改動前：1 2 3 4 5 6

改動後：`3 4 5 6 1 2`

3. **rail fence cipher**
給定 key 依照要求挪動文字位置。
依照下方範例為例，明文 `P` 對應到的 key 為 `5`，表示它需要被挪動到第五個位置；明文 `D` 對應到的 key 為 `1` ，表示它需要被挪動到第一個位置，依此類推。

key : `5 1 2 6 9 8 4 3 7`

明文 : `P D 1 I S G O O D`

密文 : `D 1 O O P I D G S`

同樣依照下圖為範例，在這裡的 key 長度為 4 ，明文的長度為 11。
在前面八個數字，每四個依照給定的 key 挪動文字位置，最後三個由於缺一個數字，就依照文字原排序不改動順序。

key ：`4 1 3 2`

明文：`P D 1 I`

`S S O G`

`O O D`

密文：`D I 1 P`

`S G O S`

`O O D`

4. **凱薩加密 (Caesar Cipher)**
給定一個密鑰 `k` ，將文字平移 `k`。
舉例來說 `k = 3` ，明文為 `bob, i love you. Alice` 會變成 `ere, l oryh brx. Dolfh`

5. **特里特米烏斯加密 (Trithemius Cipher)**

在凱薩密碼中，每個字元都會被平移一個固定的密鑰 `k`，而特里特米烏斯密碼密碼則會額外多加一個連續的偏移值(會是遞增或遞減，每次移動1)。

舉例來說， k=1、遞增的狀況下，我們想加密TAs are so H4ND5OME，而每個字元的偏移量為：

`T A s a r e s o H N D O M E`

`1 2 3 4 5 6 7 8 9 10 11 12 13 14`

加密文字會是`UCv ewk zw Q4XO5AZS`.

6. **維吉尼亞加密 (Vigenère Cipher)**

維吉尼亞密碼是使用一系列凱薩密碼來加密文字的算法，透過一組密鑰來進行加密。在凱薩密碼中，每個英文字母都會偏移一個固定的數字。例如，在偏移量為3的凱薩密碼中，`A` 會變成 `D`, `b` 會變成 `e`, `y` 會變成 `b`，以此類推。維吉尼亞密碼則是由好幾組不同偏移量的凱薩密碼編排而成，透過一個稱為「密鑰」的字串來決定其偏移值。而這個密鑰僅包含小寫英文字母。

舉例來說，假設密鑰是 `abcde`。密鑰中的每個字母會轉為其對應的數字(例如`a` = `0`, `b` = `1`, ... `z` = `25`)：

英文字母: `a b c d e`

數字: `0 1 2 3 4`

要加密 **TAs are so H4ND5OME.** 這段訊息，我們要將密鑰如下排列：

`a b c d e a b c d e a b c d`

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

接著根據前一步驟排列的密鑰(2.)，對訊息中的每個**英文字母**(1.)做位移來產生加密的文字(3.)，如下：

`T A s a r e s o H N D O M E`

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

`T B u d v e t q K R D P O H`

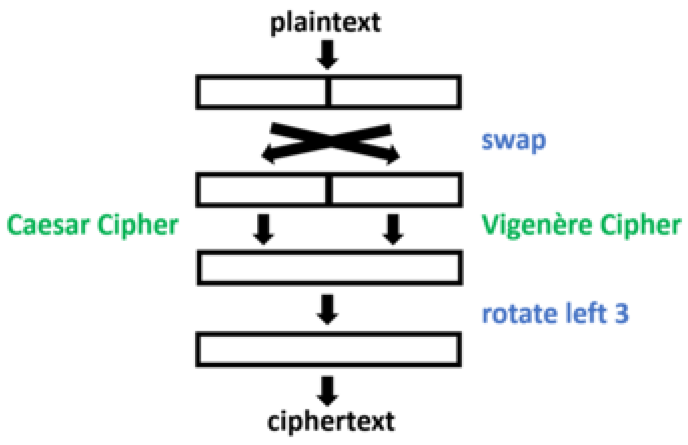
而這段純文字訊息 **TAs are so H4ND5OME.** 會變成 **TBu dve tq K4RD5POH.**，即為加密後的訊息，意思是助教好帥 😉。

註：**凱薩加密、特里特米烏斯密碼密碼、維吉尼亞密碼** 只需轉換英文字母(保留大小寫)且保留其他字元例如阿拉伯數字或標點符號。

現在我們要將上面的六種加密方法模組化，總共有 5 種模組加密方案：

第一種

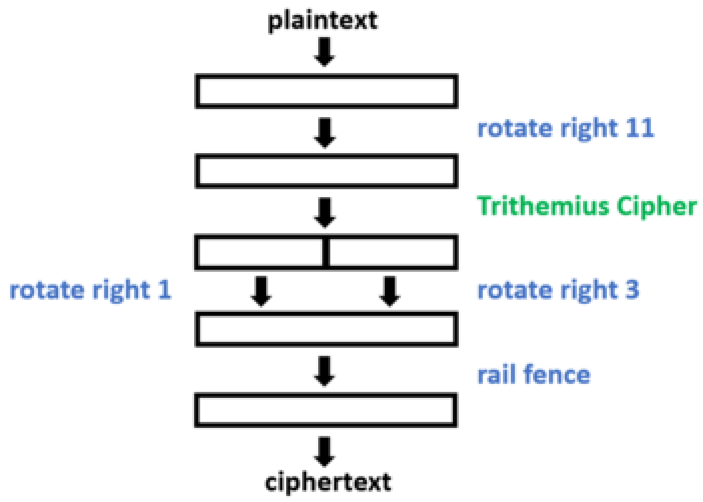
1. 將明文 `SWAP`
2. 把剛剛處理過的結果拆做前後兩半
   - 前半部: `凱薩加密`，密鑰為 `13`
   - 後半部: `維吉尼亞加密`，密鑰為 `meow`
3. 將上一個步驟的結果結合起，向 `左` 位移 `3` 位



第二種

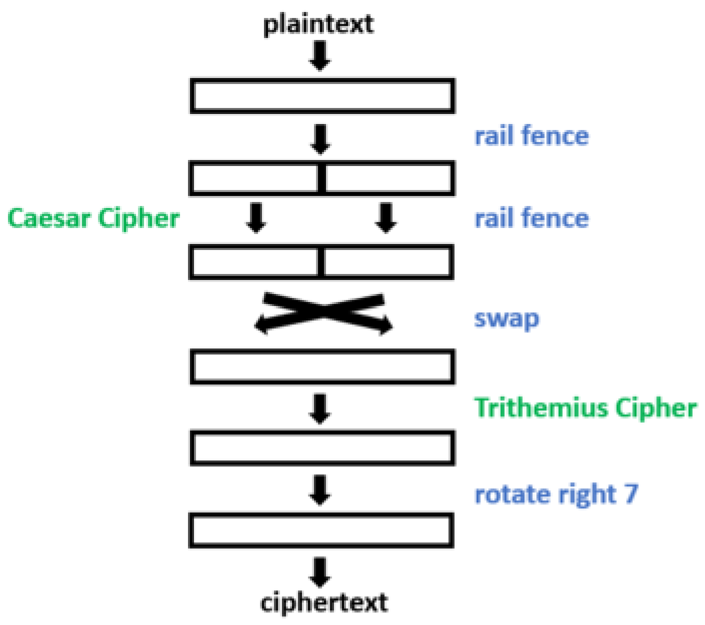
1. 將明文向 `右` 位移 `11` 位

2. 把剛剛處理過的結果做 特里特米烏斯加密 ，遞增，密鑰為 74
3. 將上一步驟結果拆做前後兩半
    - 前半部： 向 右 位移 1 位
    - 後半部： 向 右 位移 3 位
4. 做 `rail fence cipher` ，密鑰為 `4 3 1 2 7 6 5 8`



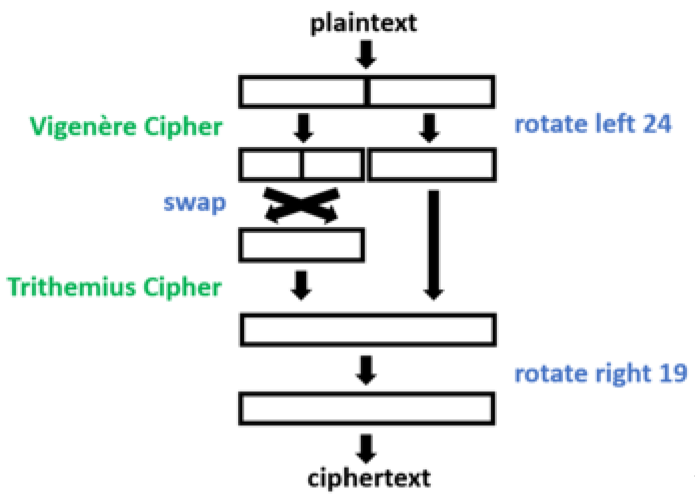
第三種

1. 將明文做 `rail fence cipher` 加密，密鑰為 `3 2 4 1`
2. 把剛剛處理過的結果拆做前後兩半
    - 前半部: 凱薩加密， 密鑰為 8
    - 後半部： `rail fence cipher` 加密，密鑰為 `1 8 4 3 6 5 7 2`
3. 將上一個步驟的結果結合起，做 `SWAP`
4. 做 特里特米烏斯加密，遞減，密鑰為 `602`
5. 將文字向 右 位移 `7`



第四種

1. 將明文拆做前後兩半
    - 前半部依序做
        - 維吉尼亞加密，密鑰為 `vigenere`
        - SWAP
        - 特里特米烏斯加密，遞增，密鑰為 `3`
    - 後半部做向 `左` 位移 `24`
2. 將上一步驟的結果結合起來，向 `右` 位移 `19`



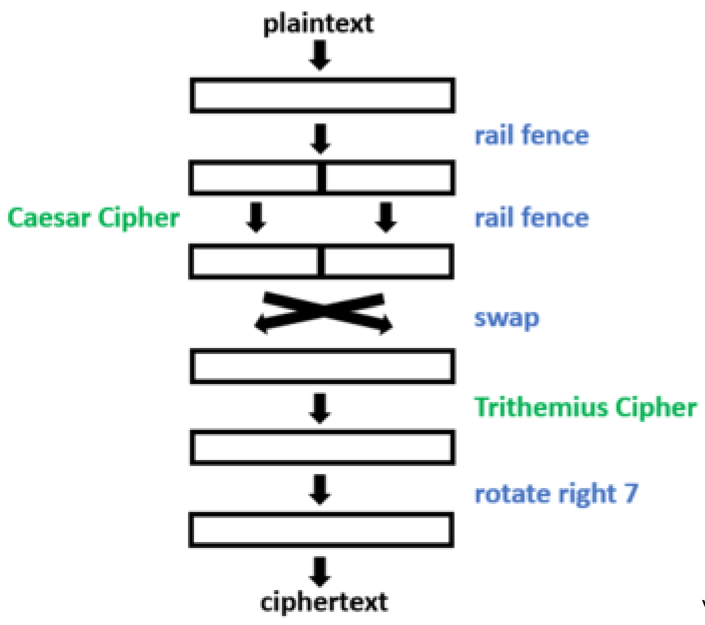
\

第五種

1. 將明文做 `rail fence cipher` 加密，密鑰為 `3 1 7 6 4 5 2 8`
2. 將剛剛處理過的結果拆做前後兩半
    - 前半部： `凱薩加密`，密鑰為 `10`

    - 後半部： `rail fence cipher` ，密鑰為 `2 4 1 3`

3. `SWAP`
4. 做 `特里特米烏斯加密` ，遞增，密鑰為 `52`
5. 將文字向 `右` 位移 `7`



\

請依照上述的加密演算法，印出加密過後的文字

Input

第一行為選擇的加密方案 第二行為明文，皆為 64 個字元

Output

加密後的結果

Loader Code

Your code will be judge using this program:

Sample1

Input

```
3
Welcome to PD1! Welcome to PD1!! Welcome to PD1! Welcome to PD1!
```

Output

```
lp !1Y!ppg pebU 1l Z!Kneev etqJ 1a O!ZcstKy yzoYwa  1J!hiZn nodN
```

Sample2

Input

```
2
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijkl
```

Output

```
zbxvhfdjprnlljhntvrpbzxdjlhfrpnthxfddbzflnjhtrpvbdzxjhflrtpnzxvb
```

Image Editor

Description

In computer, everything is represented by numbers --- so does image. Computer stores image as a matrix. A cell in the matrix is a pixel in the image, and number in the cell represents the color of that pixel.

Today, Ariel wants to do a image editor which can rotate, flip vertically, flip horizontally and crop images.

There are five operations, rotate(**r**), vertical flip(**v**), horizontal flip(**h**), crop(**c**), output(**p**) images.

For example, consider an image as follows:

`| 1 2 3 |`
`| 4 5 6 |`
`| 7 8 9 |`

1. Rotate(**r**)

   Two parameters are rotating the image 90 degrees to right or left(**l**/**r**) and the number of rotations, respectively. For example:

   `r r 10` rotate the image 90 degrees to right 10 times.

   `r l 1` rotate the image 90 degrees to left once.

   If we want to **rotate this image 90 degrees to the right**, we can rotate the matrix like this:

   `| 7 4 1 |`
   `| 8 5 2 |`
   `| 9 6 3 |`

2. vertical flip(**v**)

   "flip" or "mirror" an image in the vertical direction (up-down).

```
| 7 8 9 |
| 4 5 6 |
| 1 2 3 |
```

3. horizontal flip(**h**)

   "flip" or "mirror" an image in the horizontal direction (left-right).

4. crop(**c**)

   Four parameters are left, right, upper, lower boundary. For example:

   `c 1 1 1 3`

   Then the new image is:

   ```
   | 1 |
   | 4 |
   | 7 |
   ```

5. output(**p**)

   Output the current image!

在電腦裡，所有東西都是用數字來表達的------就連圖片也不例外。電腦以矩陣的形式來儲存每一張圖片，矩陣裡的格子代表著圖片的像素，而格子裡的數字就代表了該像素的顏色。

今天 Ariel 想要做個圖片編輯器，可以旋轉、垂直/水平翻轉以及裁切圖片。

共有五種操作，分別為旋轉(**r**)、垂直翻轉(**v**)、水平翻轉(**h**)、裁切圖片(**c**)、輸出圖片(**p**)

如果有一張圖片長這樣：

```
| 1 2 3 |
| 4 5 6 |
| 7 8 9 |
```

經過不同操作，會得到不同的結果。

一、 旋轉(**r**)

後面會有兩個參數，分別為向左/向右旋轉 90 度(**l**/**r**)以及會旋轉幾次，ex:

`r r 10` 向右旋轉 10 次

`r l 1` 向左旋轉 1 次

當我們要對圖片做旋轉時，我們實際上就是在旋轉這個矩陣本身。

如果我們想將這張圖片**向右旋轉 90 度**，那麼就等於是把矩陣旋轉成這樣：

```
| 7 4 1 |
| 8 5 2 |
```

`| 9 6 3 |`

二、垂直翻轉(**v**)

從上向下 180 度翻轉。

```
| 7 8 9 |
| 4 5 6 |
| 1 2 3 |
```

三、水平翻轉(**h**)

從左向右 180 度翻轉。

```
| 3 2 1 |
| 6 5 4 |
| 9 8 7 |
```

四、裁切圖片(**c**)

後面會有四個參數，分別為左邊界、右邊界、上邊界、下邊界，ex:

`c 1 1 1 3`

經過裁切後

```
| 1 |
| 4 |
| 7 |
```

五、輸出圖片(**p**)

將目前的圖片輸出!

Input

The second line contains 2 integers m & n, indicates the dimension of the image. And 1<= m, n <= 100. The following m lines contain n integers, representing the image itself. Each integer in the image is in the range of [0, 1000]. Then, an integer x represents the number of operations. The following x lines are operations, execute operations according to the operation's format.

Output

Output the current image while executing 'p' operation. Each number must follow by a space. You need to print newline at the end of last line.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
2 4
1 2 3 4
5 6 7 8
6
r r 1
p
h
p
c 1 2 3 4
p
```

Output

```
5 1
6 2
7 3
8 4

1 5
2 6
3 7
4 8

3 7
4 8
```

Sample2

Input

```
3 4
274 2 100 5
222 1 50 70
10 44 34 79
8
v
p
c 2 4 1 3
p
r l 11
p
r r 22
p
```

Output

```
10 44 34 79
222 1 50 70
274 2 100 5

44 34 79
1 50 70
2 100 5

2 1 44
100 50 34
5 70 79

79 70 5
34 50 100
44 1 2
```

Poker Hand

Description

There are four people `A` ,`B` ,`C` ,`D` . Each of them has one poker hand which consists of five cards. Each card in the hand will have both a suit (`clubs`, `diamonds`, `hearts`, or `spades`) and a rank (`2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `10`, `jack`, `queen`, `king`, or `ace`). Write a program that prints out their order **from highest to lowest**.

The categories of hand, from highest to lowest, are listed below:

1. straight flush (both a straight and a flush)
2. four-of-a-kind (four cards of the same rank)
3. full house (a three-of-a-kind and a pair)
4. flush (five cards of the same suit)
5. straight (five cards with consecutive ranks)
6. three-of-a-kind (three cards of the same rank)
7. two pairs
8. pair (two cards of the same rank)
9. high card (any other hand)

We abbreviate ranks and suits as follows (letters may only be lower-case):

Ranks: `2 3 4 5 6 7 8 9 t j q k a`

Suits: `c d h s`

Cards are always **compared by rank first**, and only then by suit. We compare suits using the **'reverse alphabetical order'** ranking: spades (highest), hearts, diamonds, clubs (lowest).

If any number of poker hands are **in the same category**, follow the rules below (simplified, not the standard way):

1. straight flush: The one with the higher ranking top card is better. If they are the same, compare the suit of the top cards. (for straight: `2 3 4 5 6` ~ `t j q k a`)
2. four-of-a-kind: The one with the higher ranking set of four cards is better.
3. full house: The one with the higher ranking set of three cards is better.
4. flush: Same as 1.
5. straight: Same as 1.
6. three-of-a-kind: Same as 3.
7. two pairs: The hand with the highest ranking pair wins. If they have the same rank, compare the highest suit of the highest ranking pair.
8. pair: Same as 7.
9. high card: Same as 1.

Note: not allow the use of jokers, and assume that aces are high.

Input

The input contains four lines. Each line is followed by a newline character. These four lines also represent four hands of five cards for A, B, C, D respectively.

Output

List A, B, C, D in the order from best to worst according to their poker hands. Each letter is followed by a space.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
ts js qs ks as
9s 9h 9d 9c 8s
8h 8d 8c 7s 7h
6s 6h 6d 5s 4s
```

Output

```
A B C D
```

Sample2

Input

```
ts js qh as kh
3d 3s 3h 2h 2s
jc qc kc 7c 8c
2c 3c 4c 5c 6c
```

Output

```
D B C A
```

Sample3

Input

```
ah 3h 4h 9h th
8h 8s jh jd 4c
2s 2d 2c 2h 3s
jc js 5h 5c 3d
```

Output

```
C A D B
```

基礎字串處理 - Basic String Operation

Description

給定一個字串 `str1` 以及數個指令，請根據指令對字串進行處理並執行動作。
可能會出現的指令如下：`PRINT` `REVERSE` `TO_LOWER` `TO_UPPER` `REMOVE` `COMPRESS`。
下方是各個指令的詳細解釋；建議各位使用 function 來執行各個指令。
每個指令的 output 後必須分行！

Given a string of characters`str1`and a list of commands, perform operations according to the commands.
The possible commands are: `PRINT REVERSE TO_LOWER TO_UPPER REMOVE COMPRESS`.
Command descriptions are in the following section. Using functions to treat each commands is recommended, though not required.
Remember to break the line after each command's output!

指令說明 / Command Description

PRINT

- 印出 `str1`當下的狀態。
  Print the current state of the string
- 這個指令不應該對`str1`做出任何改動。
  No modification should occur on `str1`

REVERSE

- 將`str1`頭尾顛倒。
  Reverse`str1`.
- 假設`str1 = "Mine 123! @@"`執行結果應為`str1 = "@@ !321 eniM"`。
  E.g.`str1 = "Mine 123! @@"`Becomes`str1 = "@@ !321 eniM"`.
- 你應該改變 `str1`的內容並**印出來**。
  You should modify the data in`str1`and **print the modified string**.

TO_UPPER

- 將字串內**所有小寫字母**轉成對應的大寫字母。
  Transform **all lower-case alphabets** in the string to upper-case.
- 不需要對非字母的字元做出任何改動。
  Non-alphabet character remains the same.
- 你應該改變 `str1`的內容並**印出來**。
  You should modify the data in `str1`and **print the modified string**.

TO_LOWER

- 將字串內**所有大寫字母**轉成對應的小寫字母。
  Transform **all upper-case alphabets** in the string to lower-case.

- 不需要對非字母的字元做出任何改動。
  Non-alphabet character remains the same.

- 你應該改變 `str1` 的內容並**印出來**。
  You should modify the data in `str1` and **print the modified string**.

REMOVE <target>

- 自`str1`移除所有字元 <target>。所有剩下來的字元必須**向前推移填滿所有空位。**
  From `str1`, **remove all occurrence** of the character <target>. The remaining characters must be **shifted to the front to fill up the blanks**.

- 假設`str1 = ['a', 'p', 'p', 'l', 'e', '\0'], target = 'p'`, 執行結果應為`str1 = ['a', 'l', 'e', '\0']`。有注意到長度變短了嗎？
  e.g., `str1 = ['a', 'p', 'p', 'l', 'e', '\0'], target = 'p'`, the result should be `str1 = ['a', 'l', 'e', '\0']`. Notice the length of the string is shortened.

- **執行完成後至少會留下 1 個字元。**
  There will be at least 1 character left in the string after removal.

- 空白`' '`也可能是被移除的目標。
  Spaces`' '`might also be the target to remove.

- 你應該改變 `str1` 的內容並**印出來**。
  You should modify the data in `str1` and **print the modified string**.

COMPRESS

- 使用 **Run-Length Encoding** 來對`str1`進行壓縮。以下為 Run-Length Encoding 的執行步驟：
  Use **Run-Length Encoding** to compress`str1`. The following is how to Run-Length Encoding:
  - 從`str1[0]`開始。
    Start from `str1[0]`.
  - 計算選到的字元連續出現了幾次。
    Count the number of **contiguous subsequent occurrences** of the picked character.
  - 將其改成`-<字元><出現次數>`的形式。**注意前面有個減號**，用來標示被壓縮的字元。
    Transform into the form as `-<char><occurrence>` . **Note that there is a minus sign in front of the pair**, used to mark which character is getting encoded**.**
  - 假設`str1 = "aaaabbbcccikK"`執行結果應為`str1 = "-a4-b3-c3-i1-k1-K1"`。請注意，**大小寫英文字母被視為不同的字元。**
    e.g., `str1 = "aaaabbbcccikK"` becomes `str1 = "-a4-b3-c3-i1-k1-K1"`. Notice that **lower and upper case alphabets are considered different characters**.

  - 假設`str1 = "AaaaC@@--- """"`執行結果應為`str1 = "-A1-a3-C1-@2--3- 2-"3"`。請注意，**空白和減號也會被處理。**
    e.g.,`str1 = "AaaaC@@--- """"` becomes `str1 = "-A1-a3-C1-@2--3- 2-"3"`. Notice that **spaces and minus signs are also being encoded.**

- 你應該改變 `str1` 的內容並**印出來。**
  You should modify the data in `str1` and **print the modified string**.

提示與建議 / Tips and Suggestions

---

1. 當在 `scanf()` 中使用 `%s` 時，**會取用直到下個 white space character**(例如空白`' '`, 換行`'\n'`, 水平 / 垂直 Tab`'\t'` / `'\v'`等等) **之前的所有字元。而這個字元則會留在 input 中不被讀取。**
   When using `%s` in `scanf()`, **all inputs until the next trailing whitespace character** (e.g., spaces`' '`, newline`'\n'`, horizontal / vertical tabs`'\t'` / `'\v'`) **will be consumed**, **but the whitespace character will remain untouched**.
   \
2. `getchar()` **可以從 input 中移除一個 character**，可參考以下使用例：
   `getchar()` **can be used to remove a character from input**. The following is a simple example:

```
/* 自 input 中移除一個字元並將該字元賦值給 a。 與 scanf("%c", &a) 相同效果。
   Remove 1 character from input and assign the character to a. Has same effect as
scanf("%c", &a). */
char a = getchar();

/* 僅自 input 中移除一個字元。
   Remove 1 character from input. No assignment happens here. */
getchar();
```

3. `==`**無法用來比較字串** 。
   `==`**cannot be used to compare strings.

   **
4. **你必須在字串結束處加上** `'\0'` **來標示字串終結。
   `'\0'`is required and used to mark the end of character array strings.

   **
5. 每個可以成功執行的指令都會讓你拿到一些分數。
   Every correctly implemented commands will grant you some points.

Input

First line is the original string `str1`. ∘ Might contain all Printable ASCII Code Characters, expect DEL. (a.k.a, ACSII Code 32~126) ∘ Through the whole program (including this input), the string will not be longer than

1024 characters. (1025 if terminating '\0' included) The second line contains a integer `N`, denotes how many commands would be executed. ◦ 1 ≤ `N` ≤ 100 The following `N` Lines are the commands.

Output

Print everything the commands requires you to.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
ABC   %%%
10
REVERSE
COMPRESS
REMOVE -
REMOVE
REMOVE 1
REMOVE B
REMOVE C
TO_LOWER
COMPRESS
TO_UPPER
```

Output

```
%%%   CBA
-%3- 3-C1-B1-A1
%3 3C1B1A1
%33C1B1A1
%33CBA
%33CA
%33A
%33a
-%1-32-a1
-%1-32-A1
```

Sample2

Input

```
AAAAA - -  BBCC
1
PRINT
```

Output

```
AAAAA - -  BBCC
```

Sample3

Input

```
AAAAA - -  BBCC
1
REVERSE
```

Output

```
CCBB  - - AAAAA
```

Sample4

Input

```
AAAAA - -  BBCC
1
COMPRESS
```

Output

```
-A5- 1--1- 1--1- 2-B2-C2
```

Sample5

Input

```
aaaaa - -  bbCC
1
TO_UPPER
```

Output

```
AAAAA - -  BBCC
```

Sample6

Input

```
AAAAA - -  BBCC
1
REMOVE B
```

Output

```
AAAAA - -  CC
```

Lec11

FORTUNE LOVER 世界的加法規則

Description

卡塔麗娜今年就要上小學囉~ 在小學的第一節數學課，老師要教大家怎麼做整數加法！

在 FORTUNE LOVER 裡，整數加法的規則是這樣的：

1. 先把被加數和加數當作 float 看待（以 IEEE 754 單精度浮點數的格式表達）。
    - 所以 1 並不是 00000000 00000000 00000000 00000001，而是 00111111 10000000 00000000 00000000。
    - 然後 2 也不是 00000000 00000000 00000000 00000010，而是 01000000 00000000 00000000 00000000。
    - 如果你不知道怎麼做轉換，可以用這個網站來幫你，只要在 Decimal 處輸入一個整數，按下 "Convert to binary"，就可以知道他的 IEEE 754 表示法是什麼囉~

2. 將兩個 32-bits 的二進位數字作加法，得到一個 32-bits 的二進位數字
3. 將該二進位數字當作 int （二補數的有號整數）解讀。
    - 如果該二進位數字是 00000000 00000000 00000000 00001111，那你就要把他當成 15
    - 如果該二進位數字是 11111111 11111111 11111111 11110000，那你就要把他當成 -16
    - Again，如果你不知道怎麼做轉換，可以用這個網站來幫你，只要在 Binary 處輸入32-bits 的二進位數字（不要有空白分隔），按下 "Convert to Decimal"， 就可以知道這個 32-bits 二進位數字對應的整數是多少了~

舉例來說，如果我們想進行 1 + 2 的加法：

1. 把 1 和 2 當作 float，寫出他們的 bit pattern：
    - 1 = 00111111 10000000 00000000 00000000
    - 2 = 01000000 00000000 00000000 00000000
2. 將兩個 32-bits 的二進位數字作加法，得到一個 32-bits 的二進位數字

```
   00111111 10000000 00000000 00000000
+) 01000000 00000000 00000000 00000000
---------------------------------------
   01111111 10000000 00000000 00000000
```

3. 將該二進位數字當作 int 解讀
    - 01111111 10000000 00000000 00000000 = 2139095040

所以， 1 + 2 = 2139095040。

你是卡塔麗娜的數學老師，為了讓卡塔麗娜更理解這個複雜的加法是怎麼進行的，請你在進行每一題範例講解時都把這個相加的過程清楚的寫出來喔~

Input

Two integer a, b. The range of a, b is -999,999 <= a, b <= 999,999.

Output

The process of addition and its result. Please refer to examples.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>

void addition();

void print_binary(void *n){
    for(int i = 31; i >= 0; --i){
        printf("%d", (*(int *)n >> i) & 1);
        if (i && !(i % 8)) printf(" ");
    }
}

int main(){
    addition();
    return 0;
}
```

Sample1

Input

```
1 2
```

Output

```
   00111111 10000000 00000000 00000000
+) 01000000 00000000 00000000 00000000
---------------------------------------
   01111111 10000000 00000000 00000000
1 + 2 = 2139095040
```

Sample2

Input

```
5 -5
```

Output

```
   01000000 10100000 00000000 00000000
+) 11000000 10100000 00000000 00000000
---------------------------------------
   00000001 01000000 00000000 00000000
5 + -5 = 20971520
```

Sample3

Input

```
-4 38
```

Output

```
   11000000 10000000 00000000 00000000
+) 01000010 00011000 00000000 00000000
---------------------------------------
   00000010 10011000 00000000 00000000
-4 + 38 = 43515904
```

Sample4

Input

```
-15942 -18168
```

Output

```
   11000110 01111001 00011000 00000000
+) 11000110 10001101 11110000 00000000
---------------------------------------
   10001101 00000111 00001000 00000000
-15942 + -18168 = -1928919040
```

Kira Kira Random Star

Description

Here comes the POINTER, some stars start to appear in your program.

Let's start from the easiest one !

Please implement this function:

`void star(int* a, int b, int* sum)`

一閃一閃亮晶晶 滿天都是小星星

教到指標後大家的程式編輯器裡也開始充滿了星星~~（相信某些人的頭上也開始冒出星星了）~~

就讓我們從最簡單的部份開始認識這令人心跳不已的功能吧！

請實作以下函式：

`void star(int* a, int b, int* sum)`

https://youtu.be/Y7JoFiRQ0RA

Input

Input a is a pointer to an int, b is an int

Output

Sum of the value a points to and b, stored in the memory space sum points to

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>

void star(int*, int, int*);

int main()
{
    int kirakira, dokidoki, kasumi;
    scanf("%d%d", &kirakira, &dokidoki);

    // kasumi = kirakira + dokidoki
    star(&kirakira, dokidoki, &kasumi);

    printf("%d", kasumi);

    return 0;
}
```

Sample1

Input

```
242 104
```

Output

```
346
```

Sample2

Input

```
693 488
```

Output

```
1181
```

Polynomial Function

Description

Ariel learns pointer today. She wants to implement a polynomial function named 'poly'.

The polynomial function is\

> $x^3 + y^2 + z$

The 'poly' function has three parameters `A`, `B`, `C` and all of them are pointer to integer. `x`, `y`, `z` are integer variables which pointed by `A`,`B`,`C`. **Notice** that they're maybe `NULL`. If the pointer is `NULL`, then set integer variable zero.**(referring to Sample 2)**

Finally, return a pointer from 'poly' function.
Please implement 'poly' function!

Input

The 'poly' function has three parameters A, B, C. All of them are pointer to integer.(They're maybe NULL.)

Output

Return a pointer from 'poly' function.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>

int *poly(int*, int*, int*);

int main() {
    int a, b, c;
    int *A, *B, *C;
    scanf("%d %d %d", &a, &b, &c);
    A = &a;
    B = &b;
    C = &c;
    if (a < 0) A = NULL;
    if (b < 0) B = NULL;
    if (c < 0) C = NULL;

    int *result = poly(A, B, C);
    printf("%d", *result);
    return 0;
}
```

Sample1

Input

```
1 2 3
```

Output

```
8
```

Sample2

Input

```
-1 10 5
```

Output

```
105
```

Score Calculation

Description

小宇是位聰明絕頂的國中生，某日在自學指標後，他決定來炫技算一下自己期中考的成績。

給予國文、英文、數學、自然、社會五科的分數。

計算這五科的總分以及平均。

值得注意的是，小宇為了炫技，因此有時使用指標有時不用，因此請仔細看看 main 函式內的內容為何。

Yu, a smart student, learned pointer on his own.

He decided to use pointer to calculate his midterm's scores.

Given five subjects' scores (Chinese, English, Math, Science, and Social Studies), please calculate the total score and the average score.

Notice that he may or may not use pointer in the code, please see the main function carefully.

Input

給予五科（國英數自社）的分數。 Given five subjects' scores (Chinese, English, Math, Science, and Social Studies).

Output

五科的總分以及平均。 The total score and the average score.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>

double calculate_score(int, int, int *, int, int, int *);

int main(void) {
    int chinese, english, math, science, social_studies;
    scanf("%d %d %d %d %d", &chinese, &english, &math, &science, &social_studies);
    int total = 0;
    double average = calculate_score(chinese, english, &math, science,
social_studies, &total);
    printf("%d %.1lf", total, average);
    return 0;
}
```

Sample1

Input

```
87 78 22 66 69
```

Output

```
322 64.4
```

Sample2

Input

```
99 99 99 99 99
```

Output

```
495 99.0
```

Share Candy

Description

You are a teacher in a kindergarten. Today you bring a bag of candies to the class, and you want to share them with your kids. Kids all LOVE candy! They all want as many as they can have. But for the fairness, you have to give each kid the same number of candies. Given the total number of candies and the number of kids, how many candies can a kid have? And how many candies will remain?

Attention: You HAVE TO implement this logic in a function.

你是一個幼稚園老師，今天你買了一包糖糖想要分給班上的小朋友們。小朋友都愛死糖果了，所以每個人都想要儘量得到最多的糖果，但是為了公平起見，每個人分到的糖果數必須要一樣多。給定糖果的個數與小朋友的人數，請問每個人可以得到幾顆糖果呢？最後你會剩下幾顆糖果呢？

注意：你必須將以上邏輯實作在函數裏面。

Input

The input consists of 2 integers m, n. m is the number of candies, and n is the number of kids. Notice that 0 < n < m < 100000.

Output

Your function need to "return" 2 number q, r. q is the number of candies each kid can have, and r is the number of candies remained.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>

void share_candy(int m, int n, int* q, int* r);

int main(){
    int m, n, candy_each, candy_remain;
    scanf("%d %d", &m, &n);
    share_candy(m, n, &candy_each, &candy_remain);
    printf("%d %d", candy_each, candy_remain);
    return 0;
}

// Your code goes here
```

Sample1

Input

```
114 28
```

Output

```
4 2
```

Sample2

Input

```
500 100
```

Output

```
5 0
```

Swap Again

Description

There are two arrays `A` and `B`. Both of them have the same length `n`. We want to make`A[i]` not less than that of `B[i]` (0 <=`i`<= `n`- 1) by swapping them. Please implement the `swap` function.

Input

The first line contains the length of the array. The second and third lines are the elements of array A and B.

Output

The swapped arrays separated by newline character.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define MAX_LEN 20000

void swap(int *, int *);

int main() {
  int n, A[MAX_LEN], B[MAX_LEN];

  // read input
  scanf("%d", &n);
  for (int i = 0; i < n; ++i)
    scanf("%d", &A[i]);
  for (int i = 0; i < n; ++i)
    scanf("%d", &B[i]);

  // swapping
  for (int i = 0; i < n; ++i)
    swap(&A[i], &B[i]);

  // output
  for (int i = 0; i < n; ++i)
    printf("%d ", A[i]);
  putchar('\n');
  for (int i = 0; i < n; ++i)
```

```
    printf("%d ", B[i]);

  return 0;
}
```

Sample1

Input

```
5
1 8 7 8 5
5 2 5 8 10
```

Output

```
5 8 7 8 10
1 2 5 8 5
```

Sample2

Input

```
10
36 74 47 79 6 44 42 5 98 69
10 97 90 53 39 83 68 22 14 52
```

Output

```
36 97 90 79 39 83 68 22 98 69
10 74 47 53 6 44 42 5 14 52
```

Lec12

Cropped Report Card

Description

考試考完，又到了面對現實的時刻。大雄很緊張，因為他知道肯定會有科目不及格，但是不能給媽媽看到不及格的成績，所以裁切成績單的時刻到了！

裁切成績單的規則是：

1. 裁切後的成績單內只有及格 (大於等於六十分) 的成績
2. 成績數量越多越好，這樣媽媽才不會懷疑成績單是不是被動了手腳
3. 如果數量相等，取成績合最大的區段

舉例來說，現在大雄的成績是 `40 50 60 70 80 50 90`,
那我們想要找到的成績區段就是 `60 70 80`

請你幫助大雄度過難關！

After the exam, it's time to face the reality.
Nobita is nervious because he knows that he must have failed some subjects. He will be in trouble if his mom knows that. It is time to crop the report card!

The rules of croping reocrd card are listed below:

1. The cropped report card should only contain the score higher or equal to 60 points.
2. There should be as many score records as possibile.
3. If the number of score records is the same, choose the section with the highest sum of scores.

For example, Nobita's scores are`40 50 60 70 80 50 90`,
the result should be `60 70 80`.

Please help Nobita overcome this difficulty.

Input

A line contain n integers. The 1st number to (n - 1)th number indicate the score. The nth number is -1 0 <= score <= 100 1 <= n <= 1000

Output

The scores on the report card. Each number must follow by a space.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
int *report_card (int *s) ;

int main () {
    int score[1001];
    int *show;
    int count = 0;
    do {
        scanf("%d", &score[count++]);
    } while (score[count - 1] != -1);

    show = report_card(score);

    count = 0;
    while (*(show + count) != -1) {
        printf("%d ", *(show + count));
        count++;
    }
}
```

Sample1

Input

```
50 60 70 80 90 40 80 -1
```

Output

```
60 70 80 90
```

Sample2

Input

```
40 60 70 50 80 90 -1
```

Output

```
80 90
```

Modifying Array Problem

Description

給定一陣列，利用指標操作，根據給定的模式，將陣列的值做對應的轉換。

規則如下：

一開始給定一個陣列 `arr`，其中的第一個元素（`arr[0]`）代表的是「模式」。

對於接下來的元素（`arr[1]` ~ `arr[SIZE-1]`），我們用一個新的陣列 `newArr` 來講解。

我們讓：

`newArr[0]` = `arr[1]`

`newArr[1]` = `arr[2]`

`newArr[2]` = `arr[3]`

依此類推，`newArr[i]` = `arr[i+1]`。

在 `newArr` 裡，

如果模式為 1，則 index 為 2 的倍數的值將加上 10；

如果模式為 2，則 index 為 3 的倍數的值將乘以 8；

如果模式為 3，則 index 為 5 的倍數的值將減去 2。

最後，將改動後的內容印出來（不包含「模式」的值）。

注意：須根據給定之 `main` 函式完成 `modify_array` 函式的實作，請仔細閱讀 `main` 函式的內容 😉 。

Given an array, please use pointer to modify its elements' values based on the provided mode.

The rule is showed below:

We will give you an array `arr`, the first element (`arr[0]`) means「mode」.

For the other elements (`arr[1]` ~ `arr[SIZE-1]`), we will use a new array `newArr` to introduce.

Let...

`newArr[0]` = `arr[1]`

`newArr[1]` = `arr[2]`

`newArr[2]` = `arr[3]`

and so on, `newArr[i]` = `arr[i+1]`.

In `newArr`,

if the mode is 1, then the value should add 10 if its index is the multiples of 2;

if the mode is 2, then the value should time 8 if its index is the multiples of 3;

if the mode is 3, then the value should minus 2 if its index is the multiples of 5.

Then, print the modified array excluding the mode's value.

Notice: You have to implement `modify_array()` based on the provided `main()`, please read `main()` carefully 😉.

Input

給定一陣列，第一個元素為改動模式的數值。 Given an array, the first element is the mode's value.

Output

改動後陣列的內容（不包含「模式」的值）。 The modified array excluding the mode's value.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define SIZE 201

void modify_array(int *);

int main(void) {
    int arr[SIZE];
    for (int i = 0; i < SIZE; i++) {
        scanf("%d", &arr[i]);
    }
    modify_array(&arr[SIZE/2]);
    for (int i = 1; i < SIZE; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Sample1

Input

```
1 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57
58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84
85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128
129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148
149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168
169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188
189 190 191 192 193 194 195 196 197 198 199 200
```

Output

```
11 2 13 4 15 6 17 8 19 10 21 12 23 14 25 16 27 18 29 20 31 22 33 24 35 26 37 28 39
30 41 32 43 34 45 36 47 38 49 40 51 42 53 44 55 46 57 48 59 50 61 52 63 54 65 56
67 58 69 60 71 62 73 64 75 66 77 68 79 70 81 72 83 74 85 76 87 78 89 80 91 82 93
84 95 86 97 88 99 90 101 92 103 94 105 96 107 98 109 100 111 102 113 104 115 106
117 108 119 110 121 112 123 114 125 116 127 118 129 120 131 122 133 124 135 126
137 128 139 130 141 132 143 134 145 136 147 138 149 140 151 142 153 144 155 146
157 148 159 150 161 152 163 154 165 156 167 158 169 160 171 162 173 164 175 166
177 168 179 170 181 172 183 174 185 176 187 178 189 180 191 182 193 184 195 186
197 188 199 190 201 192 203 194 205 196 207 198 209 200
```

Sample2

Input

```
2 9281 8293 5184 3152 3314 9795 1335 688 4603 966 5017 1925 2200 8093 6444 91 8008
7484 2945 2271 7564 6816 3946 2637 1483 5961 8304 2152 2097 1155 5698 8110 1749
4219 516 5908 8335 4489 3745 5787 2837 4147 9199 1620 2332 8403 6188 8737 862 7394
4311 2979 9528 4049 9284 2849 6504 1330 4447 4841 3844 3778 446 3795 7968 1227
9662 7755 7274 1742 3423 7011 9956 6114 3468 4477 2702 5423 131 8380 3755 1911
6627 985 5090 9956 1259 1350 7166 6546 7981 118 8025 616 2886 4100 1076 9151 1315
5260 4150 8749 5919 2082 992 4428 568 3224 550 1791 1964 7851 7335 49 6577 603
1203 3572 2659 1798 688 1966 8779 4843 9493 2533 9266 2937 1995 3416 8445 6014
1615 7448 195 6125 7055 2248 1979 8849 5580 419 494 6146 2794 886 5114 4408 1750
4211 3094 2491 3775 5255 2690 4130 4502 5136 955 1628 7721 6755 7220 5867 4780
3462 7571 3376 7035 2802 7917 6125 1654 9611 5181 9674 7403 6336 4689 7055 304
6573 2823 4665 268 731 1536 2415 7293 6012 9972 8897 7501 7498 8080 7224 5683 8868
6187 1010
```

Output

```
74248 8293 5184 25216 3314 9795 10680 688 4603 7728 5017 1925 17600 8093 6444 728
8008 7484 23560 2271 7564 54528 3946 2637 11864 5961 8304 17216 2097 1155 45584
8110 1749 33752 516 5908 66680 4489 3745 46296 2837 4147 73592 1620 2332 67224
6188 8737 6896 7394 4311 23832 9528 4049 74272 2849 6504 10640 4447 4841 30752
3778 446 30360 7968 1227 77296 7755 7274 13936 3423 7011 79648 6114 3468 35816
2702 5423 1048 8380 3755 15288 6627 985 40720 9956 1259 10800 7166 6546 63848 118
8025 4928 2886 4100 8608 9151 1315 42080 4150 8749 47352 2082 992 35424 568 3224
4400 1791 1964 62808 7335 49 52616 603 1203 28576 2659 1798 5504 1966 8779 38744
9493 2533 74128 2937 1995 27328 8445 6014 12920 7448 195 49000 7055 2248 15832
8849 5580 3352 494 6146 22352 886 5114 35264 1750 4211 24752 2491 3775 42040 2690
4130 36016 5136 955 13024 7721 6755 57760 5867 4780 27696 7571 3376 56280 2802
7917 49000 1654 9611 41448 9674 7403 50688 4689 7055 2432 6573 2823 37320 268 731
12288 2415 7293 48096 9972 8897 60008 7498 8080 57792 5683 8868 49496 1010
```

Sample3

Input

```
3 9836 514 4670 7917 85 1977 6163 4573 3527 4370 2640 9413 7489 2920 790 8447 2313
6810 9255 1004 2122 7134 562 5469 3621 8177 3510 4972 6966 3210 1571 6877 7964
5556 9975 134 7366 3767 8512 286 2924 9121 1879 436 4611 1722 2689 8424 9331 445
2989 1468 1282 11 514 1945 3425 1509 5482 7889 9195 7106 1411 8479 3344 7794 649
3659 3996 1100 1731 5341 1836 4418 2205 2137 9598 3427 2564 8474 1994 6369 4532 27
7142 483 8429 6117 6454 2178 354 137 6560 655 507 4234 3669 4050 1652 753 4145
6901 4185 80 2955 5908 1049 5940 7420 1315 2584 3052 657 8300 3683 9935 3663 3795
2939 1978 5090 1871 4022 2225 8503 316 2154 5866 896 2173 396 5660 4335 9794 3074
7679 2180 8553 7 4194 929 549 5461 2212 8884 8212 9941 8129 2693 890 1520 1886
1514 534 9309 1105 5760 9952 5720 6228 2955 5504 8461 777 8972 1985 7850 1381 9678
45 3131 1225 2570 8164 7219 6075 1734 9108 9728 6291 1197 6298 3899 2736 3880 9137
3202 1244 8012 5299 6646 5474 2044 9468 9879 8780 4818 8618 6807 2793
```

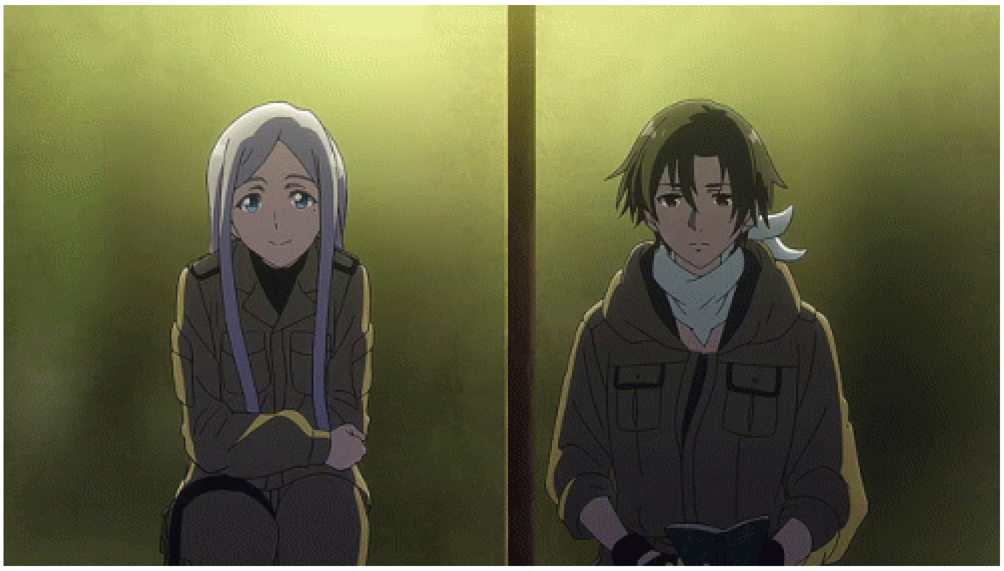
Output

```
9834 514 4670 7917 85 1975 6163 4573 3527 4370 2638 9413 7489 2920 790 8445 2313
6810 9255 1004 2120 7134 562 5469 3621 8175 3510 4972 6966 3210 1569 6877 7964
5556 9975 132 7366 3767 8512 286 2922 9121 1879 436 4611 1720 2689 8424 9331 445
2987 1468 1282 11 514 1943 3425 1509 5482 7889 9193 7106 1411 8479 3344 7792 649
3659 3996 1100 1729 5341 1836 4418 2205 2135 9598 3427 2564 8474 1992 6369 4532 27
7142 481 8429 6117 6454 2178 352 137 6560 655 507 4232 3669 4050 1652 753 4143
6901 4185 80 2955 5906 1049 5940 7420 1315 2582 3052 657 8300 3683 9933 3663 3795
2939 1978 5088 1871 4022 2225 8503 314 2154 5866 896 2173 394 5660 4335 9794 3074
7677 2180 8553 7 4194 927 549 5461 2212 8884 8210 9941 8129 2693 890 1518 1886
1514 534 9309 1103 5760 9952 5720 6228 2953 5504 8461 777 8972 1983 7850 1381 9678
45 3129 1225 2570 8164 7219 6073 1734 9108 9728 6291 1195 6298 3899 2736 3880 9135
3202 1244 8012 5299 6644 5474 2044 9468 9879 8778 4818 8618 6807 2793
```

Shin's Attacking Zone

Description

As the leader of the 86 spearhead squadron, Shin's main job is to guide the squadron members to the right place. He will receive the coordinates of the enemies from the handler during the mission, all he has to do is convert the coordinates into directions and distances that will become the instructions of the squadron members.



Input

First line contains an integer N which indicates the number of the enemies. Next, follows a list of coordinates (x, y) of the enemies, where each coordinate of an enemy occupies a line. 1 <= N <= 1024 -1000 <= x, y <= 1000, where origin is (0, 0)

Output

N lines of the direction DIR and the distance DIST of enemies where the direction comes first followed by the distance and closed by a newline character. 0 <= DIR <360, where 0 represents the positive x axis and increases through counterclockwise. Both DIR and DIST should be rounded to the hundredths.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define MAX_N 1024

void convert_to_polar(int *coords, int N);

int main()
{
    int N, coords[MAX_N + 1][2];
    scanf("%d", &N);

    for (int i = 0; i < N; i++)
        scanf("%d%d", &coords[i][0], &coords[i][1]);


    convert_to_polar((int *) coords, N);

    return 0;
}
```

Sample1

Input

```
4
1 1
-2 2
-3 -3
4 -4
```

Output

```
45.00 1.41
135.00 2.83
225.00 4.24
315.00 5.66
```

Sample2

Input

```
4
3 4
-3 4
-3 -4
3 -4
```

Output

```
53.13 5.00
126.87 5.00
233.13 5.00
306.87 5.00
```

Spread the Color

Description

Read a graph with size `M(5) * N(8)`, for each pixel there are **3 colors(Red, Green, Blue)** that we respectively use `R`, `G`, `B` to represent. If the pixel is **empty**, we use `X` to represent. After reading the graph, read `row` and `col` to get the color of `graph[row][col]`.Now, you need to spread the color starting from the position `(row, col)` and finally print out the result. The spreading rules are below:

Spread color `graph[row][col]` starting from `(row, col)`in 4 directions(Up, Down, Left, Right). You will encounter three situations.\

1. Pixel's color is `X`: Fill the color with `graph[row][col]` and go on.\
2. Pixel's color is the same as `graph[row][col]`: Go on.\
3. Pixel's color is different from `graph[row][col]`: Stop spreading.

Hint: See two-dimensional array as one-dimensional array.

Given `char *p = &graph[0][0]`, you can use `*(p+row*N+col)` to visit the element in `graph[row][col]`.

Input

Input a is a pointer which indicates graph[0][0], row and col are int that indicate the starting position for spreading.

Output

Your function needs to change the colors for each pixel in the graph after spreading.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define M 5
#define N 8
char colors[4] = {'R', 'G', 'B', 'X'};   // Red, Green, Blue, Empty

void spread(char*, int, int);

int main(void) {
    char graph[M][N];
    int row, col;

    for(int i = 0; i < M; i++) {
        for(int j = 0; j < N; j++)
            scanf("%c", &graph[i][j]);
        getchar();      // Ignore '\n'
    }
    scanf("%d %d", &row, &col);    // Starting position

    spread(&graph[0][0], row, col);

    // Print out the spreading result
    for(int i = 0; i < M; i++) {
        for(int j = 0; j < N; j++)
            printf("%c", graph[i][j]);
        printf("\n");
    }
    return 0;
}
```

Sample1

Input

```
XXXXXXXX
XXGXXXXX
XRRXRXBX
XXXXXXXX
XXBXXXXX
2 2
```

Output

```
XXXXXXXX
XXGXXXXX
RRRRRRBX
XXRXXXXX
XXBXXXXX
```

Sample2

Input

```
RXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
0 0
```

Output

```
RRRRRRRR
RXXXXXXX
```

```
RXXXXXXX
RXXXXXXX
RXXXXXXX
```

Sum the values

Description

Given an array's initial address(means arr[0]) which contains 5 elements, please print out the sum of these five elements.

給定一個陣列的起始記憶體位置並且這個陣列包含五個元素，請你輸出陣列所有元素的和

Input

Input arr is a pointer to an array address

Output

Your function need to assign correct value(int type) to variable result.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>

void star(int*, int*);

int main()
{
    int arr[5], result;
    for(int i=0; i<5; i++)
        scanf("%d", &arr[i]);

    star(arr, &result);

    printf("%d", result);

    return 0;
}
```

Sample1

Input

```
1 2 3 4 5
```

Output

```
15
```

Sample2

Input

```
900 904 262 677 562
```

Output

```
3305
```

阿抄的小聰明 - David's Petty Tricks

Description

阿抄身為一個開明且優秀的教師，不會做一些對學生學習沒有益處的事，例如公開成績時他從不將學生的成績排序。——至少表面上是。

今天，阿抄終於忍不住了決定好好的排一下，看看到底學生們到底做的多糟，
然而，阿抄用陣列儲存學生的原始成績資料的資料庫都是公開的，突然排序勢必引起軒然大波，
聰明的他想到：『只要我把記憶體位址拿來排序另外儲存，大家就只會看到奇怪的資料了，我真棒。』

問題是，雖然阿抄能夠寫出檢測用的程式，卻沒辦法寫出這個偷偷排序的函式（對，邏輯一點問題都沒有），請幫幫阿抄完成，維護他的表面吧。

Problem Requirement

請仔細觀察 Loader Code，**設計出可以通過`main()`中各個檢測的函式`sort_pointer()`**。註解中會告訴你各個測試的檢測對象。
Please carefully read the loader code, **design the function`sort_pointer()`so that all tests in`main()`are passed**. Purpose of each tests are written in the comments.\

The "Address of every`arr[]`entries appears in`ptr_arr[]`" Check

- 此檢查用來確定`arr[0]`~`arr[n]`的位址都有出現在`ptr_arr[]`中一次且僅有一次。
  The verifies that the addresses of`arr[0]`~`arr[n]`appears in`ptr_arr[]`once and only once.

- 以 XOR-sum 來執行 checksum 的方式檢查，有興趣可以點開連結閱讀。
  Performing checksum with the XOR-sum method to verify. You can read about it by clicking into the links.\

void sort_pointer**(int **ptr_arr**, int ***arr**, int** size**)**

- **由大到小**排序指向`arr[]`之中各個元素的指標，存入`ptr_arr`之中。
  Sort the values within`arr[]`in **descending order**, and store the pointer of the elements into`ptr_arr`.
- 你**不應該**對`arr`做出任何的改動。
  You **should not** make any change to`arr`.

Judging

1. 本題的評分方法較為特殊，**你不需要自己撰寫任何關於輸出的程式碼**，而是觀察 main function 的動作，補完程式碼並讓程式正確執行。
   This is a custom judge problem. **You don't need to write any program outputting text**, finish the program so that the main function can be executed successfully.
2. `assert()`這個 macro 在**括號中的值為**`false`**時會報錯並直接結束程式**。你需要讓程式完整執行完才能獲得分數。
   When the expression with the parenthesis of the macro`assert()`**evaluates to**`false`**, the program will raise an error and shut-down itself.** You must let the program fully execute in order to get the points.

Trivia

1. `<type> **ptr` 等，稱作 **Pointer-to-Pointer**，而[不是 Double Pointer]{.underline}。

   Things like `<type> **ptr`, are called **Pointer-to-Pointer**, [not Double Pointer.]{.underline}
2. 兩個 pointer 相減，得到的差的型態其實是`ptrdiff_t`而不是`int`，其定義於`<stddef.h>`中。
   The result when subtracting two pointers actually has a type of`ptrdiff_t`, not`int`, defined in`<stddef.h>`.

Input

The 1st line is a integer N, indicating how many entries are in the array. 1 ≤ N ≤ 100 The 2nd line lists the entries of the array, starting from index 0. All numbers appearing in the array are within the range [1, 1000].

Output

The program should successfully execute, Printing a single line of string, "All test passed successfully! 😃"

Loader Code

Your code will be judge using this program:

```c
/* Loader Start */
#include <stdio.h>
#include <stdint.h>
#include <assert.h>

// Function Declairation
void sort_pointers(int **ptr_arr, int *arr, int size);

// Main Function
int main(){
    // Read length of array
    int N;
    scanf("%d", &N);

    // Read entries of array, starting from idx = 0, also store a copy
    int arr[N], backup[N];
    for(int i = 0; i < N; ++i){
        scanf("%d", &arr[i]);
        backup[i] = arr[i];
    }

    // Sort the pointers of arr[] into ptr_arr[]
    int *ptr_arr[N];
    sort_pointers(ptr_arr, arr, N);

    // Check if arr[] is unchanged
    for(int i = 0; i < N; ++i){
        assert(arr[i] == backup[i]);
    }

    // Check if all entry of ptr_arr[] are pointers of arr[] entries
    for(int i = 0; i < N; ++i){
        assert((ptr_arr[i] >= arr) && (ptr_arr[i] < &arr[N]));
    }

    // Check if value pointed by pointers in ptr_arr[] are in descending order
    for(int i = 0; i < N - 1; ++i){
        assert(*ptr_arr[i] >= *ptr_arr[i + 1]);
    }

    // Check if address of every arr[] entries appears in ptr_arr[]
    uintptr_t a = (uintptr_t)arr, b = (uintptr_t)ptr_arr[0];
    for(int i = 1; i < N; ++i){
        a ^= (uintptr_t)&arr[i];
        b ^= (uintptr_t)ptr_arr[i];
    }
    assert(a == b);

    // Success if program runs until here.
    printf("All test passed successfully! :)\n");
    return 0;
}
```

Sample1

Input

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

Output

```
All test passed successfully! :)
```

Sample2

Input

```
5
7 21 5 22 100
```

Output

```
All test passed successfully! :)
```

Lec13

Big Big Numbers Addition

Description

You might know that computer can only represent a limited range of numbers. For a 32-bit machine, it can only deal with signed integers within [-2147483648, 2147483647]. But is there no any other way for computers to deal with a number out of this range, for real?

Recall your memory in elementary school, how do you perform an addition? Everyone might know this way:

```
  1234
+  456
```

```
-------
  1690
```

You take digits at same position, add them together. If the result is bigger than 9, carry 1 into next round. Repeat these step until last digit is processed.

If we use string to represent an integer, and use the addition process above to calculate, digits by digits, then there will be no range problem.

Now, given two VERY LARGE positive integers, can you add them together?

大家應該都知道，電腦能夠表示的數值範圍是非常有限的。一台 32 位元的電腦，最多只能表達在 [-2147483648 - 2147483647] 這個範圍內的有號整數而已。但難道真的沒有任何辦法能讓電腦進行超過這個範圍的整數運算嗎？

回想看看你在小學學算術時，你是怎麼進行加法的？我猜大家都學過直式加法，像是：

```
  1234
+  456
-------
  1690
```

這樣，把在同一個位數的位元相加，如果結果大於 9，就進 1 到下一位數去。重複這個步驟直到最左邊的數字計算完畢。

如果我們能用字串的方式來表示整數，然後用這個方法將他們一位一位相加，那就不會有超出範圍無法計算的問題了。

現在，給定你兩個**超大**的正整數，你能把他們加起來嗎？

Input

The arguments passed to your function will be two strings a and b, representing the VERY LARGE integers. The maximum length of these strings do not exceed 100.

Output

You have to put the result of a + b into argument res. We guarantee that there will be enough space to hold the result of addition.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<string.h>

void add(char a[], char b[], char res[]);
```

```
int main(){
    char a[100], b[100], res[105];
    scanf("%s %s", a, b);
    add(a, b, res);
    printf("%s", res);
    return 0;
}
```

Sample1

Input

```
12345 6789
```

Output

```
19134
```

Sample2

Input

```
12345678900000000 87654321
```

Output

```
12345678987654321
```

Caterpillars Line Up

Description

A caterpillar has a head 'A', a tail 'Z' and a body with `n` lower case letters. (`n`>=0)

Now several caterpillars stand in line but some caterpillars' direction isn't correct. Please help them to line up again!

Input

A string.

Output

A string.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<stdbool.h>

void lineup_again(char line[]);
void swap(char *a, char *b);
void reverse_caterpillar(char *l, char *r);

int main() {
    char line[1000000];
    scanf("%s", line);
    lineup_again(line);
    printf("%s", line);
}
```

Sample1

Input

```
AhuZZrvlAArZ
```

Output

```
AhuZAlvrZArZ
```

Sample2

Input

```
ZgogcngqrAZAZxuuAZigkxA
```

Output

```
ArqgncgogZAZAuuxZAxkgiZ
```

Game Patapon

Description

Patapon 是一款音樂節奏遊戲，你必須跟著節奏輸入指定的動作，才能打倒怪物！

在本題中有三種合法節奏（也就是接下來所謂的「指令」）：

PATA PATA PATA PON：可以前進一步。

PON PON PATA PON：可以原地攻擊怪物，扣除怪獸一滴血。

CHAKA CHAKA PATA PON：可以後退一步並防禦怪物攻擊。

以下有幾個注意事項：

當距離怪物超過 3 時，無論如何攻擊都攻擊不到怪物；但若跟怪物距離為 0 時，會直接被怪物踩爛死掉。

遊戲時可能會打錯遊戲指令，在這題也是，當出現不合法動作（例如：CHAKA CHAKA PON PON），會直接被忽略。

每當做十個動作「後」（無論是否為合法動作），怪物會進行一次攻擊，如果攻擊前的該動作並非防禦（也就是 CHAKA CHAKA PATA PON），會直接被怪物打敗然後死掉。

但若在怪物攻擊前將怪物血量打至零，算成功打敗怪物。

每次最多進行三十次動作，請將每次遊戲進行的結果印出！（參考下方 I/O 說明）

Patapon is a music game, you have to follow the rhythm and enter the commands to beat out the monster!

There are three kinds of valid rhythm (or we say「command」below):

PATA PATA PATA PON: move one step forward

PON PON PATA PON：no move and attack on the monster (one blood)

CHAKA CHAKA PATA PON：move one step backward and defend the monster's attack

Notices:

When the distance between you and the monster is bigger than 3, you can never attack on the monster.

But if the distance is 0, the monster will trample on you, and you will die.

Sometimes there are some wrong commands, for example, CHAKA CHAKA PON PON, those commands will be ignored with no effect.

After entering ten commands (no matter the command is valid or not), the monster will attack on you.

If the command before the monster's attack is not the defence (i.e. CHAKA CHAKA PATA PON), you will be attacked and die.

But, if the monster's blood is 0 before the monster attacks on you, then it means you beat out the monster.

There are at most 30 commands in one round, please print out the result of the game! (See I/O description)

Input

一開始給兩個數字，分別為「與怪獸的距離」和「怪獸的血量」。 接著給予一字串，分別是每個動作指令，並以 . 或 , 隔開，最後會有一換行。 There are two numbers, the first one is「the distance to the monster」, and the second one is「the monster's bloods」Then, given a string that contains commands separated by '.' or ','. Followed by a newline.

Output

若成功打敗怪物，印出 "YES" 以及一空格，最後印出遊戲中打敗怪物前「合法指令」的數量。 反之，若失敗（包含玩家死掉或指令輸完但怪物沒死），印出 "NO" 以及一空格，最後印出遊戲中怪物剩餘的血量。 If you beat out the monster successfully, print out "YES" and one space, then print out the count of「the valid commands」before beating out the monster. On the contrary, if failed (e.g. you died or the commands end before beating out the monster), print out "NO" and one space, then print out the remaining bloods of the monster.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5 4
PATA PATA PATA PON,PON PON PATA PON.PON PON PATA PON,PATA PATA PATA PON.PATA PATA
PATA PON,CHAKA CHAKA PATA PON.PON PON PATA PON,PON PON PATA PON.PON PON PATA
PON,CHAKA CHAKA PATA PON.PON PON PATA PON,PATA PATA PATA PON,CHAKA PATA PATA
PON,PON PATA PATA PON,PON PON PATA PON
```

Output

```
YES 13
```

Sample2

Input

```
5 4
PATA PATA PATA PON.PON PON PATA PON.PON PON PATA PON.PATA PATA PATA PON.PATA PATA
PATA PON.CHAKA CHAKA PATA PON.PON PON PATA PON.PON PON PATA PON.PON PON PATA
PON.PON PON PATA PON.PON PON PATA PON.PATA PATA PATA PON.PON PON PATA PON
```

Output

```
YES 10
```

Sample3

Input

```
5 4
PATA PATA PATA PON,PON PON PATA PON,PON PON PATA PON,PATA PATA PATA PON,PATA PATA
PATA PON,CHAKA CHAKA PATA PON,PATA PATA PATA PON,PON PON PATA PON,PON PON PATA
PON,PON PON PATA PON,PON PON PATA PON,PATA PATA PATA PON,PON PON PATA PON
```

Output

```
NO 1
```

Sample4

Input

```
5 4
PATA PATA PATA PON,PATA PATA PATA PON,PATA PATA PATA PON.PATA PATA PATA PON,PATA
PATA PATA PON.PATA PATA PATA PON,PATA PATA PATA PON.PATA PATA PATA PON
```

Output

```
NO 4
```

Sample5

Input

```
5 4
PATA PATA PATA PON.PATA PATA PATA PON.PON PON PATA PON.CHAKA CHAKA PATA PON.PON
PON PATA PON.PON PON PATA PON.PON PON PATA PON,PON PON PATA PON.PON PON PATA
PON,CHAKA CHAKA PATA PON.PON PON PATA PON,PON PON PATA PON.PON PON PATA PON,PON
PON PATA PON.PON PON PATA PON,PON PON PATA PON.PON PON PATA PON,PON PON PATA
PON,PON PON PATA PON,CHAKA CHAKA PATA PON,PON PON PATA PON,PON PON PATA PON,PON
PON PATA PON,PON PON PATA PON,PON PON PATA PON,PON PON PATA PON,PON PON PATA
PON,PON PON PATA PON,PON PON PATA PON.CHAKA CHAKA PATA PON
```

Output

```
NO 3
```

IEEE Reference Format

Description

大家一定都有寫報告的經驗對吧？寫報告的時候最重要的，就是要把你引用的來源給註記清楚。在資訊系，我們寫報告時很常會引用來自 IEEE 的論文或期刊，IEEE 規範的會議論文引用格式為：

> J. K. Author, "Title of paper," in Abbreviated Name of Conf., (location of conference is optional), (Month and day(s) if provided) year, pp. xxx-xxx.

一個實際範例如下：

> Y. Azar et al., "28 GHz propagation measurements for outdoor cellular communications using steerable beam antennas in New York city," 2013 IEEE International Conference on Communications (ICC), Budapest, 2013, pp. 5143-5147, doi: 10.1109/ICC.2013.6655399.

這裡頭包含了以下元素：

主要作者：J. K. Author, "et al." 代表有複數作者但未全數列出，範例內為 `Y. Azar et al.`
論文標題：Title of paper，範例內為 `28 GHz propagation measurements for outdoor cellular communications using steerable beam antennas in New York city`
會議名稱：Abbreviated Name of Conf.，範例內為`2013 IEEE International Conference on Communications (ICC)`
會議舉辦地點（選填）：(location of conference is optional)，範例內為`Budapest`
會議舉辦時間：(Month and day(s) if provided) year，年份為必填、月份與日期選填。範例內為`2013`
出版物資訊：pp. xxx-xxx，包含頁碼和數位物件辨識碼等，範例內為`pp. 5143-5147, doi: 10.1109/ICC.2013.6655399`

給定以上資訊，請你把所有的資訊組合在一起，組成符合 IEEE 論文引用格式的字串。

Input

The arguments will include the information mentioned above. Include authors, title, conference name, location, date and page information.

Output

Your function have to return a string of information about this paper that match the IEEE reference format.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<string.h>

char *reference(char author[], char title[], char conference[], char location[], char date[], char ppdoi[]);

int main(){
    char author[100], title[150], conference[150], location[30], date[5], ppdoi[60];
    scanf("%[^\n]\n%[^\n]\n%[^\n]\n%[^\n]\n%[^\n]\n%[^\n]", author, title, conference, location, date, ppdoi);
    printf("%s", reference(author, title, conference, location, date, ppdoi));
    return 0;
}
```

Sample1

Input

```
Y. Azar et al.
28 GHz propagation measurements for outdoor cellular communications using steerable beam antennas in New York city
```

```
2013 IEEE International Conference on Communications (ICC)
Budapest
2013
pp. 5143-5147, doi: 10.1109/ICC.2013.6655399
```

Output

```
Y. Azar et al., "28 GHz propagation measurements for outdoor cellular
communications using steerable beam antennas in New York city," 2013 IEEE
International Conference on Communications (ICC), Budapest, 2013, pp. 5143-5147,
doi: 10.1109/ICC.2013.6655399.
```

Sample2

Input

```
K. He, X. Zhang, S. Ren and J. Sun
Deep Residual Learning for Image Recognition
2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
Las Vegas, NV
2016
pp. 770-778, doi: 10.1109/CVPR.2016.90
```

Output

```
K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image
Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition
(CVPR), Las Vegas, NV, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.
```

Number Sum Calculator

Description

Please calculate the sum of the numbers.

Hint: Use `strtok()` and `atoi()`

Input

A string whose length isn't greater than 100. Every number is separated by '+'.

Output

The sum of the numbers.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

#define N 100

int addition(char str[]);

int main() {
    char str[N];
    scanf("%s", str);
    int sum = addition(str);
    printf("%d", sum);
    return 0;
}
```

Sample1

Input

```
1+2+3+4+5
```

Output

```
15
```

Sample2

Input

```
1+22+333
```

Output

```
356
```

Rising Hope

Description

Divination is a traditional way to predict future. Now given a string R as the result of a magician doing some divination, please calculate the "Hope point" from this string.

Hope points starts from 0, you should scan result string from start to end, if the character is P(Positive), means the hope is rising by 1

if the character is N(negative), means the hope point is decreased by 1.

0 <= length of R <= 1000

R may contain every upper case letters.

占卜是一種傳統的預測未來手段， 不論是把龜殼拿去火烤，觀察茶葉渣，請水晶球幫你看透真相（有時候是看透程式碼的bug）或是一群人聚在一起跳奇怪的舞蹈都是占卜手段。今天你得到一個字串 R， 代表占卜的結果，請你以這個字串計算『希望值』(Hope point)

希望值一開始為 0，若占卜結果中包含P(Positive)就會上升1, 包含N(negative)則會下降1

0 <=R 的長度 <= 1000

R 可能包含所有大寫字母

https://youtu.be/IzhMzY5avLI

Input

A string as function input

Output

An integer

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int rising_hope(char *R);

int main(int argc, char *argv[])
{
    char in[1005];
    scanf("%s", in);
    int hope = rising_hope(in);
    printf("%d\n", hope);

    return 0;
}
```

Sample1

Input

```
A
```

Output

```
0
```

Sample2

Input

```
PPAP
```

Output

```
3
```

Sample3

Input

```
PPNPPNP
```

Output

```
3
```

Sample4

Input

```
LISA
```

Output

```
0
```

Sorting Student Name

Description

你還記得 PK 學園轉學生 嗎？

來了一個轉學生，表示班上有很多同學的座號需要被改變，座號的排序規則為：

- 從名字拼音的開始排序，比較接近 `a` 的座號會比較前面

- 大小寫字母沒有分別 (ex. `AMY` 與 `amy` 視作完全相同)

- 如果剛好出現兩個班上學生姓名的前面字母完全相等，則名字較短的座號會比較前面。 (ex. `Sam` 與 `Samuel`, `Sam` 的座號會比較小)

請你 **按照座號**，印出所有班上學生的名稱！

Do you remember PK high school transfer student?

The sorting rules of student number are listed below:

- Sorted by the letter of name, if the letter is closer to `A`, the student number is smaller. (For instance, the student number of `Chloe` is smaller than that of `Sam` because compare with `C` and `S`,`C` is closer to `A`)

- It is no different between lowercase and uppercase (ex. `AMY` and `amy` are the same)

- In this case, `Sam` and `Samuel`, the letters of `Sam` is the same as the first three letters of `Samuel`, but the length of `Sam` is shorter than that of `Samuel`, `Sam`'s student number is smaller than Samuel.

Please base on the student number, print the name of all students in this class!

Input

First line is a number (N) represents for how many students are there in the class. Following strings are the students' names. 2 <= N <= 100 2 <= length of students' names <= 100

Output

Sorted students' names.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
3
Saiko
Nendou
saiki
```

Output

```
Nendou
saiki
Saiko
```

Sample2

Input

```
6
pat
patsy
patrick
percy
philemen
penelope
```

Output

```
pat
patrick
patsy
penelope
percy
philemen
```

String Insertion

Description

Given a paragraph P and 2 strings s, t, find all string s in paragraphP and insert a string t after string s.

Note: The size of the modified paragraph P is shorter than 100000 and the size of both string s and string t are less than 100.

Input

The first part is the paragraph P itself. It may contain multiple lines. There is a line "---" after the paragraph P. The last two lines are string s and string t. Both string s and string t are followed by a newline character, which is not belong to string s and string t.

Output

The modified paragraph.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <string.h>
#define MAX_LEN_P 100000
#define MAX_LEN_S 100

void strins(char *P, char *s, char *t);

int main() {
  char P[MAX_LEN_P + 1], s[MAX_LEN_S + 2], t[MAX_LEN_S + 2];
  size_t P_len = 0;
  for (char buf[MAX_LEN_P + 1] = ""; strcmp(buf, "---\n"); fgets(buf, MAX_LEN_P +
1, stdin)) {
    size_t len = strlen(buf);
    if (len + P_len > MAX_LEN_P) {
      fprintf(stderr, "The length of the paragraph exceeds %d\n", MAX_LEN_P);
      return 1;
    }
    strncat(P, buf, len + 1);
    P_len += len;
  }
  fgets(s, MAX_LEN_S + 2, stdin);
  s[strlen(s) - 1] = '\0';  // remove newline
  fgets(t, MAX_LEN_S + 2, stdin);
  t[strlen(t) - 1] = '\0';  // remove newline
  strins(P, s, t);
  printf("%s", P);
```

```
    return 0;
}
```

Sample1

Input

```
abc
def
abcdef
abc def
---
abc
d
```

Output

```
abcd
def
abcddef
abcd def
```

Sample2

Input

```
hello world
---
o
o
```

Output

```
helloo woorld
```

Lec16

LEVEL5 -judgelight-

Description

Given an array of `struct ESPer` which contains `level` and `name`,

please sort this array in ascending order by level.

hint: You can use C standard library's `qsort` function:

學園都市是個進行超能力開發的機關，會定期對學生進行能力檢測

他們使用的資料結構包含兩個欄位：`level` 和 `name`

請你寫一個 function 來對包含這個 struct 的陣列做排序

hint: 你可以使用內建的 `qsort`

https://youtu.be/EGmLt7mYSo4

Input

An array of "struct ESPer", array length

Output

sort the array in ascending order by level.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>

struct ESPer {
    char name[64];
    int level;
};

int cmp(const void *a, const void *b);
void sort_level(struct ESPer *arr, int length);

int main()
{
    int n;
    struct ESPer tokiwadai[100];

    scanf("%d", &n);
    for(int i = 0; i < n; ++i) {
        scanf("%d %s", &(tokiwadai[i].level), tokiwadai[i].name);
    }

    sort_level(tokiwadai, n);
    for(int i = 0;i < n;++i) {
        printf("%d %s\n", tokiwadai[i].level, tokiwadai[i].name);
    }

    return 0;
}
```

Sample1

Input

```
5
6 NanjoYoshino
5 MisakaMikoto
0 SatenRuiko
1 UiharuKazari
4 ShiraiKuroko
```

Output

```
0 SatenRuiko
1 UiharuKazari
4 ShiraiKuroko
5 MisakaMikoto
6 NanjoYoshino
```

Look Up the Word in the Dictionary

Description

Jack wants to search a word. Please help him to find which page he should turn to.

Hint: use `strcmp()`

Input

There are 10 records in a dictionary and a target word.

Output

Return the target page. If the target word is not in the dictionary, return 0.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<string.h>
#define N 10

typedef struct dict_t_struct {
    char word[10];
    int page;
} dict_t;

int search(dict_t arr[], char target[]);

int main() {
    dict_t dictionary[N];
    char targetWord[10];

    for(int i=0; i<N; i++)
        scanf("%s %d\n", dictionary[i].word, &dictionary[i].page);

    scanf("%s", targetWord);
    int targetPage = search(dictionary, targetWord);
    printf("%d", targetPage);
    return 0;
}
```

Sample1

Input

```
apple 2
banana 5
cow 7
dog 9
element 13
frog 22
go 44
horse 51
in 52
jet 100
element
```

Output

```
13
```

Sample2

Input

```
apple 2
banana 5
cap 6
dog 9
element 13
frog 22
go 44
horse 51
in 52
jet 100
cow
```

Output

```
0
```

Matrix Addition and Multiplication

Description

Write a program that implements matrix addition and multiplication of two matrices.

Input

The input contains three parts. The first and second part represent the information of the first matrix and the second matrix respectively. The first line of each part contains row and column number of the matrix, and the following line is an array of integers which represent the matrix itself. The second part is a character which represents the operation of the matrix.

Output

The content of the result matrix. Each element is followed by a space character.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define SIZE 50

struct mat {
    int row;
    int col;
    int value[SIZE][SIZE];
};

void scan_mat(struct mat *);
void print_mat(const struct mat *);
void add_mat(const struct mat *, const struct mat *, struct mat *);
void mul_mat(const struct mat *, const struct mat *, struct mat *);

int main(void) {
    struct mat m1, m2, result;
    char op;
    scan_mat(&m1);
    scanf(" %c", &op);
    scan_mat(&m2);
    switch (op) {
        case '+':
            add_mat(&m1, &m2, &result);
            break;
        case '*':
            mul_mat(&m1, &m2, &result);
            break;
    }
    print_mat(&result);
    return 0;
}

void scan_mat(struct mat *m_p) {
    scanf("%d %d", &m_p->row, &m_p->col);
    for (int i = 0; i < m_p->row; ++i) {
        for (int j = 0; j < m_p->col; ++j) {
            scanf("%d", &m_p->value[i][j]);
        }
    }
}

void print_mat(const struct mat *m_p){
    for (int i = 0; i < m_p->row; ++i) {
        for (int j = 0; j < m_p->col; ++j) {
            printf("%d ", m_p->value[i][j]);
            if (j == m_p->col - 1) {
                printf("\n");
            }
        }
    }
}
```

Sample1

Input

```
2 3
3 2 6
2 4 5
+
2 3
4 5 6
3 1 4
```

Output

```
7 7 12
5 5 9
```

Sample2

Input

```
2 2
1 2
3 4
*
2 2
5 6
7 8
```

Output

```
19 22
43 50
```

Notice Struct Pointer

Description

Given two struct Nodes, please multiply their `num`.

Be careful these two Node's type, one is a struct variable and another is a struct pointer.

Input

Two struct Nodes, long long int ans and every variable num < 10000

Output

Multiply these two node's num

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>

struct Node {
    int num;
};
void mul(struct Node *, struct Node, long long int *);

int main(void) {
    struct Node node_a, node_b;
    long long int ans = 1;
    scanf("%d", &node_a.num);
    scanf("%d", &node_b.num);
    mul(&node_a, node_b, &ans);
    printf("%lld", ans);
    return 0;
}
```

Sample1

Input

```
8687 363
```

Output

```
3153381
```

Sample2

Input

```
3741 3313
```

Output

```
12393933
```

Sei's online potion store

Description

Sei is a researcher at the Medicinal Flora Research Institute. She has been work there for a year and is struggling to improve her KPI. One day she finds out that she can sell potions online to improve her poor KPI (?. After several days of hard working, she has finished a framework of the ordering system, there are only two little functions that aims to calculate the total cost of orders and reordering the orders gaps her from finish, please help her to fulfill those functions.

She currently sells 15 potion products over the online shop that are given name `Prod_A, Prod_B ...,` `Prod_O`, each product costs differently, the information could be found in the finished part (loader code) of the platform. You do not need to concern the orders with the same pick up time.

Input

The first line contains an integer N that indicates the number of orders which followed by the detail of N orders. For each order, the order ID comes first, then the pick up time of the order which followed by the item count M of the order, finally M items and the count are given.

Output

The order ID and the total cost of the orders, the order that has a closer pick up time comes first.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

typedef struct {
    unsigned char hour;
    unsigned char minute;
} hm_t;

typedef struct {
    const char *item_name;
    unsigned cost;
    unsigned cnt;
} item_t;

typedef struct {
    unsigned order_id;
    hm_t pick_up_time;
    unsigned total_cost;
    unsigned item_cnt;
    item_t *items;
} order_t;

const item_t avaliable_items[] = {
    {"Prod_A",  10},
    {"Prod_B",  20},
    {"Prod_C",  15},
    {"Prod_D",  30},
    {"Prod_E",  50},
    {"Prod_F",  60},
    {"Prod_G",  75},
    {"Prod_H",   5},
    {"Prod_I",  25},
    {"Prod_J",  80},
    {"Prod_K", 100},
    {"Prod_L",  90},
    {"Prod_M",  45},
    {"Prod_N",  40},
    {"Prod_O",  85},
};

const unsigned avaliable_item_cnt = 15;
#define MAX_ORDER_CNT 1024

unsigned fetch_orders(order_t orders[]);
void sum_total_costs(order_t orders[], unsigned order_cnt);
void reorder(order_t orders[], unsigned order_cnt);
void print_result(order_t orders[], unsigned order_cnt);
void clean_up(order_t orders[], unsigned order_cnt);

int main()
{
    order_t orders[MAX_ORDER_CNT];
    unsigned order_cnt = fetch_orders(orders);
    sum_total_costs(orders, order_cnt);
    reorder(orders, order_cnt);
    print_result(orders, order_cnt);
    clean_up(orders, order_cnt);
    return 0;
}

unsigned fetch_orders(order_t orders[])
{
    unsigned order_cnt;
    scanf("%u", &order_cnt);

    // Prepare the orders.
    order_t *curr = orders;
    for (unsigned i = 0; i < order_cnt; i++, curr++) {
        hm_t *curr_pu = &(curr->pick_up_time);
        scanf("%u %hhu:%hhu %u",
                                &(curr->order_id),
                                &(curr_pu->hour),
                                &(curr_pu->minute),
                                &(curr->item_cnt));

        // Create a variable length array with malloc.
        item_t *curr_items = curr->items = malloc(sizeof(item_t) * curr-
>item_cnt);
        for (unsigned j = 0; j < curr->item_cnt; j++) {
            char item_name[32];
            scanf("%s", item_name);
            for (unsigned k = 0; k < avaliable_item_cnt; k++) {
                if (!strcmp(item_name, avaliable_items[k].item_name)) {
                    // Copy the memory content from avaliable_items to curr_items.
                    curr_items[j] = avaliable_items[k];
                    break;
                }
            }
            scanf("%u", &(curr_items[j].cnt));
        }
    }
    return order_cnt;
}

void print_result(order_t orders[], unsigned order_cnt)
{
    for (unsigned i = 0; i < order_cnt; i++) {
        printf("#%u %u\n", orders[i].order_id, orders[i].total_cost);
    }
}

void clean_up(order_t orders[], unsigned order_cnt)
{
    for (unsigned i = 0; i < order_cnt; i++) {
        free(orders[i].items);
    }
}
```

Sample1

Input

```
3
123 1:40 2
Prod_A 2
Prod_B 3
234 2:10 3
Prod_D 12
Prod_E 1
Prod_C 4
345 1:30 4
Prod_A 3
Prod_B 5
Prod_C 1
Prod_D 2
```

Output

```
#345 205
#123 80
#234 470
```

Sample2

Input

```
5
100 3:10 1
Prod_O 1
101 1:00 3
Prod_K 2
Prod_L 5
Prod_A 20
102 0:40 2
Prod_C 4
Prod_I 2
103 5:45 1
Prod_E 2
104 6:00 4
Prod_F 1
Prod_N 1
```

```
Prod_L 1
Prod_A 1
```

Output

```
#102 110
#101 850
#100 85
#103 100
#104 200
```

Sort Snacks

Description

In lab, students buy snacks once a month. They want to choose snacks with good value for money. (物超所值)

The `value` means `weight/price`.

Please help them sort snacks in descending order based on snacks' `value`.
If the `value` of snacks are equal, sort snacks in ascending order based on snacks' `price`.
If the `price` of snacks are equal, sort snacks in ascending order based on snacks' `id`.

You can use C standard library's `qsort` function to sort snacks:

`void qsort (void* base, size_t num, size_t size, int (*compar)(const void*,const void*));`

The reference link: https://www.cplusplus.com/reference/cstdlib/qsort/?kw=qsort

Input

n kinds of snacks.

Output

Output the id and value of n sorted snacks.

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<stdlib.h>

struct snack
{
    int id;
    int price;
    int weight;
    double value;
};

int cmp(const void *a, const void *b);
void sort_snacks(struct snack snacks[], int n);

int main() {
    int n;
    struct snack snacks[100];
    scanf("%d", &n);
    for(int i=0; i<n; i++) {
        snacks[i].id = i+1;
        scanf("%d %d", &snacks[i].price, &snacks[i].weight);
    }
    sort_snacks(snacks, n);
    for(int i=0; i<n; i++) {
        printf("%d %.2f\n", snacks[i].id, snacks[i].value);
    }
    return 0;
}
```

Sample1

Input

```
5
1 2
4 6
4 10
2 3
12 18
```

Output

```
3 2.50
1 2.00
4 1.50
2 1.50
5 1.50
```

Sample2

Input

```
7
8 2
44 10
7 2
22 5
6 9
10 4
4 1
```

Output

```
5 1.50
6 0.40
3 0.29
7 0.25
1 0.25
4 0.23
2 0.23
```

Struct Array

Description

Given a struct array (named Node) and every Node has variable `a` (int type).

Node array has 5 Node elements, please multiply every Node's `a` variable.

Input

struct Node array (contain five nodes), long long int ans and each variable a < 30.

Output

Multiply every Node's a variable

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#define SIZE 5

struct Node {
    int a;
};
```

```
void mul(struct Node *, long long int *);

int main(void) {
    struct Node node_array[SIZE];
    for(int i=0; i<SIZE; i++){
        scanf("%d", &node_array[i].a);
    }
    long long int ans = 1;
    mul(node_array, &ans);
    printf("%lld", ans);
    return 0;
}
```

Sample1

Input

```
25 21 17 8 15
```

Output

```
1071000
```

Sample2

Input

```
11 2 28 24 3
```

Output

```
44352
```

Weighted Score

Description

期末快到了，該到結算成績的時候，請你幫助教算出所有學生們的期末成績以及最高分章節成績與最低分章節。
在這裡有三個 struct:

- `weight` : 每個章節會有不同的比重，**全部的比重加起來為 100**
- `score` : 每個章節加權前的分數
- `report_card` :
    - `final` : 加權後的分數 (無條件捨去到整數位)
    - `max_score` : 最高分章節成績
    - `min_score` : 最低分章節成績

Please help TAs to calculate student's final score, maximum score and minimum score.

There are three structs:

- `weight` : Each chapter contain different weight, **sum of them is 100**.
- `score` : The score of each chapter
- `report_card` :
    - `final` : Score after weighting
    - `max_score` : Maximum score
    - `min_score` : Minimum score

Input

First line contain two numbers: chapter_num (n), student_num (m) 1 <= chapter_num (n) <= 10 1 <= student_num (m) <= 100 Each of following n lines contain one number and a string. The number means the weight of chapter. The string means the name of chapter. Followings are the student information. First line is student name. Next n lines are student score and chapter. 0 <= student score <= 100

Output

Student name, final score, maximum score and minimum score

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>

struct weight {
    char chapter[10];
    int percent;
};

struct score {
    char chapter[10];
    int score;
};

struct report_card {
    char name[32];
    struct score record[10];
    int final;
    int max_score;
    int min_score;
};

void calculate (struct report_card[], struct weight[], int student_num, int
chapter_num);

int main () {
    int student_num, chapter_num;
    struct weight chapter[10];
    struct report_card card[100];

    scanf("%d %d\n", &chapter_num, &student_num);
    for (int i = 0; i < chapter_num; ++i) {
        scanf("%d %s", &chapter[i].percent, chapter[i].chapter);
    }

    for (int i = 0; i < student_num; ++i) {
        scanf("%s", card[i].name);
        for (int j = 0; j < chapter_num; ++j) {
            scanf("%d %s", &card[i].record[j].score, card[i].record[j].chapter);
        }
    }

    calculate(card, chapter, student_num, chapter_num);

    for (int i = 0; i < student_num; ++i) {
        printf("%s %d %d %d\n", card[i].name, card[i].final, card[i].max_score,
card[i].min_score);
    }
}
```

Sample1

Input

```
3 3
30 array
30 loop
40 pointer
amy
80 array
```

```
70 loop
100 pointer
sam
80 pointer
80 array
80 loop
alice
60 loop
50 pointer
60 array
```

Output

```
amy 85 100 70
sam 80 80 80
alice 56 60 50
```

Sample2

Input

```
2 4
10 union
90 struct
lili
80 union
80 struct
christopher
80 union
80 struct
pata
0 struct
100 union
pon
100 struct
0 union
```

Output

```
lili 80 80 80
christopher 80 80 80
pata 10 100 0
pon 90 100 0
```

windowlocation

Description

在 Chrome 瀏覽器內按下 F12，會打開瀏覽器的開發者工具。在 console 當中輸入 `window.location`，會看到一個叫作 `Location` 的結構跑出來，上頭記載了你當前所在的網址所包含的資訊，包含：

- 通訊協定 (protocol)
- 主機名稱 (host)
- 通訊埠 (port)
- 所在路徑 (pathname)
- 查詢字串/Query String (search)
- 識別符號 (hash)

那麼，這些不同的部份是怎麼被切出來的呢？其實網址是有固定格式的，就跟你家的門牌號碼一樣。網址的格式為：

`[通訊協定]://[主機名稱]:[通訊埠]/[所在路徑]?[查詢字串]#[識別符號]`

舉例，`https://www.abc.com/user/12345/photo?from=20180101&to=20201231#favorite`可以被切成：

- 通訊協定 = `https`
- 主機名稱 = `www.abc.com`
- 通訊埠 = 預設不顯示（但因為使用的是 https，所以可知是 443）
- 所在路徑 = `/user/12345/photo`
- 查詢字串 = `?from=20180101&to=20201231`
- 識別符號 = `#favorite`

但也並非所有部份都位同時存在於一個網址中，像是本頁面的網址就沒有查詢字串的部份。

現在，給你一些網址，請你幫忙找出上述的片段，並裝入 `Location` 結構當中。

Input

An valid url. Must includes protocol and host part, but port, pathname, search and hash are not necessarily existing.

Output

You should put the information parsed from url into the struct passed by argument. For information not existing in url, if it's an integer, set to 0; if it's a string, make it an empty string. Hint: you should prepare memory storage yourself.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct
{
    char *protocol;
    char *host;
    char *pathname;
    char *search;
    char *hash;
    int port;
} Location;

Location *parse_url(char *url);

int main()
{
    char url[500] = "";
    fgets(url, 500, stdin); // Get url string
    Location *l = parse_url(url);
    printf("Location {\n  protocol: %s,\n  host: %s,\n", l->protocol, l->host);
    if (l->port)
        printf("  port: %d,\n", l->port);
    else
        printf("  port: (default),\n");
    printf("  pathname: /%s,\n  search: ?%s,\n  hash: #%s,\n}\n", l->pathname, l-
>search, l->hash);
    return 0;
}
```

Sample1

Input

```
https://www.abc.com:80/user/12345/photo?from=20180101&to=20201231#favorite
```

Output

```
Location {
  protocol: https,
  host: www.abc.com,
  port: 80,
  pathname: /user/12345/photo,
  search: ?from=20180101&to=20201231,
  hash: #favorite,
}
```

Sample2

Input

```
http://1.2.3.4:443/it/is/a/very/long/path/name
```

Output

```
Location {
  protocol: http,
  host: 1.2.3.4,
  port: 443,
  pathname: /it/is/a/very/long/path/name,
  search: ?,
  hash: #,
}
```

Sample3

Input

```
ip://193.55.28.30
```

Output

```
Location {
  protocol: ip,
  host: 193.55.28.30,
  port: (default),
  pathname: /,
  search: ?,
  hash: #,
}
```

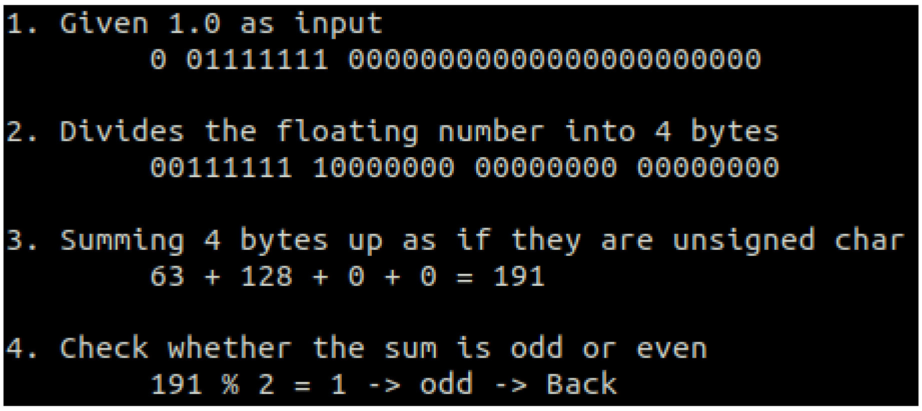
Lec16-2

The Sly TA and His Sly Gun

Description

Chou is a secret agent of his country. One day, he is assigned a job to find the stolen treasure of his country. To help him fulfill the job, his boss gave him a weapon called SlyGun, as the name indicated, there are two possible ways, front and back, for the gun to fire, and the direction of the next fire will be decided by following process:

1. SlyGun will be assigned a floating point number
2. The given floating number will be divided into 4 bytes
3. Treat the 4 bytes as unsigned characters and sum them up into an unsigned character
4. The SlyGun will fire forward if the sum of 4 bytes is even, backward, otherwise



```
1. Given 1.0 as input
       0 01111111 00000000000000000000000

2. Divides the floating number into 4 bytes
       00111111 10000000 00000000 00000000

3. Summing 4 bytes up as if they are unsigned char
       63 + 128 + 0 + 0 = 191

4. Check whether the sum is odd or even
       191 % 2 = 1 -> odd -> Back
```

Input

First line contains an integer N indicates the number of fires following are N floating point numbers Fi each occupies a line 0 < N <= 1000 0.0 <= Fi <= 1000.0

Output

"Front" and "Back" for each fire, each occupies a line

Loader Code

Your code will be judge using this program:

Sample1

Input

```
4
925.887024
768.018005
345.265015
551.627014
```

Output

```
Front
Front
Front
Front
```

Sample2

Input

```
10
344.606995
943.013000
276.812988
944.718994
916.382996
336.312988
115.228996
748.950012
996.010010
682.520020
```

Output

```
Front
Front
```

```
Back
Front
Front
Back
Front
Front
Back
Back
```

Lec17

HAPPY PARTY TRAIN

Description

One tourist train is departing, and it's powered by a steam locomotive.

Unfortunately, when passengers holding a party in cars,

the coal used as fuel in the back of the locomotive is on fire, burning red.

And every car contains different number of flammable items.

When passengers see the fire coming, they try to move these items to prevent fire burn to their car.

If number of peoples in the car >= amounts of flammable items, fire will not burn to this car and the following cars.

Otherwise, passengers in this car will evacuate to the the next car.

Now you know the number of people & flammable items in each car,

Please calculate how many cars will be burnt by fire.

This train is represented with linked-list, each `Car` node's `next` will point to the next car, last cat will points to `NULL`

```
   Head car              Car 1              Car 2                Car N
+------------+  --> +------------+  --> +------------+        +------------+
|     X      |  |   | passenger  |  |   | passenger  |        | passenger  |
|------------|  |   |------------|  |   |------------|        |------------|
|     X      |  |   | flammable  |  |   | flammable  | ...... | flammable  |
|------------|  |   |------------|  |   |------------|        |------------|
|    next    |---   |    next    |---   |    next    |        | next = NULL|
+------------+      +------------+      +------------+        +------------+
```

一列鐵道觀光專車在春天來臨的花香中，乘著無盡想像發車了

為了為乘客們帶來種種回憶，主辦單位特別使用蒸氣火車頭牽引這列專車

很不幸的，乘客們正在車廂中開派對時，車頭後作為燃料的煤炭起火了，正發出熊熊火光

而後方每節車廂中都有數量不等的易燃物

當大家發現火災發生時，便試著將易燃物搬開來避免火勢延燒

若該節車廂人數 >= 易燃物數量時，火勢便會停止蔓延，不會延燒到該節和後續的車廂

但若人數不足，則車廂中所有人都會移往下一節車廂逃生

直到延燒停止或所有乘客無處可逃並跳車為止。

你已知的有每節車廂中的人數及易燃物數量

請計算火勢總共會延燒幾節車廂？

這列火車以 linked-list 形式紀錄，每個 `Car` 節點的 `next` 會指向下一節車廂，最後一節車廂則指向 `NULL`

```
   Head car                 Car 1                  Car 2                    Car N
+------------+   --> +------------+   --> +------------+          +------------+
|     X      |   |   | passenger  |   |   | passenger  |          | passenger  |
|------------|   |   |------------|   |   |------------|          |------------|
|     X      |   |   | flammable  |   |   | flammable  |  ......  | flammable  |
|------------|   |   |------------|   |   |------------|          |------------|
|    next    |---    |    next    |---    |    next    |          | next = NULL|
+------------+       +------------+       +------------+          +------------+
```

https://youtu.be/eVwdeIDjXeM

Input

Head node of the linked-list representing the train.

Output

Return one integer how many car is on fire.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>

struct Car {
    int passenger;
    int flammable;
    struct Car *next;
};

void attachCar(struct Car *head, int passenger, int flammable) {
    struct Car *curr = head;
    while(curr->next) curr = curr->next;             // Find the end of the train

    struct Car *newcar = malloc(sizeof(struct Car)); // Make a new car
    newcar->passenger = passenger;                   // Let passengers in
    newcar->flammable = flammable;                    // Load flammable items
    newcar->next = NULL;                              // This is the last car

    curr->next = newcar;                              // Attach new car to the
train
}

int fire(struct Car *head);

int main(int argc, char *argv[])
{
    int cars;
    int human[13], moeru[13];

    struct Car head;
    head.next = NULL; // passenger & flammable for head in useless

    scanf("%d", &cars);
    for (int i = 0; i < cars; ++i) {
        scanf("%d", &human[i]);
    }
    for (int i = 0; i < cars; ++i) {
        scanf("%d", &moeru[i]);
    }

    for (int i = 0; i < cars; ++i) {
        attachCar(&head, human[i], moeru[i]);
    }

    printf("%d", fire(&head)); // The train is on fire now

    return 0;
}
```

Sample1

Input

```
9
2 1 4 7 4 8 3 6 4
5 9 8 6 1 3 5 2 4
```

Output

```
3
```

Sample2

Input

```
4
17 17 11 8
23 35 36 37
```

Output

```
2
```

Returns

Description

Sometimes you want to return more than 1 values, you can use pointers to do this.

Please allocate an array of int with length `n` using `malloc`, and fill it with `index^2`

使用指標可以傳遞多個數值到 function 外，請使用`malloc`分配長度為`n`的整數陣列，並在其中每一格填入`index^2`

https://youtu.be/KGR5WZV8gT

Input

int n

Output

A pointer to an array of int

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>

int *Returns(int n);

int main(int argc, char *argv[])
{
    int in, *out;
    scanf("%d", &in);
    out = Returns(in);
    for(int i = 0; i < in; ++i) {
        printf("out[%d] = %d\n", i, out[i]);
    }
    free(out);
    return 0;
}
```

Sample1

Input

```
4
```

Output

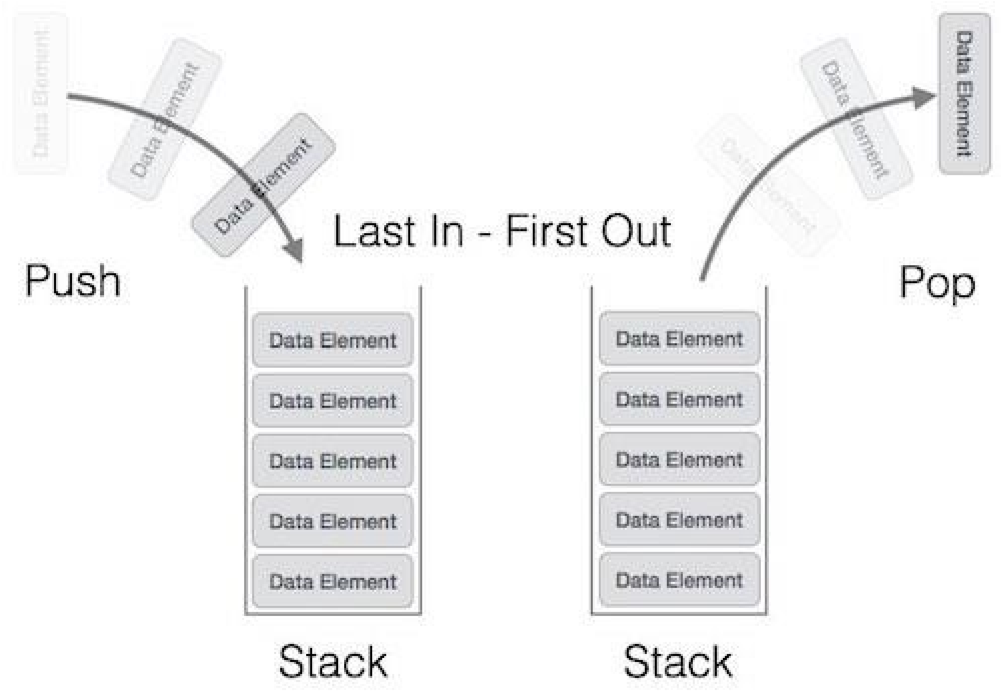
```
out[0] = 0
out[1] = 1
out[2] = 4
out[3] = 9
```

Stack

Description

Stack is an abstract data type that serves as a collection of elements, with two main principal operations:

1. Push: which adds an element to the top of the stack
2. Pop: which removes an element from the top of the stack if the stack is not empty



Please implement push and pop operations of the stack using linked list.

Reference: https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm

Input

The input contains two parts. The first part is the number of the operations. The second part contains the content of the operations, which is push or pop an integer to/from the stack.

Output

The output contains two lines. The first line is the size of the stack, and the second line consists of elements' data and each data is followed by a space. You should output them in reverse chronological order.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct element {
    int data;
    struct element *next;
};

void print_stack(struct element *top);
void push(struct element **top_p, int data);
struct element *pop(struct element **top_p);

int main(void) {
    struct element *top = NULL;
    int num, data;
    char command[5];
    scanf("%d", &num);
    while (num--) {
        scanf("%4s", command);
        if (strcmp(command, "push") == 0) {
            scanf("%d", &data);
            push(&top, data);
        }
        else if (strcmp(command, "pop") == 0) {
            struct element *e = pop(&top);
            free(e);
        }
    }
    print_stack(top);
    while (top) {
        struct element *ptr = top;
        top = top->next;
        free(ptr);
    }
    return 0;
}

void print_stack(struct element *top) {
    int size = 0;
    for (struct element *ptr = top; ptr != NULL; ptr = ptr->next) {
        size++;
    }
    printf("%d\n", size);
    for (struct element *ptr = top; ptr != NULL; ptr = ptr->next) {
        printf("%d ", ptr->data);
    }
}
```

Sample1

Input

```
5
push 1
push 2
push 3
pop
pop
```

Output

```
1
1
```

Sample2

Input

```
5
push 1
pop
push 2
push 3
pop
```

Output

```
1
2
```

Throwing Cards Away

Description

Given an ordered deck of n cards numbered 1 to n with card 1 at the top and card n at the bottom, throw away the top card and move the next card that is on the top of the deck to the bottom of the deck. Repeat this process m times and find the sequence of discarded cards.

給一副排組，由上而下的編號為 1~n，共會進行 m 次抽取，每次抽走最上層的牌並輸出此牌的編號，再將最上層的牌放到牌組的最下方。

Input

Two integers n & m.

Output

The sequence of the discarded cards. Every number is followed by one space.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
7 7
```

Output

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

Sample2

Input

```
9 7
```

Output

```
1 3 5 7 9 4 8
```

法陣模擬器 - The Magic Circle Simulator

Description

> **01/09 Update：在圖例中各色晶球右上方新增文字標示。**

本學期陪伴著大家的阿超，其實一直都在學習著魔法。
而他所學習的，是由蘊含力量的『晶球 (Orbs)』組成『魔法陣 (Magic Circle)』後來施放的。
將各種不同的晶球置入後會引發各種不同的反應，最終生成一個可以協助施法的魔法陣。

就跟學習寫程式一樣，施放魔法也需要不停的練習才能專精；
但阿超遇到了一個問題：晶球**很貴**。置入後又拿不回來，可經不起他隨意亂試亂浪費。
於是他打算以程式來模擬放置晶球後的魔法陣的結果，來節省他得來不易的辛苦錢。

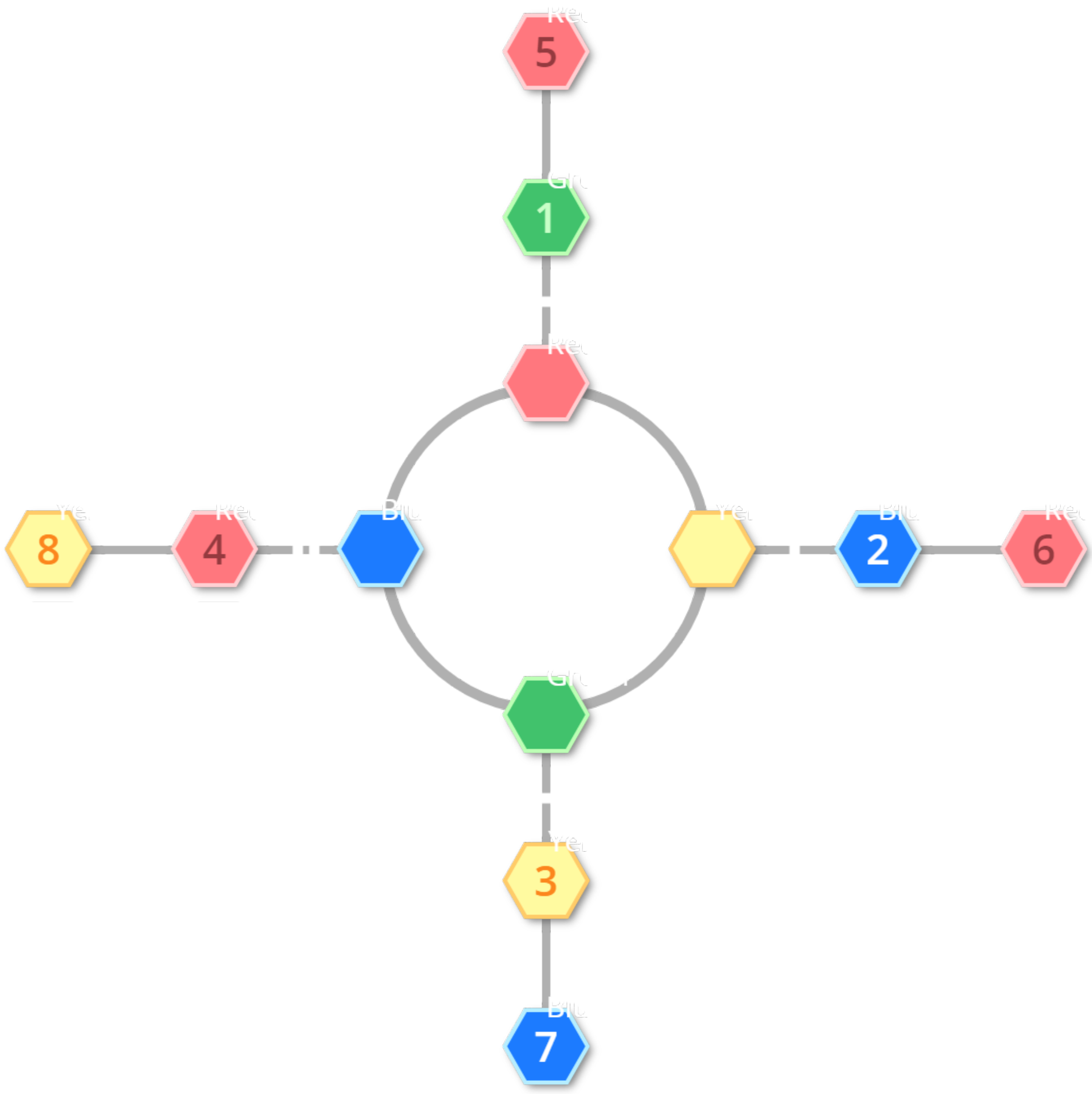
The Magic Circle

一個完整的魔法陣由以下兩個部分組成：

- 中心圓環，稱作『根』(Root)，**只能放置下方說明的普通晶球**
    - 將其中一個編號為 0，下方圖例為 12 點鐘方向。
- 自中心圓環連接出的的晶球列，稱作『枝』(Stem)，能放置任意晶球

- 以根部的各晶球為首只能形成單一路徑，不能有圓環、分支。
- 放置晶球時，必須自編號為 0 的根部開始依順時針擺放，不論發生什麼反應。

以下是一個簡單範例，球中的數字標示放置的順序：



Orbs

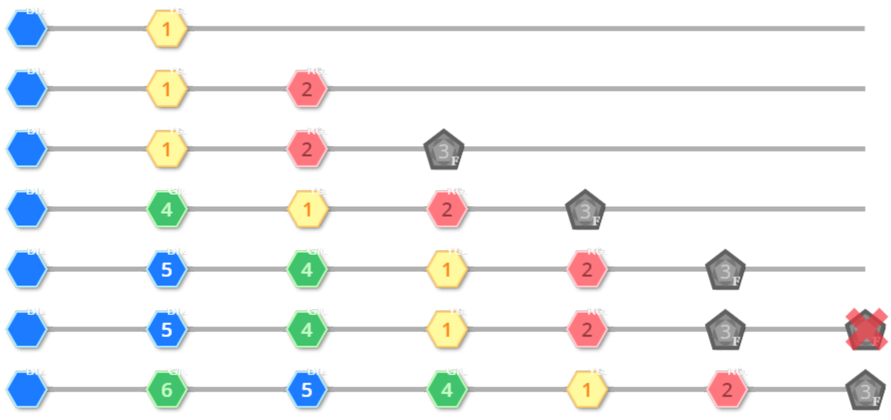
晶球名稱後方的括弧內的大寫英文字母，代表在 input 中對應的字元。\

Red Orb (R) / Blue Orb (B) / Green Orb (G) / Yellow Orb (Y) /

- 四種顏色的**普通晶球**，放置時不會產生任何反應。\

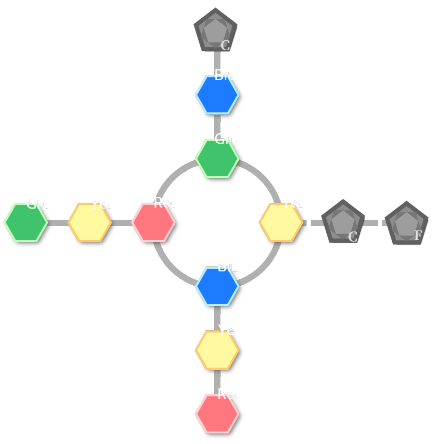
漂浮晶球 Floater Orb (F)

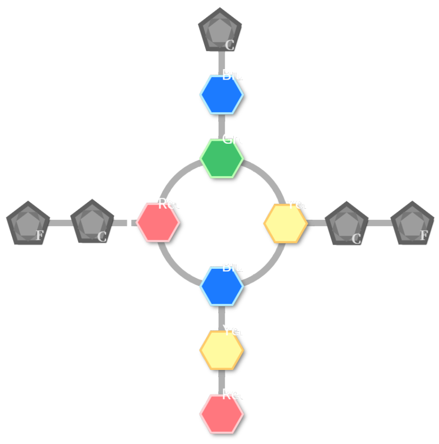
- 放置時不產生反應。
- 在同一個 Stem 上這個晶球後被置入的晶球，皆會被移動至該 Stem 的第一個位置。
- 同一個 Stem 上只可以有一個漂浮晶球，當置入第二個時該晶球會馬上消失，不對魔法陣做出任何改變。
- **[所有其他晶球的反應皆會在漂浮反應後發生]{.underline}**[。]{.underline}
- 以下為在某 Stem 上以 Y R F G B F G 的順序置入晶球的範例：
  [
  ]{.underline}



複製晶球 Cloning Orb (C)

- 置入在 Stem s 之後，計算目前整個魔法陣上總共有幾個複製晶球，假設為 k 個。
- 將自 s 的逆時針方向的第 k 個 Stem 的構成轉換為 s 的構成。
- 範例：

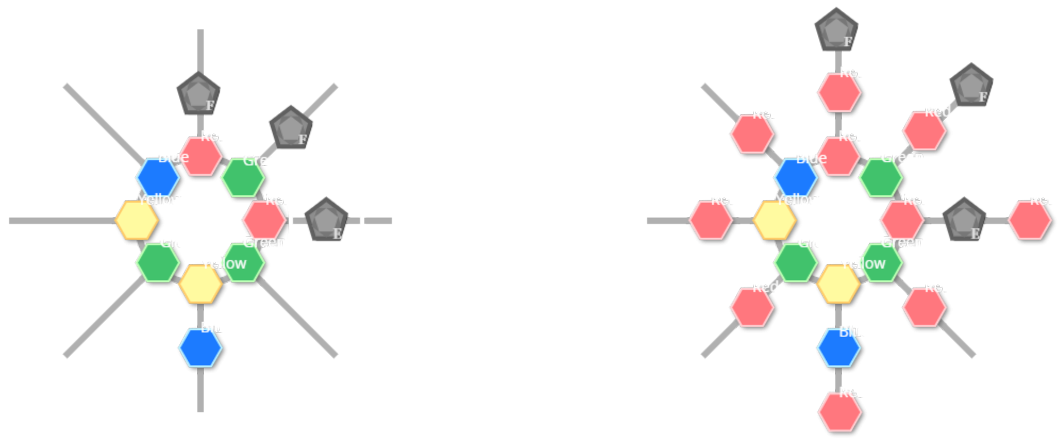




擴展晶球 Expand Orb (E)

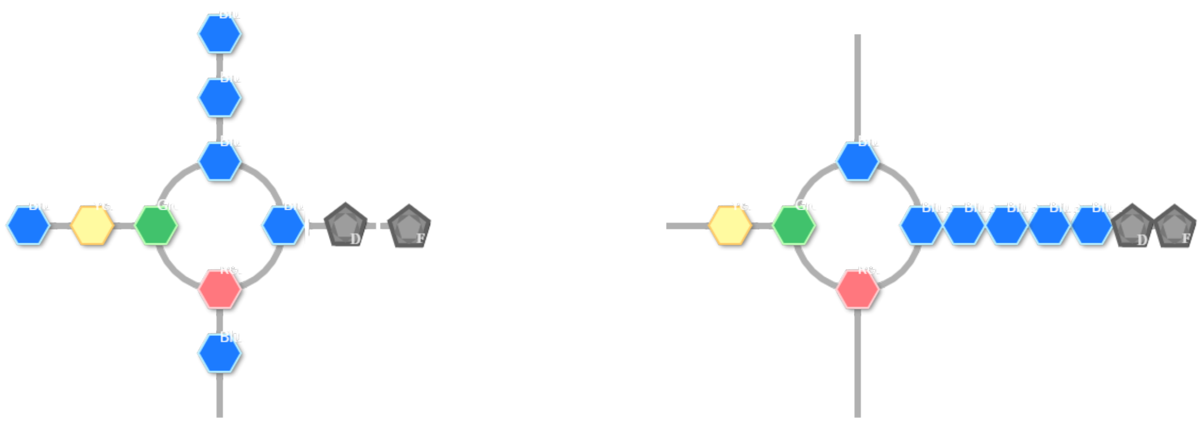
- 置入 Stem s 之後，生成與 s 根部相同的普通晶球置入魔法陣上的所有 Stem。

- 範例：



吸引晶球 Dyson Orb (D)

- 置入 Stem s 之後，將 s 以外的魔法陣上所有跟 s 的根相同的普通晶球吸引過來，一一置入。
- 範例:



Program Requirement

仔細閱讀以上說明以及下方的 loader code，撰寫程式碼令 Loader 可成功印出阿超完成的魔法陣的構成。

`head_t` 中的 `count` 可以提供你紀錄任何整數資料，但你不一定需要使用。

Input

The first line contains two integer `size` and `k` `size` indicates the size of the center ring. `k` indicates how many orbs will be placed into the magic circle by hand. 1 ≤ size ≤ 100 0 ≤ k ≤ 200; The Second Line contains `size` characters, Shows the configuration of the center ring, starting from index 0. The Third Line contains `k` characters, Shows the order to insert the orbs. The available characters are 'R' / 'G' / 'B' / 'Y' / 'C' / 'D' / 'E' / 'F'.

Output

Print the final configuration of the magic circle. From index 0, print the root at form "CO_X", then chain " -> X" To show the stem. nothing should appear after "CO_X" If the stem is empty The function print_magic_circle() is already finished, you don't need to implement it, but make sure your data is represented correctly for it to run.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>

// Enumrations
enum orb_type {
    O_UNSET = -1,
    O_RED = 17, O_BLUE = 1, O_GREEN = 6, O_YELLOW = 24,
    O_FLOAT = 5, O_CLONE = 2, O_EXPAND = 4, O_DYSON = 3
};

// Structure Declairation
struct node_t {
    struct node_t *next;
    enum orb_type orb;
};

struct head_t {
    struct node_t *head;
    int count;
    enum orb_type orb;
};

// Function Prototypes
struct head_t *alloc_list_heads(int size);              // Allocate memory for
array of list heads
void init_head_arr(struct head_t *arr, int size);       // Initialize array of
linked-lists
void print_magic_circle(struct head_t *arr, int size);  // Print the structure of
magic circle
void insert_next(struct head_t *arr, int size, const char c);  // Insert the next
orb onto the magic circle

// Main Function
int main() {
    int size;       // Size of center ring
    int count;      // Number of stem orbs will be put;

    // Allocate ring
    scanf("%d%d%*c", &size, &count);
    struct head_t *center_ring = alloc_list_heads(size);
    // Read and set ring
    init_head_arr(center_ring, size);

    // Place the orbs
    for (int i = 0; i < count; ++i) {
        insert_next(center_ring, size, getchar());
    }

    // Print magic circle configuration
    print_magic_circle(center_ring, size);
    return 0;
}

// Function Definitions
void print_magic_circle(struct head_t *arr, int size) {
    if (!arr) {
        return;
    }

    for (int i = 0; i < size; ++i) {
        // Print center orb
        printf("CO_%c", arr[i].orb + 'A');

        // Print stem
        struct node_t *cur = arr[i].head;
        while (cur) {
            printf(" -> %c", 'A' + cur->orb);
            cur = cur->next;
        }
        if (i + 1 < size) putchar('\n');
    }
    return;
}
```

Sample1

Input

```
3 11
RGB
RGBFFYYCDFE
```

Output

```
CO_R -> G -> C -> G -> F
CO_G -> G -> E -> C -> G -> F
```

```
CO_B -> B -> Y -> D -> G
```

Sample2

Input

```
5 6
YYYYY
YYGYYD
```

Output

```
CO_Y -> Y -> D -> Y -> Y -> Y
CO_Y
CO_Y -> G
CO_Y
CO_Y
```

Lec20

Bingo!

Description

UPDATE !!

由於有同學反應在 windows 電腦上面的 clion 無法正常執行 loader code 內容，

所以更新 loader code ,

請注意讀取時與 bingo function 的參數資料型態。

大家都玩過賓果吧！
以下是 4 * 4 的範例：

1 0 1 **1**
0 0 **1** **1**
0 **1** 0 **1**
1 0 1 **1**

可以看到有一條直的，一條斜的，總共兩條連線。

為了環保愛地球，一個 `n * n` 的賓果使用 `n * n` 大小的空間實在太浪費了！
聰明的 Amy 想到，在電腦中，所有的變數背後都是 0 或 1，那我們可以使用一些數字來替代這張表格，減少使

用的空間。

舉例來說，把這些 0 與 1 視作二進制的表示方式，並把他轉化為十進制

`1 0 1 1` -> 2^3 + 2^1 + 2^0 = `11`
`0 0 1 1` -> 2^1 + 2^0 = `3`
`0 1 0 1` -> 2^2 + 2^0 = `5`
`1 0 1 1` -> 2^3 + 2^1 + 2^0 = `11`

所以我們可以使用 `11 3 5 11` 來表示上圖的表格。

在這裡，我們要玩的是進階版賓果！
題目會給出 8 個 8 bits 的數字來表示 `8*8` 大小的表格，請你算出當中會有幾條連線

Here is 4 * 4 sample:

1 0 1 **1**
0 0 **1** **1**
0 **1** 0 **1**
1 0 1 **1**

There are two lines above, vertical and diagonal.

It's wasteful to use `n * n` space to play `n * n` bingo.
As every number is stored as 0 or 1 in computer, we can use some other numbers to represent that table.

For example, those 0 and 1 can be regarded as binary, and we can convert it into decimal.

`1 0 1 1` -> 2^3 + 2^1 + 2^0 = `11`
`0 0 1 1` -> 2^1 + 2^0 = `3`
`0 1 0 1` -> 2^2 + 2^0 = `5`
`1 0 1 1` -> 2^3 + 2^1 + 2^0 = `11`

So the above bingo can be represented as `11 3 5 11`.

Here we are going to play `8 * 8` bingo.
Eight 8 bits numbers are provided to represent the `8 * 8` bingo. Please count how many lines are in the bingo.

Input

8 8-bits numbers

Output

a number

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdint.h>

int bingo (uint16_t num[]);

int main () {
    uint16_t num[8];
    for (int i = 0; i < 8; ++i) {
        scanf("%hu", &num[i]);
    }

    printf("%d\n", bingo(num));
}
```

Sample1

Input

```
1 2 4 8 16 32 64 128
```

Output

```
1
```

Sample2

Input

```
255 2 6 10 18 35 66 130
```

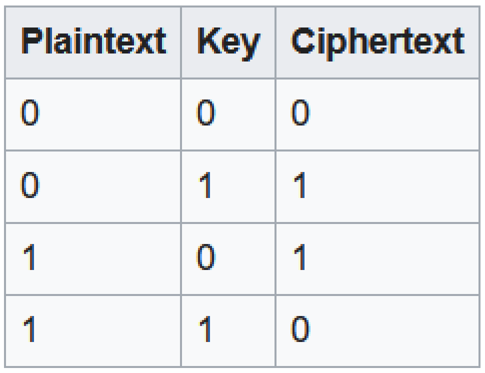
Output

```
3
```

XOR Cipher

Description

Given a string of text, which is our input data, encrypt the data by applying the bitwise XOR operator to every character using a given key.


XOR Cipher Trace Table

Plaintext	Key	Ciphertext
0	0	0
0	1	1
1	0	1
1	1	0

For example, given a plaintext data `TAs are so handsome` and the key `key`, the result cipher is:

- Plaintext data: `'T'`, `'A'`, `'s'`, `' '`, `'a'`, ...
- Key: `'k'`,`'e'`, `'y'`, `'k'`, `'e'`, ...
- Result (hex): `3f`, `24`, `0a`, `4b`, `04`, `0b`, `0e`, `45`, `0a`, `04`, `45`, `11`, `0a`, `0b`, `1d`, `18`, `0a`, `14`, `0e`

Note that you should output the raw result instead of hex value, which means that it might contain invalid ascii characters in the output.

Input

The first line is the key (1 <= len(key) <= 1000). The second line is the given data (1 <= len(data) <= 10000).

Output

The cipher text.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
key
TAs are so handsome
```

Output

```
?$
K□□□E
```

```
□E□
□□□
□□
```

Sample2

Input

```
T/dz%-*u`IFa
.EWS!1qH$a>^`4sG=U7SiVCY|M#^ (iNh8x3`<.9IIr"QM+(4S5yc[&m\8
```

Output

```
zj3)□□[=D(x?4□□=□x□&    □□8(bG$□□C;□q>R4□JCldXW1□mI`|Q□Fv□<q
```

2020_final

Find Maximum Fruit Number

Description

-英文題目-

Given a sequence (char input[]), there are many fruits in this sequence. (at most 20 kinds of fruit)

The fruits are connected with +_+ symbol.

Please output every fruit which has the maximum number of **occurrence.**

If there are many fruits which have the same maximum number of occurrence, follow char fruit[][15]'s order.

`hint: answer can more than one fruit`

-中文題目-

給定一個 sequence,

該 sequence 包含很多種水果(最多會出現20種水果),

並且每種水果以 +_+ 符號隔開，

其中這20種可能出現的水果已經定義在 Loader,

也傳入 find_fruit() (等同於 char fruit[][15]),

例如: fruit[0] 代表第一種水果, fruit[1]代表第二種水果等等，

請將所有出現最多次的水果都印出來

`如果有相同出現次數的水果，順序請依照 char fruit[][15] 中的順序，例如優先印出 guava 再來 litchi`

相同次數印出的先後順序:

guava>litchi>longan>watermelon>pomelo>pear>banana>papaya>pumpkin>tomato>mango>kiwi>persimmon>cantaloupe>strawberry>grape>peach>orange>coconut>lemon

`提示: 出現最多次的水果可能不只一種`

`提示: 請務必仔細閱讀 Loader code，來弄清楚整個程式的架構`

`hint: strtok()`

Input

There is one line for the sequence (char input[]) and has at most 20000 char. The fruits are connected with +_+ in this sequence (at least one fruit in this sequence).

Output

Output the fruit which has the maximum number of occurrence. (you need to print "\n" at the end of each line)。如果有相同出現次數的水果，順序請依照 char fruit[][15] 中的順序，例如優先印出 guava 再來 litchi 等等

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define N 20000

void find_fruit(char input[], char fruit[][15]);

int main(void) {
    char fruit[20][15] = {
        "guava",
        "litchi",
        "longan",
        "watermelon",
        "pomelo",
        "pear",
        "banana",
        "papaya",
        "pumpkin",
        "tomato",
        "mango",
        "kiwi",
        "persimmon",
        "cantaloupe",
        "strawberry",
        "grape",
        "peach",
        "orange",
        "coconut",
        "lemon"
    };
    char input[N];
    scanf("%s", input);
    find_fruit(input, fruit);
    return 0;
}
```

Sample1

Input

```
watermelon+_+watermelon+_+coconut+_+grape+_+coconut
```

Output

```
watermelon
coconut
```

Sample2

Input

```
pumpkin+_+coconut+_+guava+_+longan+_+peach+_+pomelo+_+coconut+_+litchi+_+lemon+_+c
oconut+_+persimmon+_+pear+_+coconut+_+banana+_+strawberry+_+tomato+_+cantaloupe+_+
orange+_+coconut+_+pear+_+tomato+_+orange+_+grape+_+pomelo+_+litchi+_+grape+_+pump
kin+_+cantaloupe+_+strawberry+_+coconut+_+kiwi+_+grape+_+pumpkin+_+pomelo+_+tomato
+_+pear+_+guava+_+papaya+_+coconut+_+kiwi+_+pear+_+mango+_+orange+_+peach+_+tomato
+_+watermelon+_+orange+_+longan+_+guava+_+papaya+_+papaya+_+tomato+_+pomelo+_+grap
e+_+pear+_+pear+_+longan+_+pear+_+kiwi+_+pumpkin+_+watermelon+_+strawberry+_+water
melon+_+pomelo+_+mango+_+persimmon+_+litchi+_+kiwi+_+persimmon+_+lemon+_+longan+_+
orange+_+litchi+_+kiwi+_+pear+_+mango+_+strawberry+_+longan+_+pomelo+_+grape+_+tom
ato+_+pomelo+_+peach+_+banana+_+lemon+_+longan+_+watermelon+_+litchi+_+papaya+_+st
rawberry+_+litchi+_+watermelon+_+litchi+_+pear+_+lemon+_+kiwi+_+tomato+_+guava+_+s
trawberry+_+litchi+_+kiwi+_+orange+_+kiwi+_+persimmon+_+pumpkin+_+pumpkin+_+waterm
elon+_+grape+_+watermelon+_+lemon+_+mango+_+pomelo+_+watermelon+_+coconut+_+mango+
_+longan+_+guava+_+strawberry+_+grape+_+guava+_+guava+_+orange+_+grape+_+litchi+_+
strawberry+_+strawberry+_+pear+_+grape+_+banana+_+lemon+_+tomato+_+orange+_+peach+
_+persimmon+_+tomato+_+orange+_+guava+_+pomelo+_+persimmon+_+watermelon+_+pomelo+_
+strawberry+_+pumpkin+_+papaya+_+persimmon+_+mango+_+longan+_+pear+_+pomelo+_+toma
to+_+orange+_+pear+_+banana+_+pomelo+_+coconut+_+persimmon+_+mango+_+watermelon+_+
pumpkin+_+peach+_+watermelon+_+tomato+_+cantaloupe+_+kiwi+_+litchi+_+strawberry+_+
guava+_+cantaloupe+_+lemon+_+persimmon+_+tomato+_+grape+_+coconut+_+tomato+_+straw
berry+_+watermelon+_+lemon+_+peach+_+pumpkin+_+pomelo+_+coconut+_+orange+_+litchi+
_+pomelo+_+litchi+_+lemon+_+tomato+_+kiwi+_+grape+_+tomato+_+papaya+_+coconut+_+co
conut+_+persimmon+_+litchi+_+kiwi+_+banana+_+longan+_+pomelo+_+pear+_+strawberry+_
+pear+_+persimmon+_+pear+_+strawberry+_+papaya+_+pumpkin+_+strawberry+_+grape+_+pu
mpkin+_+coconut+_+cantaloupe+_+orange+_+kiwi+_+mango+_+coconut+_+mango+_+lemon+_+l
itchi+_+pear+_+pumpkin+_+guava+_+grape+_+coconut+_+persimmon+_+orange+_+tomato+_+m
ango+_+lemon+_+pear+_+pumpkin+_+pear+_+kiwi+_+guava+_+mango+_+pear+_+papaya+_+mang
o+_+lemon+_+watermelon+_+mango+_+tomato+_+peach+_+papaya+_+guava+_+banana+_+pear+_
+watermelon+_+orange+_+coconut+_+guava+_+pear+_+coconut+_+peach+_+grape+_+longan+_
+pear+_+peach+_+cantaloupe+_+pomelo+_+longan+_+cantaloupe+_+tomato+_+pear+_+cantal
oupe+_+persimmon+_+mango+_+cantaloupe+_+strawberry+_+longan+_+peach+_+pear+_+kiwi+
_+pomelo+_+persimmon+_+pomelo+_+watermelon+_+mango+_+papaya+_+guava+_+pumpkin+_+pa
paya+_+coconut+_+lemon+_+grape+_+pear+_+cantaloupe+_+guava+_+longan+_+coconut+_+pe
ach+_+pomelo+_+kiwi+_+coconut+_+tomato+_+pear+_+mango+_+kiwi+_+coconut+_+peach+_+c
antaloupe+_+banana+_+litchi+_+orange+_+longan+_+banana+_+cantaloupe+_+orange+_+pea
ch+_+persimmon+_+mango+_+peach+_+lemon+_+guava+_+papaya+_+papaya+_+pear+_+cantalou
pe+_+lemon+_+papaya+_+kiwi+_+peach+_+watermelon+_+grape+_+strawberry+_+pomelo+_+gu
ava+_+peach+_+peach+_+mango+_+persimmon+_+litchi+_+pumpkin+_+banana+_+mango+_+long
an+_+persimmon+_+watermelon+_+guava+_+guava+_+grape+_+mango+_+pumpkin+_+papaya+_+m
ango+_+peach+_+strawberry+_+grape+_+tomato+_+cantaloupe+_+grape+_+persimmon+_+litc
hi+_+coconut+_+papaya+_+grape+_+watermelon+_+lemon+_+kiwi+_+lemon+_+litchi+_+pomel
o+_+persimmon+_+litchi+_+mango+_+watermelon+_+pomelo+_+strawberry+_+banana+_+pomel
o+_+banana+_+strawberry+_+banana+_+strawberry+_+litchi+_+peach+_+longan+_+papaya+_
+watermelon+_+kiwi+_+persimmon+_+coconut+_+peach+_+strawberry+_+tomato+_+watermelo
n+_+tomato+_+persimmon+_+grape+_+cantaloupe+_+watermelon+_+peach+_+orange+_+papaya
+_+coconut+_+lemon+_+mango+_+strawberry+_+pear+_+tomato+_+mango+_+kiwi+_+watermelo
n+_+peach+_+pear+_+peach+_+pomelo+_+pumpkin+_+grape+_+papaya+_+kiwi+_+papaya+_+coc
onut+_+lemon+_+cantaloupe+_+papaya+_+grape+_+watermelon+_+kiwi+_+mango+_+peach+_+b
anana+_+coconut+_+pear+_+cantaloupe+_+pumpkin+_+peach+_+pomelo+_+longan+_+cantalou
pe+_+pear+_+persimmon+_+pomelo+_+pumpkin+_+pumpkin+_+litchi+_+peach+_+persimmon+_+
litchi+_+kiwi+_+persimmon+_+pear+_+coconut+_+mango+_+pomelo+_+persimmon+_+tomato+_
+lemon+_+papaya+_+guava+_+tomato+_+grape+_+coconut+_+pumpkin+_+persimmon+_+kiwi+_+
pumpkin+_+guava+_+papaya+_+kiwi+_+cantaloupe+_+persimmon+_+watermelon+_+tomato+_+p
ersimmon+_+pomelo+_+mango+_+pumpkin+_+pumpkin+_+pomelo+_+lemon+_+guava+_+tomato+_+
mango+_+longan+_+pear+_+strawberry+_+watermelon+_+pear+_+cantaloupe+_+grape+_+bana
na+_+pumpkin+_+cantaloupe+_+banana+_+persimmon+_+pear+_+grape+_+persimmon+_+persim
mon+_+banana+_+orange+_+pear+_+litchi+_+banana+_+orange+_+orange+_+pumpkin+_+cocon
ut+_+coconut+_+pomelo+_+orange+_+mango+_+cantaloupe+_+lemon+_+cantaloupe+_+lemon+_
+cantaloupe+_+peach+_+peach+_+banana+_+pomelo+_+longan+_+banana+_+orange+_+tomato+
_+coconut+_+longan+_+peach+_+longan+_+grape+_+longan+_+lemon+_+persimmon+_+grape+_
+orange+_+litchi+_+pear+_+coconut+_+lemon+_+watermelon+_+longan+_+tomato+_+cantalo
upe+_+pumpkin+_+pumpkin+_+banana+_+lemon+_+longan+_+grape+_+grape+_+guava+_+lemon+
_+tomato+_+papaya+_+peach+_+mango+_+orange+_+kiwi+_+banana+_+guava+_+coconut+_+gua
```

```
va+_+kiwi+_+mango+_+peach+_+litchi+_+watermelon+_+litchi+_+lemon+_+watermelon+_+pe
ach+_+litchi+_+persimmon+_+tomato+_+tomato+_+guava
```

Output

```
pear
```

Happy Happy Warehouse Manager

Description

Mr. Huang is a worker who works in a warehouse. He loves to stake things up to build a tower, Every time the factory sends him newly manufactured goods, he will stack them on the top of his tower. Once a shop asking for goods, the goods that are on the top of the tower will be sent out first. After a busy day, goods on that tower may change a lot, which left him a difficult job to record. Now, you are asked to help him to maintain his happy tower.

There are three kinds of goods: `Juice`, `Wine` and `Laptop`. (I know that stacking laptop with drinks is a bad idea, but Mr. Huang loves to do that...) These are defined in the loader.

Then you will receive a list of records that represents what happened today, the record looks like:

`Produce Juice 5`, this means that factory sends Mr. Huang 5 boxes of juice.

`Consume 7`, means a shop asking for 7 boxes of goods (You don't need to concern what is in those box, just send them out).

After a busy day, you have to list the things left on the tower from the top.

`Hint`: You may implement using linked list to maintain that stack.

黃桑是一個快樂的倉儲工，他最喜歡把所有的東西都疊成高高塔。當工廠送來了新的貨物，他都會將他們疊到他的高塔上，而當有商店要批貨時，他也會直接將高塔頂端的貨物寄出去給他們。由於業務繁忙，因此每天高塔上的貨物變動會很大，這會讓每天的結算變得非常麻煩，現在要麻煩你替他設計一個倉儲系統，幫幫黃桑統計每天的貨物吧~!

黃桑經手的貨物總共有三種: `果汁`、`紅酒`跟`筆記型電腦`。(我知道把這些東西疊在一起不是個好主意，但親愛的黃先生喜歡.....)詳細的定義請見Loader。

每天結束後，你都會從黃桑那拿到一張記滿當日進出貨物的紀錄，紀錄長得像這樣:

`Produce Juice 5`, 代表工廠送來了5箱果汁。

`Consume 7`, 代表有商店批了7箱貨(你不需要在意箱子裡裝的是甚麼東西，把東西送出去就對了)。

當你統計完之後，請將塔上剩餘的東西從上至下列出。

`系統提示`: 或許你可以用linked list 的方式在你的電腦中蓋出虛擬的高塔。

Input

A list of records with EOF terminated. Hint: To generate EOF manually while you are testing your code locally, press ctrl+Z on windows and ctrl+D on linux.

Output

A list of remained goods from top to bottom. The output format is: "Good_type x remain_count" follows a newline character. If the tower is empty, you need to print "Empty" followed by a newline character.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

const char *good_types[] = {
    "Juice",
    "Wine",
    "Laptop"
};

void *produce(void *top_of_stack, const char *good_type, size_t count);
void *consume(void *top_of_stack, size_t count);
void list_remain(void *top_of_stack);

int main() {
    char record[64], *job, *good;
    size_t count;
    void *top_of_stack = NULL;
    while (fgets(record, 64, stdin)) {
        job = strtok(record, " ");
        if (!strcmp(job, "Produce")) {
            good = strtok(NULL, " ");
            count = atol(strtok(NULL, " "));
            for (int i = 0; i < 3; i++) {
                if (!strcmp(good, good_types[i])) {
                    top_of_stack = produce(top_of_stack, good_types[i], count);
                    break;
                }
            }
        }
        else if (!strcmp(job, "Consume")) {
            count = atol(strtok(NULL, " "));
            top_of_stack = consume(top_of_stack, count);
        }
    }
    list_remain(top_of_stack);
    return 0;
}
```

Sample1

Input

```
Produce Juice 10
Produce Laptop 5
Consume 3
Consume 3
Produce Wine 2
```

Output

```
Wine x 2
Juice x 9
```

Sample2

Input

```
Produce Juice 1
Consume 1
```

Output

```
Empty
```

Sample3

Input

```
Produce Juice 2
Produce Laptop 10
Produce Wine 1
Produce Juice 100
Produce Juice 10
```

Output

```
Juice x 110
Wine x 1
Laptop x 10
Juice x 2
```

Move the Maximum to the Tail

Description

Given n **different positive** integers, please move the maximum value to the tail of the linked list.

給定n個**相異的正整數**，請你將當中最大的那個數字移到該Linked-List的最後面。

Input

5 <= n <= 1000, each value will be less 100000.

Output

The content of the linked list.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>

struct node {
    int val;
    struct node *next;
};
struct node *head = NULL;
struct node *tail = NULL;
void MoveToTail();

int main(void) {
    int n;
    scanf("%d", &n);
    for(int i = 1; i <= n; ++i) {
        struct node *tmp = malloc(sizeof(struct node));
        scanf("%d", &tmp->val);
        tmp->next = NULL;
        if(i == 1)
            head = tmp;
        else
```

```
            tail->next = tmp;
        tail = tmp;
    }
    MoveToTail();
    for(struct node *cur = head; cur != NULL; cur = cur->next) {
        printf("%d ", cur->val);
    }
    return 0;
}
```

Sample1

Input

```
5
1 2 5 4 3
```

Output

```
1 2 4 3 5
```

Permutation

Description

Given a string with n alphabets(A~Z), print all permutations of it in alphabetical order. For example, if the input string is `ABC`, then output should be

`ABC`

`ACB`

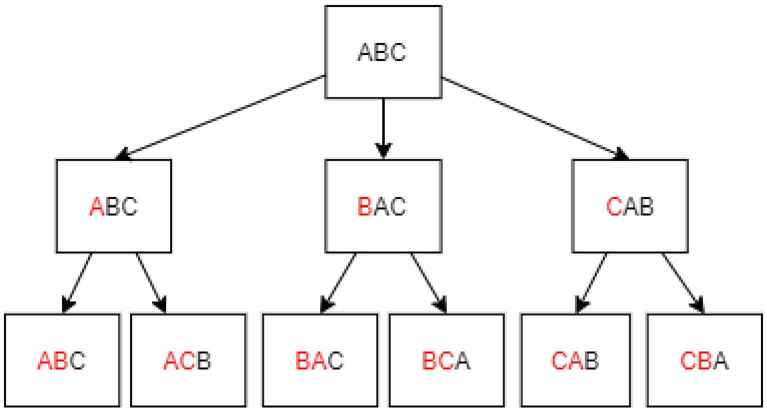
`BAC`

`BCA`

`CAB`

`CBA`

Hint: You can use recursion to solve this problem.



給定一字串有 n 個相異的字母(A~Z)，請列出此 n 個字母的所有排列可能性。(依照字典序輸出)

例如輸入為 `ABC`則輸出

`ABC`

`ACB`

`BAC`

`BCA`

`CAB`

`CBA`

Hint: 遞迴

Input

A string with n alphabets.(1<=n<=6)

Output

Print out n! kinds of permutations. Each permutation is followed by a newline character '\n'.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
BAKQ
```

Output

```
ABKQ
ABQK
AKBQ
AKQB
AQBK
AQKB
BAKQ
BAQK
BKAQ
BKQA
BQAK
BQKA
KABQ
KAQB
KBAQ
KBQA
KQAB
KQBA
QABK
QAKB
QBAK
QBKA
QKAB
QKBA
```

Sample2

Input

```
ACEG
```

Output

```
ACEG
ACGE
AECG
AEGC
AGCE
AGEC
CAEG
CAGE
CEAG
CEGA
CGAE
CGEA
```

```
EACG
EAGC
ECAG
ECGA
EGAC
EGCA
GACE
GAEC
GCAE
GCEA
GEAC
GECA
```

Rearrange Chatting History

Description

Nowadays, "LINE" is a well known communication application. "LINE" provides a feature that you can store your chatting history into text files(`.txt`). Unfortunately, you get a text file with incorrect order. Therefore, your task is to rearrange the chatting history into the ascending order with timestamp (**00:00 - 23:59**). Assume that at most one message per minute and the history is always on the same day.

現今，"LINE"是個知名的通訊應用程式。" LINE"提供了一項功能，你可以將聊天記錄存儲到文本文件(`.txt`)中。很不幸的是，你拿到了一份錯誤順序的文本文件。因此，你的任務就是將聊天紀錄依照時間升序(**00:00

- 23:59**)重新排序。假設每分鐘最多只有一則訊息且記錄一定都是在同一天。

Input

First line is the total message (<= 100). Then, the followings are the each message information separated by newline with the format: hour:miniute name content.

Output

All message information in correct order with the format same as Input.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#define MESSAGE_LENGTH 64
#define NAME_LENGTH 10

struct timestamp {
    int hour, minute;
};
```

```
struct message_info {
    struct timestamp time;
    char name[NAME_LENGTH];
    char message[MESSAGE_LENGTH];
};

int cmp(const void *a, const void *b);
void sort_message(struct message_info *m, int *total);

int main(void) {
    struct message_info m[100];
    int total;
    scanf("%d", &total);
    for(int i = 0; i < total; i++) {
        scanf("%d:%d %s %[^\n]", &m[i].time.hour, &m[i].time.minute, m[i].name,
m[i].message);
    }
    sort_message(m, &total);
    for(int i = 0; i < total; i++) {
        printf("%02d:%02d %s %s\n", m[i].time.hour, m[i].time.minute, m[i].name,
m[i].message);
    }
    return 0;
}
```

Sample1

Input

```
9
09:30 STUDENT I'm worried that I won't pass PD1.
09:00 TA Hello everyone.
09:35 TA Why do you think so?
09:46 TA Send a sticker.
09:41 STUDENT However, today is the final exam.
09:01 TA If anyone has question about PD1.
09:45 TA Well, I can just say "Haiya".
09:02 TA If no. Hope everyone will be statisfied with your grade.
09:40 STUDENT Because I still have lots of practices to do.
```

Output

```
09:00 TA Hello everyone.
09:01 TA If anyone has question about PD1.
09:02 TA If no. Hope everyone will be statisfied with your grade.
09:30 STUDENT I'm worried that I won't pass PD1.
```

```
09:35 TA Why do you think so?
09:40 STUDENT Because I still have lots of practices to do.
09:41 STUDENT However, today is the final exam.
09:45 TA Well, I can just say "Haiya".
09:46 TA Send a sticker.
```

Sample2

Input

```
5
10:27 B Good morning.
00:14 A Good night.
17:41 B Good evening.
12:54 A Good afternoon.
22:50 A Good night again.
```

Output

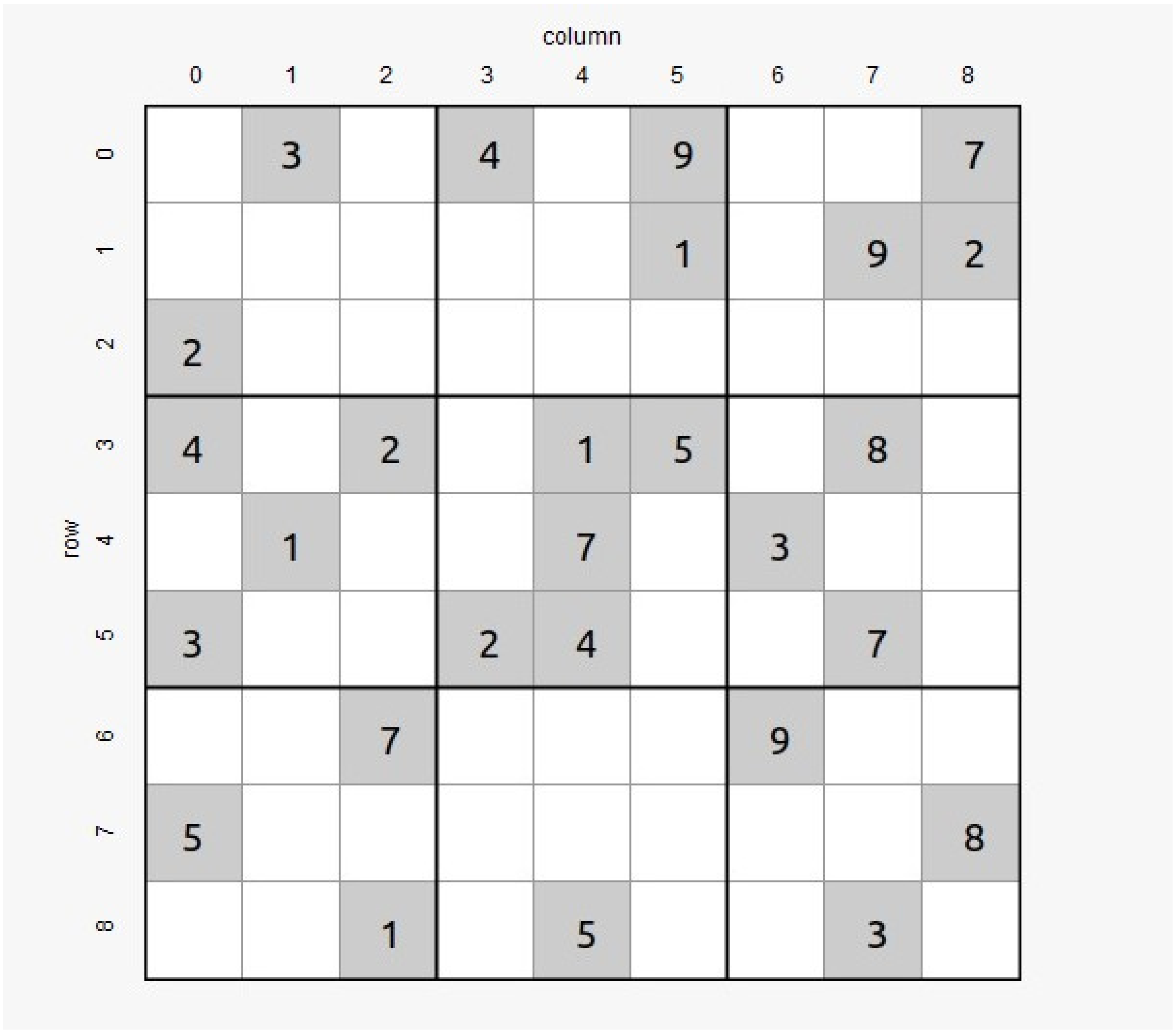
```
00:14 A Good night.
10:27 B Good morning.
12:54 A Good afternoon.
17:41 B Good evening.
22:50 A Good night again.
```

Sudoku Solver

Description

Sudoku is a logical based, combinatorial number-placement puzzle. The objective is to fill a **9×9** grid with digits so that **each column, each row, and each of the nine 3×3 subgrids that compose the grid contain all of the digits from 1 to 9**.

For example:



The puzzle is represented using a 9 by 9 structure array (`struct element grid[9][9]`), and each element's `num` is an integer between 0~9 (**0 means there is no value yet, corresponding to the white area in the figure, and `can_modify` is set to true**). **The elements that you should not change are the gray shaded area in the figure** and they belong to this problem (`can_modify` **is set to false**).

Implement the `solve()` function that can automatically solve a given sudoku puzzle (you should directly modify `grid[9][9]` and fill in the correct answer).

Note: There is exactly one solution for each testcase.

數獨是一個邏輯和數字組合的遊戲。其目標是將數字填入 9x9 的格子內，並使其每一行，列，和 3x3 的格子皆由 1～9 的數字所組成。

例如：如上圖

本次題目中，數獨由一個 9x9 的 struct array 表示 (`struct element grid[9][9]`)，而每個 element 的 `num` 為介於 0~9 的 integer (0 表示還沒有填入任何數字，且其 `can_modify` 設為 true)。不應更改圖中灰色區域的格子，這些格子為題目 (`can_modify` 設為 false)。

實作 `solve()` 函數來自動解出給定的數獨 (直接修改 `grid[9][9]` 即可)。

註：每個測資都會是唯一解。

Input

The input is a sudoku puzzle. The elements, which you should fill in, are represented by '.'.

Output

The output is the solved sudoku puzzle.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdbool.h>

struct element {
    int num;        // 1~9: valid, 0: don't have value yet
    bool can_modify;       // true: white area in the figure, false: gray area in
the figure
};

void solve(struct element (*grid_p)[9]);
void print_grid(struct element (*grid_p)[9]) {
    for (int i = 0; i < 9; ++i) {
        for (int j = 0; j < 9; ++j) {
            printf("%d ", grid_p[i][j].num);
        }
        printf("\n");
    }
}

int main(void) {
    struct element grid[9][9];
    char ch;
    for (int i = 0; i < 9; ++i) {
        for (int j = 0; j < 9; ++j) {
            scanf(" %c", &ch);
            if (ch == '.') {    // can_modify assign true, num assign 0
                grid[i][j].num = 0;
                grid[i][j].can_modify = true;
            }
            else {      // value gotten from input, so we can't change this
                grid[i][j].num = ch - '0';
                grid[i][j].can_modify = false;
            }
        }
    }
    solve(grid);
    print_grid(grid);
    return 0;
}
```

Sample1

Input

```
. 3 . 4 . 9 . . 7
. . . . . 1 . 9 2
2 . . . . . . . .
4 . 2 . 1 5 . 8 .
. 1 . . 7 . 3 . .
3 . . 2 4 . . 7 .
. . 7 . . . 9 . .
5 . . . . . . . 8
. . 1 . 5 . . 3 .
```

Output

```
1 3 6 4 2 9 8 5 7
7 5 8 6 3 1 4 9 2
2 9 4 5 8 7 1 6 3
4 7 2 3 1 5 6 8 9
6 1 5 9 7 8 3 2 4
3 8 9 2 4 6 5 7 1
8 2 7 1 6 3 9 4 5
5 6 3 7 9 4 2 1 8
9 4 1 8 5 2 7 3 6
```

2020_midterm

Calculating Volume

Description

Read an integer `mode`, you need to calculate the volume for the mode. In different modes, you need to read different number of double precision floating-points and then print out the volume that is rounded to 2 decimal places. The followings are 4 modes:

1. Mode 1:
   In this mode, you need to `scanf` **a double precision floating-point which represents the length of cube(立方體)**.
   volume = length * length * length.

2. Mode 2:
   In this mode, you need to `scanf` **three double precision floating-points which sequentially represent length, width and height of cuboid(長方體)**.
   volume = length * ** width * height.

3. Mode 3:
   In this mode, you need to `scanf` **two double precision floating-points which sequentially(依序) represents the radius and height of cylinder(圓柱體)**.
   volume = 3.14 * radius * radius * height.

4. Mode 4:
   In this mode, you need to `scanf` **an double precision floating-point which represents the length of regular tetrahedron(正四面體).**
   volume = √2 / 12 * length * length * length.

Given 1 <= `mode`<= 4, each double precision floating-point is in range 1 ~ 100

Hint:

1. You can use the **sqrt()** and **pow()** function by using the following preprocessor directive:
   `#include <math.h>`

2. All the values should be double precision floating-point, including **PI(3.14)**.

讀入一個整數 `mode`，你必須計算該模式下的體積。在不同模式中，你必須要能讀入不同個數的雙精度浮點數(double)並且印出體積(四捨五入至小數點後第2位)。以下為4種模式：

1. 模式 1:
   在模式 1 中，你必須 `scanf` **一個雙精度浮點數代表立方體的長度(length)。**
   立方體體積 = 長度 * 長度 * 長度。

2. 模式 2:
   在模式 2 中，你必須 `scanf` **三個雙精度浮點數依序代表長方體的長(length)、寬(width)、高(height)**
   長方體體積 = 長 * 寬 * 高。

3. 模式 3:
   在模式 3 中，你必須 `scanf` **兩個雙精度浮點數依序代表圓柱體的半徑(radius)、高(height)**
   圓柱體體積 = 3.14 * 半徑 * 半徑 * 高。

4. 模式 4:
   在模式 4 中，你必須 `scanf` **一個雙精度浮點數代表四面體的長度(length)。**
   四面體體積 = √2 / 12 * 長度 * 長度 * 長度。

給定 1 <= `mode` <= 4 且每個雙精度浮點數的範圍會在 1 ~ 100。

提示:

你可以藉由使用`#include<math.h>` 來呼叫函式 **sqrt()** 和 **pow()**。

所有的值都必須使用雙精度浮點數來計算，包含**PI(3.14)**。

Input

Include 2 lines. First line represents the mode. Second line is the information for the mode.

Output

Double precision floating-point is rounded to 2 decimal places.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1
6
```

Output

```
216.00
```

Sample2

Input

```
2
6 8 10
```

Output

```
480.00
```

Department Store Anniversary

Description

The NCKU department store is having an anniversary sale. Customers can get free gift if they spend more than $100. On the 8th floor, people line up in rows of two in front of the customer service counter to get their gift. Suddenly, one of the desk goes out of service, so customers in front of this desk must merge into the other line. For the sake of fairness, staffs need to help people to line up in one line **in the order of their arrival**.

Luckily, you are the one who remember the order of everyone's arrival. (what a reliable staff, huh?) Can you decide the final order of the new line?

成大百貨公司正在舉辦周年慶，消費滿 100 元即可兌換滿額贈禮。顧客們紛紛擠上八樓，兩個服務台都排滿了長長的隊伍。突然間，其中一個服務台的機器故障了，前方的人龍通通得合併到正常運作的服務台前的隊伍。為了確保先到的人都能先領到禮品，合併後的隊伍順序必須要是**顧客抵達八樓的順序**才行。還好，你就是那個記得所有顧客的抵達順序的員工（真厲害）。你能協助其它員工一起安排隊伍嗎？

Input

The input consists of 3 lines. The first line contains 2 integers m, n. Each line indicates the number of people. And 10 <= m, n <= 1000. The second line is an array of integers and its length is m. The third line is an array of integers and its length is n. The numbers a[i] in the arrays indicate the order of arrival, and 1 <= a[i] <= 10000. Every number is unique, and may not be continuous.

Output

An array of integer, indicates the order of people in the new line. Each number must followed by a space. No need to print newline at the end of the line.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
3 7
1 8 9
2 5 7 11 16 17 21
```

Output

```
1 2 5 7 8 9 11 16 17 21
```

Odd Magic Square

Description

The **odd** magic square consists of consecutive integers (starting with 1 and ending with n^2) placed into n rows by n columns.

The sum of each row, each column and each diagonal are the same.

Hint:

1. Place number 1 in the middle of first row.
2. Check if the **upper right** grid is empty:
- if empty: fill the next number in this **upper right** grid
- if not empty: fill the next number in the **bottom of current grid**
3. Check if the last step **exceeds the boundary**, you need to return to the **other side**.

   You can see the following example to understand the process !

奇數魔方陣是指在nxn的方陣中，放入從1開始到n^2的連續數字。

使每一行、每一列與每條對角線的和皆相同。

提示:

1. 首先在第一列的最中間那個方格填入1。
2. 接著檢查該方格的**右上方**是否為空的:
- 如果是空的: 將下個數字填入**右上方的方格**
- 如果不是空的: 將下個數字填入**當前方格的正下面那格**
3. 如果在上一步會**超出邊界**，則需要返回**另一邊**

   你可以看下方的範例來了解過程!

For example:n = 3

step1:

0 **1** 0

0 0 0

0 0 0

step2:

0 1 0

0 0 0

0 0 **2**

step3:

0 1 0

3 0 0

0 0 2

step4:

0 1 0

3 0 0

4 0 2

Finally, after 9 steps:

8 1 6

3 5 7

4 9 2

Input

There is an odd integer between 1 and 50

Output

Output the square. There is a space after each number.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5
```

Output

```
17 24 1 8 15
23 5 7 14 16
4 6 13 20 22
10 12 19 21 3
11 18 25 2 9
```

Sample2

Input

```
7
```

Output

```
30 39 48 1 10 19 28
38 47 7 9 18 27 29
46 6 8 17 26 35 37
5 14 16 25 34 36 45
13 15 24 33 42 44 4
21 23 32 41 43 3 12
22 31 40 49 2 11 20
```

Plague Inc

Description

"Plague Inc." is a famous game, you can collect DNA points to evolve your own virus in this game.

Now you made a new virus in laboratory, and want to know how many people will be infected in a month after you "accidentally" leak this virus into the world.

Given a double value `R` and an integer value `N`, `R` is `spread rate`, means each day the number of infected people will multiplied by `R`

Please calculate the number of infected people after `N` days after you let the first people infected by this virus.

"瘟疫公司" 是個非常有名的遊戲，在遊戲中你可以收集DNA點數並讓你的病毒進化

現在你在實驗室中製造了一種新型病毒，並且想知道在你 "不小心" 讓這種病毒洩漏出去後的一個月，會有多少人被感染

定義兩個數值：雙倍精度浮點數`R`和整數`N`， `R`為傳播率，每天被感染者人數會乘上`R`

請計算你用這種病毒感染第一人後`N`天被這種病毒感染的總人數

Input

One line contains R and N, separated with a space, 1 <= R <= 2, 1 <= N <= 30

Output

An integer, representing the number of infected people (ignore digits after decimal point)

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1.53 29
```

Output

```
227013
```

Sample2

Input

```
1.55 23
```

Output

```
23857
```

Rotate Image

Description

In computer, everything is represented by numbers --- so does image. Computer stores image as a matrix. A cell in the matrix is a pixel in the image, and number in the cell represents the color of that pixel.

If we want to rotate an image, it means that we want to rotate the matrix itself. For example, consider an image as follows:

$$\begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix}$$

If we want to **rotate this image 90 degrees to the right**, we can rotate the matrix like this:

$$\begin{vmatrix} 7 & 4 & 1 \\ 8 & 5 & 2 \\ 9 & 6 & 3 \end{vmatrix}$$

Now given the degree of **right rotation**, can you help me rotate my images?

在電腦裡，所有東西都是用數字來表達的------就連圖片也不例外。電腦以矩陣的形式來儲存每一張圖片，矩陣裡的格子代表著圖片的像素，而格子裡的數字就代表了該像素的顏色。

當我們要對圖片做旋轉時，我們實際上就是在旋轉這個矩陣本身。舉例來說，如果有一張圖片長這樣：

```
| 1 2 3 |
| 4 5 6 |
| 7 8 9 |
```

如果我們想將這張圖片**向右旋轉 90 度**，那麼就等於是把矩陣旋轉成這樣：

```
| 7 4 1 |
| 8 5 2 |
| 9 6 3 |
```

給定**向右旋轉的角度**，你能將圖片做正確的旋轉嗎？

Input

The input consists of 3 part. The first line is an integer, indicates the degree of right rotation. It will be one of the following numbers: 90, 180 or 270. The second line contains 2 integers m & n, indicates the dimension of the image. And 3 <= m, n <= 100. The following m lines contain n integers, representing the image itself. Each integer in the image is in the range of [0, 100].

Output

The rotated image. Each number must follow by a space. You need to print newline at the end of last line.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
90
3 4
1 5 6 7
5 8 9 2
3 0 4 7
```

Output

```
3 5 1
0 8 5
4 9 6
7 2 7
```

Vigenère Cipher Encoder

Description

The Vigenère cipher is a method of encrypting alphabetic text by using a series of interwoven Caesar ciphers, based on the letters of a keyword. In a Caesar cipher, each letter of the alphabet is shifted along some number of places. For example, in a Caesar cipher of shift 3, `A` would become `D`, `b` would become `e`, `y` would become `b` and so on. The Vigenère cipher has several Caesar ciphers in sequence with different shift values, which are determined by a text string called "key". The "key" only contains lowercase letters.

For example, let's assume the "key" is `abcde`. Each alphabet of the "key" is converted to its corresponding numeric value (e.g.`a` = `0`, `b` = `1`, ... `z` = `25`):

- alphabet: `a b c d e`
- numeric value: `0 1 2 3 4`

To encrypt the message, say "**TAs are so H4ND5OME.**", we should arrange the "key" as follows:

`a b c d e a b c d e a b c d`

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

We now shift each **alphabet** of the message (1.) by the numeric "key" we arranged in the previous step (2.) to create ciphertext (3.) as shown below:

1. `T A s a r e s o H N D O M E`
2. `0 1 2 3 4 0 1 2 3 4 0 1 2 3`
3. `T B u d v e t q K R D P O H`

And the plaintext message "**TAs are so H4ND5OME.**" becomes "**TBu dve tq K4RD5POH.**" in ciphertext.

Write a program that prints out the Vigenère cipher for the given input (described in Input Format).

Note: Transform only English alphabet (case-preserving) and preserve other characters such as Arabic numerals or punctuation.

維吉尼亞密碼是使用一系列凱薩密碼來加密文字的算法，透過一組密鑰來進行加密。在凱薩密碼中，每個英文字母都會偏移一個固定的數字。例如，在偏移量為3的凱薩密碼中，`A` 會變成 `D`, `b` 會變成 `e`, `y` 會變成 `b`，以此類推。維吉尼亞密碼則是由好幾組不同偏移量的凱薩密碼編排而成，透過一個稱為「密鑰」的字串來決定其偏移值。而這個密鑰僅包含小寫英文字母。

舉例來說，假設密鑰是 `abcde`。密鑰中的每個字母會轉為其對應的數字(例如`a` = `0`, `b` = `1`, ... `z` = `25`)：

- 英文字母: `a b c d e`
- 數字: `0 1 2 3 4`

要加密 "**TAs are so H4ND5OME.**" 這段訊息，我們要將密鑰如下排列：

`a b c d e a b c d e a b c d`

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

接著根據前一步驟排列的密鑰(2.)，對訊息中的每個**英文字母**(1.)做位移來產生加密的文字(3.)，如下：

1. `T A s a r e s o H N D O M E`
2. `0 1 2 3 4 0 1 2 3 4 0 1 2 3`
3. `T B u d v e t q K R D P O H`

而這段純文字訊息 "**TAs are so H4ND5OME.**" 會變成 "**TBu dve tq K4RD5POH.**"，即為加密後的訊息，意思是助教好帥 😉。

寫一個程式，根據給定的輸入來輸出對應的維吉尼亞加密。

註：只需轉換英文字母(保留大小寫)且保留其他字元例如阿拉伯數字或標點符號。

Input

The input contains three lines. All of these lines are followed by a newline character. The first line contains the length of the "key", and 1 <= length(key) <= 100. The second line is the "key", which contains only lowercase letters. Finally, the third line is the plaintext message.

Output

Output the ciphertext of the given input. There is no newline character at the end of the output.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5
abcde
TAs are so H4ND5OME.
```

Output

```
TBu dve tq K4RD5POH.
```

Sample2

Input

```
4
duck
PD1 is sooooo awesome!!
```

Output

```
SX1 kc viqyri cghmqwh!!
```

2021_final

[20 Points] Let's Build Great Great Walls

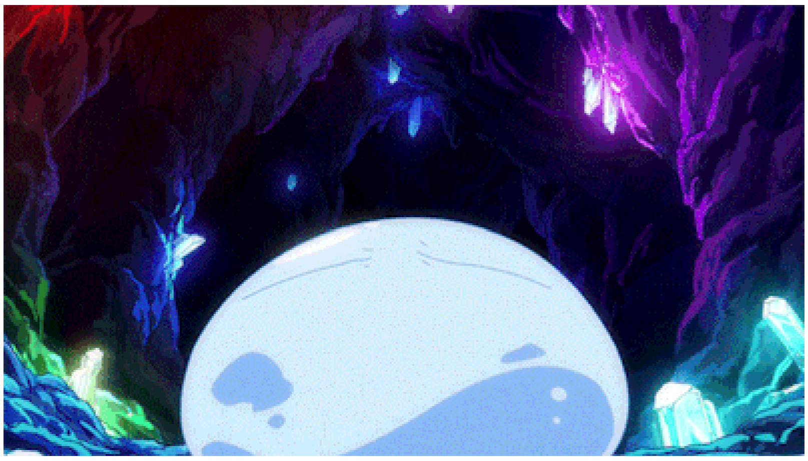
Description

利姆路是隻可愛的始萊姆，雖然是它只是隻始萊姆，但它統領著森林中的魔物們。自古以來，魔物們最大的威脅莫過於人類的殺戮，利姆路為了保護它所管理的魔物們，它決定讓魔物們在村莊四周蓋上城牆，但是在它領導的眾多魔物村莊中，僅有部份村莊生產建造城牆所需的材料，因此其他村莊需要向這些村莊下訂這些材料。經過一年汲汲營營的建造，利姆路的辦公桌上積滿了來自各個村莊下訂建材的訂單，請你寫一個程式幫助它統整這些訂單。

在這題中，你僅需要完成程式的特定函式，在這個函式中，你會拿到**由訂單組成的 linked-list**，其中每筆訂單都會包含以下資訊：訂單的編號、下定的村莊、下定的建材編號以及數量。請注意：**建材數量的資訊被包裝在 union 中**，你必須根據建材編號讀取特定的欄位。你要實做的函式需要**替每個村莊收集屬於它的訂單**，並將它們串接成**獨立的 linked-list**，其中的訂單須按照**訂單編號由少至多**進行排序，並且，請替每個村莊**計算過去這年它共花了多少錢**購買建材（不同建材的單價可以在 loader code 中找到）。最後，為了簡化之後的查詢速度，請**將村莊按照名稱排列。**

Rimuru is a cute slime who was killed by a random murder in his previous life, now he becomes the leader of the monsters live in the forest. To help the monsters prevent threads from the human world, he plans to make monsters build great walls around their villages. Since there are lots of villages in his monster kingdom, and only several of them are able to supply the building parts, they must trade those ingredients between the villages. After a year went through, the orders between villages are piled up like a mountain, please, help him to unify the orders.

In the following program, you will get **several orders constructed as a linked-list**, each order contains following things: order id, consumer of the order, the ingredient id and the amount of the ingredient. Please notice that **the amount of the ingredient are packed in an union**, you have to retrieve it from the specific member depends on the ingredient id. For each village, your program should not only **gather the orders belongs to it**, **reconstruct them into an independent linked-list**, **sort the orders in ascending order by the order id**, but also **calculate the total cost** of all orders (the unit cost of each ingredient can be found in the loader program). Finally, your program should **sort the result list of villages via their name**.



Input

輸入的第一行有兩個整數：V 跟 O，分別代表村莊的數量跟訂單的總筆數，接著有 V 行村莊的名字，剩下的 O 行為訂單的內容，訂單內容為以下格式：「訂單編號:建材編號/建材數量-下訂村莊」。 限制： 1. 1 <= V <= 75 2. 1 <= O <= 1024 3. 1 <= 建材數量 <= 1024 4. 村莊名稱 < 16 5. 訂單編號是一個 16 進位的字串，長度為 8 The first line contains two integers V, O that indicates the number of villages and orders, the next V lines are the village names. The rest of O lines are the orders which are in the format "order_id:integration_id/amount-consumer", each order occupies a line. restrictions: 1. 1 <= V <= 75 2. 1 <= O <= 1024 3. 1 <= amount <= 1024 4. length of village name < 16 5. order id is a hex string with length 8

Output

程式須輸出每個村莊的名、它一年來的建材費用以及排序後，屬於該村莊的訂單的編號。 Each village and its total cost to get the ingredients followed by the id of the orders belong to the village, the output ids should be sorted in ascending order.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define ORDER_ID_LEN 8
#define MAX_VILLAGE_NAME_LEN 15

typedef struct order_s {
    struct order_s *next;
    char order_id[ORDER_ID_LEN + 1];
    char consumer[MAX_VILLAGE_NAME_LEN + 1];
    enum {
        INGRED_A = 1,
        INGRED_B,
        INGRED_C,
    } ingredient_id;
    union {
        char ing_a[8];
        long ing_b;
        double ing_c;
    } amount;
} order_t;

typedef struct village_s {
    char name[MAX_VILLAGE_NAME_LEN + 1];
    unsigned long total_cost;
    order_t *order_list;
} village_t;

static const long unit_costs[] = {
    [INGRED_A] = 256,
    [INGRED_B] = 512,
    [INGRED_C] = 1024,
};

static village_t *init_village_array(int v_cnt);
static order_t *init_order_list(int o_cnt);
static void unify_orders(village_t *vills, int v_cnt, order_t *orders);
static void print_result(village_t *vills, int v_cnt);

int main()
{
    int village_cnt, order_cnt;
    scanf("%d%d", &village_cnt, &order_cnt);
    village_t *villages = init_village_array(village_cnt);
    if (!villages) {
        goto init_village_failed;
    }
    order_t *orders = init_order_list(order_cnt);
    if (!orders)  {
        goto init_order_failed;
    }
    unify_orders(villages, village_cnt, orders);
    print_result(villages, village_cnt);
    free(orders);

init_order_failed:
    free(villages);

init_village_failed:
    return 0;
}

static village_t *init_village_array(int v_cnt)
{
    village_t *villages = malloc(sizeof(village_t) * v_cnt);
    if (!villages) { // Ran out of memory
        return NULL;
```

```c
    }
    for (int i = 0; i < v_cnt; i++) {
        scanf("%s", villages[i].name);
    }
    return villages;
}

static order_t *init_order_list(int o_cnt)
{
    order_t *orders = malloc(sizeof(order_t) * o_cnt);
    if (!orders) { // Ran out of memory
        return NULL;
    }
    for (int i = 0; i < o_cnt; i++) {
        // Read the content of order
        // the term %[^c] will use 'c' as delimiter while scanning the input
        scanf(" %[^:]:%u/%[^-]-%s",
                orders[i].order_id,
                &orders[i].ingredient_id,
                orders[i].amount.ing_a,
                orders[i].consumer);

        switch (orders[i].ingredient_id) {
            case INGRED_B:
                orders[i].amount.ing_b = atol(orders[i].amount.ing_a);
                break;
            case INGRED_C:
                orders[i].amount.ing_c = atof(orders[i].amount.ing_a);
                break;
            default:
                break;
        }
        // Construct orders into linked-list
        if (i) {
            orders[i - 1].next = orders + i;
        }
    }
    // End the linked-list
    orders[o_cnt - 1].next = NULL;
    return orders;
}

static void print_result(village_t *vills, int v_cnt)
{
    for (int i = 0; i < v_cnt; i++) {
        printf("%s %ld\n  ->", vills[i].name, vills[i].total_cost);
        for (order_t *optr = vills[i].order_list; optr; optr = optr->next) {
            printf(" %s", optr->order_id);
        }
        printf("\n");
    }
}
```

Sample1

Input

```
3 15
Guanmiao
Sinsing
Taoyuan
79B3801D:3/59-Guanmiao
48D32269:2/834-Guanmiao
3A699371:3/249-Taoyuan
3C52083B:1/337-Guanmiao
6916A3F4:2/299-Taoyuan
753181AF:2/84-Taoyuan
158C8543:1/260-Guanmiao
4C9470D9:3/624-Guanmiao
76FDD0A1:2/723-Taoyuan
34C1A646:1/788-Sinsing
49C2EB29:1/772-Sinsing
657D29BB:1/746-Taoyuan
24335A5C:3/940-Taoyuan
526BB92A:3/171-Guanmiao
6761B64C:3/117-Taoyuan
```

Output

```
Guanmiao 1454336
  -> 158C8543 3C52083B 48D32269 4C9470D9 526BB92A 79B3801D
Sinsing 399360
  -> 34C1A646 49C2EB29
Taoyuan 2094592
  -> 24335A5C 3A699371 657D29BB 6761B64C 6916A3F4 753181AF 76FDD0A1
```

Sample2

Input

```
4 18
Yujing
Yanshuei
Sinsing
```

```
Gangshan
059D0501:2/441-Sinsing
362B46E0:3/992-Sinsing
34713753:2/167-Sinsing
1C1E24ED:3/679-Yanshuei
6504A3FD:1/487-Yujing
598BD3F5:1/491-Sinsing
6F8037D0:2/725-Yujing
28B56E0C:3/891-Yanshuei
3931CEF1:1/109-Yanshuei
39B15E23:2/789-Sinsing
572EC1A6:2/2-Yujing
5137318F:2/458-Yujing
12117749:2/701-Sinsing
0D23AF31:2/821-Yanshuei
2B897B63:3/462-Yujing
19A04B37:2/126-Yujing
50005E7A:3/1019-Yanshuei
44AA8935:1/533-Gangshan
```

Output

```
Gangshan 136448
  -> 44AA8935
Sinsing 2215680
  -> 059D0501 12117749 34713753 362B46E0 39B15E23 598BD3F5
Yanshuei 3099392
  -> 0D23AF31 1C1E24ED 28B56E0C 3931CEF1 50005E7A
Yujing 1268992
  -> 19A04B37 2B897B63 5137318F 572EC1A6 6504A3FD 6F8037D0
```

Sample3

Input

```
5 20
South
Sinshih
Danei
Daliao
Hunei
166C47E2:1/245-Sinshih
7375D5C3:2/553-South
12B62880:2/762-South
58C3A4C6:1/403-Daliao
37B7F642:3/38-Sinshih
```

```
44369036:2/338-Daliao
052D5D75:1/626-Sinshih
2DACD55A:3/245-Danei
75F41330:1/350-Sinshih
6EE8A654:2/269-Daliao
2A0633F4:3/371-South
0CA3ECFD:2/970-Hunei
0380EBFD:2/778-Hunei
31EC81E4:1/366-Danei
60BCEDD1:1/647-Danei
7C8F0D3E:3/105-Danei
05B5E563:1/355-Sinshih
1987954B:2/653-Danei
50C09674:1/506-Danei
52654703:3/689-Danei
```

Output

```
Daliao 413952
  -> 44369036 58C3A4C6 6EE8A654
Danei 1787136
  -> 1987954B 2DACD55A 31EC81E4 50C09674 52654703 60BCEDD1 7C8F0D3E
Hunei 894976
  -> 0380EBFD 0CA3ECFD
Sinshih 442368
  -> 052D5D75 05B5E563 166C47E2 37B7F642 75F41330
South 1053184
  -> 12B62880 2A0633F4 7375D5C3
```

Sample4

Input

```
7 25
Annan
Zuojhen
Nanhua
River
Daliao
Alian
Maolin
1A299881:3/197-River
229EC19C:1/1021-Zuojhen
46BA7EA1:3/618-Annan
402C8159:3/319-River
4AFA3035:2/551-Maolin
```

```
07C76667:3/191-River
1F5F8051:3/589-Daliao
16609029:2/282-Daliao
690461E6:1/22-Alian
0C16D905:3/145-Maolin
0F046D3F:1/133-Maolin
0EDDACF7:2/652-Alian
2416050E:3/401-Zuojhen
72A828B7:1/566-Nanhua
0F34EE9E:3/486-Nanhua
21A00A47:1/883-River
412F50DC:2/945-Alian
12091799:3/393-Alian
688A9F39:1/882-Annan
7BC3E7D0:1/349-Annan
2143B6AD:3/84-Daliao
4E385D3A:2/109-Zuojhen
1BD4570B:3/673-Annan
0CE41551:1/384-Nanhua
4F1BFD9A:1/16-Alian
```

Output

```
Alian 1229824
  -> 0EDDACF7 12091799 412F50DC 4F1BFD9A 690461E6
Annan 1637120
  -> 1BD4570B 46BA7EA1 688A9F39 7BC3E7D0
Daliao 833536
  -> 16609029 1F5F8051 2143B6AD
Maolin 464640
  -> 0C16D905 0F046D3F 4AFA3035
Nanhua 740864
  -> 0CE41551 0F34EE9E 72A828B7
River 950016
  -> 07C76667 1A299881 21A00A47 402C8159
Zuojhen 727808
  -> 229EC19C 2416050E 4E385D3A
```

[20 Points] String Combination

Description

沒有錯，又是組合！

給予 n 個字串及數字 k，請將所有可能的組合列出來。

例如：給予 `n = 5`, `k = 2`，以及 `andy becky cindy jacky willy` 五個字串，

代表從 5 個字串中挑選 2 個出來，注意，須根據字串給予時的順序組合進行列印。詳細請見範例。

Yah, Combination Again!

Given n strings and number k, please print all possible combinations out.

For example, given `n = 5`, `k = 2`, and `andy becky cindy jacky willy` five strings,

which means that you have to choose 2 strings from the 5 strings.

Notice that you need print out them according to the given order. See examples.

Input

輸入第一行為 n 跟 k，第二行有 n 個字串（以空格分開）。 1 <= n <= 20，且每個字串不超過十個字。 Print out k strings from the n strings, according to the given order in the input.

Output

印出 n 個字串取 k 個的結果，須根據字串給予時的順序組合進行列印。 Print out k strings from the n strings, according to the given order in the input.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5 2
andy becky cindy jacky willy
```

Output

```
andy becky
andy cindy
andy jacky
andy willy
becky cindy
becky jacky
becky willy
cindy jacky
cindy willy
jacky willy
```

Sample2

Input

```
5 3
andy becky cindy jacky willy
```

Output

```
andy becky cindy
andy becky jacky
andy becky willy
andy cindy jacky
andy cindy willy
andy jacky willy
becky cindy jacky
becky cindy willy
becky jacky willy
cindy jacky willy
```

Sample3

Input

```
8 1
andy becky cindy jacky larry mandy sandy willy
```

Output

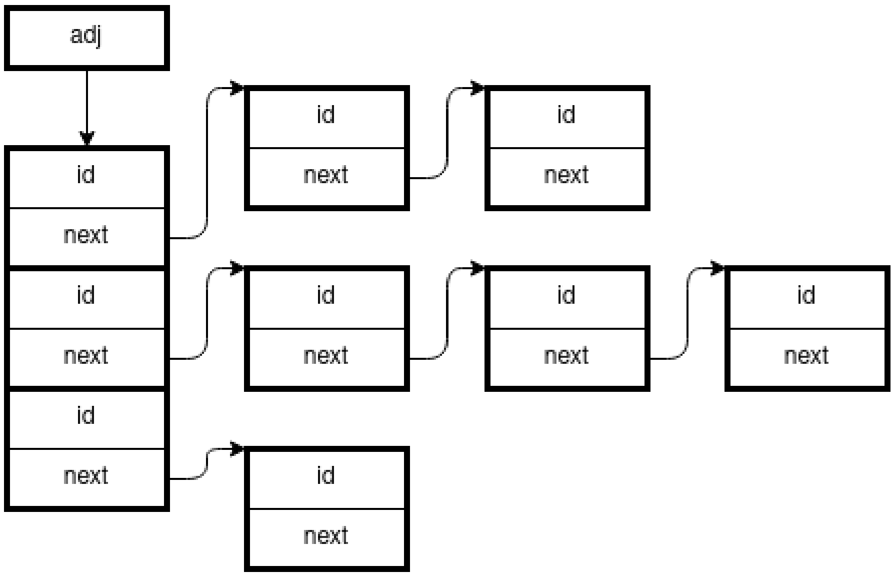
```
andy
becky
cindy
jacky
larry
mandy
sandy
willy
```

[25 Points] Birdy's Network

Description

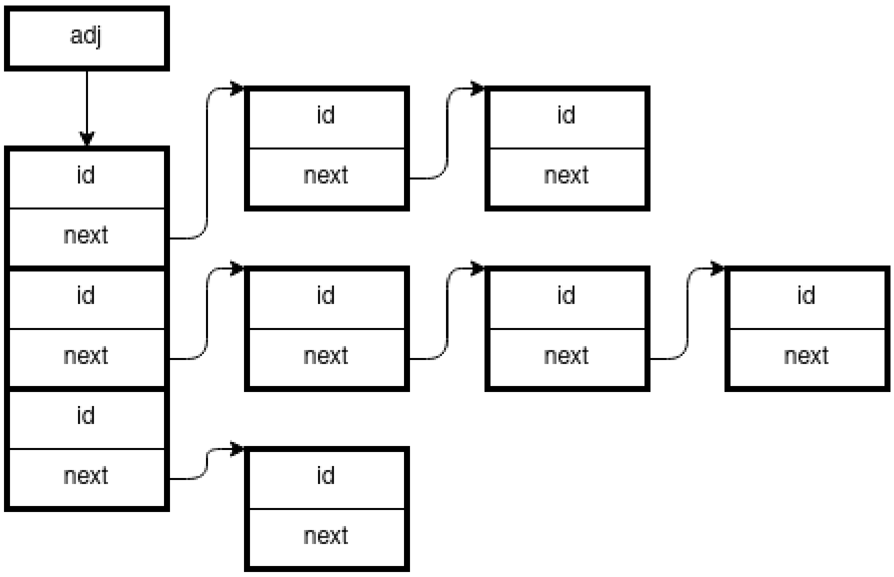
Birdy 是一個網路管理員，他想要使用一個 **鄰接表** (adjacency list) 來紀錄他負責管理的網路。
我們可以使用一條 list 紀錄當前節點的所有相鄰節點(neighbor node)，並將所有的節點資訊儲存於一條陣列

中，
以下是本鄰接表的示意圖：



同時，他也希望鄰接表上的節點按照 id 由小到大的排序。
請你幫助他建立並排序這張鄰接表。

Birdy is a network administrator. He wants to use an **adjacency list** to represent his network. One representation of adjacency list is an array of linked lists which records the neighbors for each node. The structure of the adjacency list is shown as following:



He also wants to make the order of the adjacency list ascending. Please help him load the data to an adjacency list and reorganize its order.

Note that the connection is directed.

Input

Input contains multiple lines (<= 1000 lines). The first number in each line is node id. The numbers after "->" are the neighbor ids of current node (in ascending order). The last number in each line is always 0, which indicates the end of the line. All ids are positive integers (1 <= id <= 2147483647).

Output

Output contains the neighbor ids of each node (list in ascending order).

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>

struct node {
    int id;
    struct node *next;
};

struct node *read_data(int *size); // TODO: implement this function

void print_data(struct node *adj, int size) {
    if (!adj) {
        return;
    }
    for (int i = 0; i < size; ++i) {
        printf("%d -> ", adj[i].id);
        for (struct node *ptr = adj[i].next; ptr != NULL; ptr = ptr->next) {
            printf("%d ", ptr->id);
        }
        putchar('\n');
    }
    return;
}

void free_data(struct node *adj, int size) {
    if (!adj) {
        return;
    }
    for (int i = 0; i < size; ++i) {
        struct node *ptr = adj[i].next;
        while (ptr) {
            adj[i].next = ptr->next;
            free(ptr);
            ptr = adj[i].next;
        }
    }
    free(adj);
    return;
}

int main() {
    int size;
    struct node *adj = read_data(&size);
    if (!adj) {
        fprintf(stderr, "Error allocating memory.");
        exit(1);
    }
    print_data(adj, size);
    free_data(adj, size);
    return 0;
}
```

Sample1

Input

```
100356 -> 4 458 0
4 -> 458 0
458 -> 0
```

Output

```
4 -> 458
458 ->
100356 -> 4 458
```

Sample2

Input

```
32 -> 14 22 0
37 -> 14 32 0
8 -> 22 37 0
14 -> 8 22 32 37 0
22 -> 8 14 32 37 0
```

Output

```
8 -> 22 37
14 -> 8 22 32 37
```

```
22 -> 8 14 32 37
32 -> 14 22
37 -> 14 32
```

[25 Points] Order Recovery

Description

小混蛋工廠最近遇到了大麻煩！

他們的系統不知道出了什麼問題，接收到的訂單可能會少字，並以奇怪的格式呈現。

請協助他們恢復訂單內容，並告訴他們該訂單要的貨到底是什麼！

舉例來說，當訂單收到的是 `ter:mel`，代表要的貨有兩個（以 `:` 隔開），其中一個包含字串 `ter`，另一個包含字串 `mel`。

然而，工廠內有太多東西了，以符合 `ter` 來說，可能是 `water`，可能是 `battery`，也可能是 `butter`。（沒錯，這間工廠什麼都有。）

因此小混蛋工廠的員工決定只選擇「最先符合該字串」的物品，當作客人要的貨，上述三項的話，將會選擇 `water` 當作結果。

（因為 `water` 是在第三個字母開始符合，其他兩者都是在第四個字母才開始符合。）

當然，若有兩個以上字串符合上述情況，則優先選擇字數最短的字串，字數相同時，則比較兩字串各字元，優先選擇字元較前面的字串。

例如：訂單要求 `ham`，則 `hammer` 和 `hamburger` 將選擇 `hammer`。

又例如：訂單要求 `yo`，則 `yolk` 和 `yoyo` 將選擇 `yolk`（因為順序上 `l` 比 `y` 前面）。

最後回到訂單的舉例，若 `ter` 選出的貨為 `water`, `mel` 選出的貨為 `melon`，則最後以 `+` 連結，輸出 `water+melon`。

另外，每張訂單最少有 1 樣物品，但最多不會超過 30 樣物品。

Bestards' Factory has a big trouble!

Somewhere is wrong in their system, so the order they received may miss some characters, and is showed in a strange way.

Please help them recover the order, and tell they what the customers want.

For example, if the order's content is `ter:mel`, it means there are two items（separated by `:`）, one includes substring `ter`, and the other includes substring `mel`.

However, there are too many goods, we use `ter` as an example, it may be `water`, `battery`, or `butter`. (Yah, the factory sells everything!)

Hence we choose the goods that「is matched by the substring from the more forward position」, as the goods that the customers want.

In the example, we will choose `water`. (Since `water` matches `ter` from the third position, while the other two match from the fourth position.)

If there are at least two goods meet the condition, then we choose the string with shorter length.

If the length are the same, then compare two strings, and choose the one that is less than the other.

For example, if `ham`, then `hammer` and `hamburger`, we choose `hammer`.

Another example, if `yo`, then `yolk` and `yoyo`, we choose `yolk`. (Since in ascii, `l` is prior to `y`.)

Back to the first example, if `ter` represents for `water`, and `mel` represents for `melon`, then concatenate them by using `+` and print out `water+melon` as the result.

Every order orders at least 1 goods, but won't exceed 30 goods.

Input

輸入為一字串，其中以 ':' 分隔訂單物品。 The input is a string, and the goods are separated by ':'.

Output

將各物品以 '+' 連結，輸出該筆訂單內容。 Concatenate each goods by using '+', and then print it out.

Loader Code

Your code will be judge using this program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define GOODS_SIZE 120
#define MAX_ORDER_SIZE 30
#define MAX_NAME_LEN 15

void recovering_order(char *, char **);

char names[GOODS_SIZE][MAX_NAME_LEN] = {
    "aluminum", "ammunition", "orange", "aspirin", "ax",
    "basketball", "zipper", "bell", "blanket", "drum",
    "canvas", "flute", "eraser", "mirror", "comb",
    "eggplant", "diamond", "zucchini", "door", "butter",
    "egg", "dart", "chair", "encyclopedia", "china",
    "fan", "glue", "elevator", "flower", "hat",
    "garlic", "strawberry", "glove", "flour", "ice",
    "ship", "hammer", "fur", "key", "honey",
    "grape", "icecream", "lock", "iron", "ivory",
    "jacket", "oil", "uniform", "kiwi", "pan",
    "helmet", "keyboard", "yoyo", "jet", "knife",
    "tie", "lemon", "xerox", "ink", "mutton",
    "newspaper", "melon", "coconut", "motorcycle", "longan",
    "nail", "rice", "magnet", "peanut", "ointment",
    "oar", "jeans", "nut", "arrow", "oven",
    "jewel", "wool", "noodle", "pencil", "lamp",
    "refrigerator", "needle", "roast", "vest", "yarn",
    "saw", "hamburger", "soap", "glass", "wine",
    "tangerine", "pillow", "umbrella", "zinc", "train",
    "uglifruit", "toast", "underwear", "vanilla", "jeep",
    "unicycle", "vase", "rope", "violin", "warship",
    "volleyball", "water", "yogurt", "submarine", "passionfruit",
    "yacht", "rose", "window", "yolk", "kite",
    "liquor", "xylophone", "toothbrush", "battery", "doll"
};

int main(void) {
    char order[500] = "", *result = NULL;
    scanf("%s", order);
    recovering_order(order, &result);
    printf("%s", result);
    free(result);
    return 0;
}
```

Sample1

Input

```
ter:mel
```

Output

```
water+melon
```

Sample2

Input

```
nut:il
```

Output

```
nut+oil
```

Sample3

Input

```
in:lo:low
```

Output

```
ink+lock+flower
```

[30 Points] Magic Bug

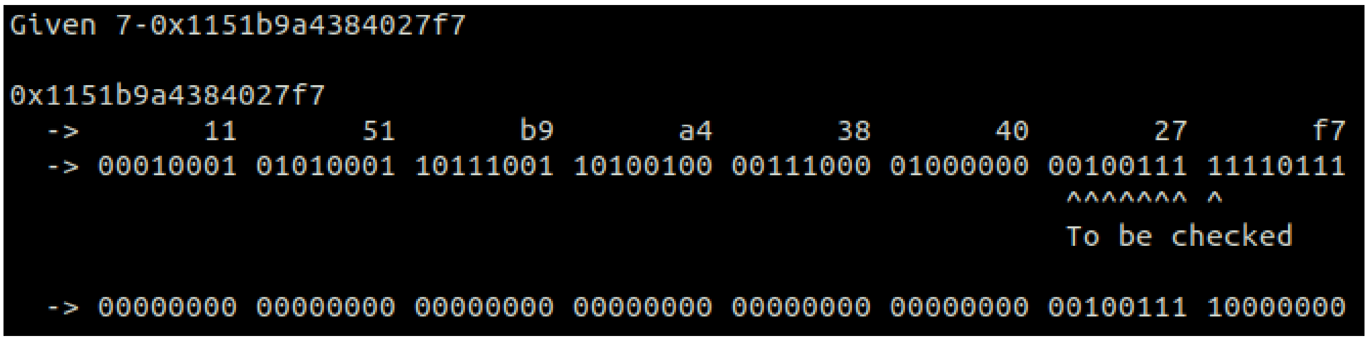
Description

司波達也是名魔法工程師，平常的工作是替客戶調校及改善魔法道具（想不到吧，魔法也跟程式一樣喔～），今天他接到了一個棘手的案子，不管用什麼方式都找不到 Bug 在哪裡，因此他決定直接把魔法道具的記憶體倒出來看看，請你幫他開寫一支程式來印出魔法道具的記憶體內容。

你的程式會得到兩個數字：stb 跟 num，其中 num 是長度固定為 8 位元組的數字，而 stb 代表司波想要檢測的記憶體內容的開頭。司波每次固定檢查從 stb 開始的 8 個位元，為了避免模糊焦點，請將不需要被檢測的記憶體內容設為 0。

Tatsuya is a magic engineer whose daily job is to calibrate and improve the magic tools. One day he got sucked in a tricky job that an unknown bug takes him a week and still no progress. To find out the mistake of the magic program, he decides to dump the entire memory of the magic tool, please help him to fulfill that.

Your program will recieve two integer: stb and num, where num is an 8-byte number, and stb indicates the begining bit that Tatsuya interested in. Notice that only 8 bits will be checked each time, please zero out the rest of bits to zero before print out the content.



```
Given 7-0x1151b9a4384027f7

0x1151b9a4384027f7
  ->       11       51       b9       a4       38       40       27       f7
  -> 00010001 01010001 10111001 10100100 00111000 01000000 00100111 11110111
                                                            ^^^^^^^ ^
                                                            To be checked

  -> 00000000 00000000 00000000 00000000 00000000 00000000 00100111 10000000
```

Input

輸入的第一行是一個整數 N 代表記憶體內容的數量，之後的 N 行皆是記憶體內容，且輸入符合「stb-num」的格式，其中 stb 代表要檢查的記憶體區段的起始位置，而 num 為一個固定為 8 位元組的 16 進位數字。 The first line contains an integer N that indicates the number of memory contents to print. The rest of N lines are the memory content in the format "stb-num" where stb indicates the start bit, and num is an integer in hex format with fixed length in 8 bytes.

Output

對於每行記憶體內容，請印出相對應的二進位內容。 N lines of integers in binary format, there should be a whitespace between two bytes of with in an integer.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
6
6-0x36c3bf4154901a51
24-0x6cd08f786630bc7d
7-0x49662b322f44fe5
4-0xd19c56a0052f04e
24-0x50f25f5349ba72e6
27-0x4486df6f6fdda642
```

Output

```
00000000 00000000 00000000 00000000 00000000 00000000 00011010 01000000
00000000 00000000 00000000 00000000 01100110 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 01001111 10000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 01000000
```

```
00000000 00000000 00000000 00000000 01001001 00000000 00000000 00000000
00000000 00000000 00000000 00000111 01101000 00000000 00000000 00000000
```

Sample2

Input

```
11
48-0x3f9b91e666a0e8b
15-0x64feb3044281b8ad
25-0x5256e33a7882f5d1
47-0x75f7469226eb2fdd
27-0xe1013e93f389ce9
45-0x74205f5211db3a0
18-0xf58b2e3427ec21d
15-0x65e523c00e7427f3
9-0x49028dfe4917ca7e
20-0x70222da20482132a
30-0x373d4c0b087bcc49
```

Output

```
00000000 11111001 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000001 10000000 00000000
00000000 00000000 00000000 00000000 01111000 00000000 00000000 00000000
00000000 01110111 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000001 00111000 00000000 00000000 00000000
00000000 00000010 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000010 01111100 00000000 00000000
00000000 00000000 00000000 00000000 00000000 01110100 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000001 11001010 00000000
00000000 00000000 00000000 00000000 00000100 10000000 00000000 00000000
00000000 00000000 00000000 00001011 00000000 00000000 00000000 00000000
```

Sample3

Input

```
6
7-0x1151b9a4384027f7
26-0x5d63ecf71ec5a259
54-0x606a23ce0dfa5d1c
19-0x1f4b24710d61928b
```

```
1-0xbd1352f748aa177
42-0xad8c1441228c898
```

Output

```
00000000 00000000 00000000 00000000 00000000 00000000 00100111 10000000
00000000 00000000 00000000 00000011 00011100 00000000 00000000 00000000
00100000 01000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000101 01100000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000001 01110110
00000000 00000000 11000000 00000000 00000000 00000000 00000000 00000000
```

Sample4

Input

```
1
26-0x463180af3385451e
```

Output

```
00000000 00000000 00000000 00000011 00110000 00000000 00000000 00000000
```

[30 Points] Pull The Radish

Description

在開心農場裡，Ariel 種了 `N` 根蘿蔔，每個蘿蔔的間距為一公尺，今天是收成的日子，但 Ariel 不想要一次拔光所有蘿蔔，所以他決定以自己為起點 (起點為 `y`)，分別向左和向右，每隔 `x` 公尺就拔一根蘿蔔，請輸出 Ariel 今天總共收成了幾根蘿蔔，並告訴他現在農場的情況。

`farm[N]` 表示農場狀況，若 `farm[i]` 為 1 ，表示在 index 為 `i` 時，有種一根蘿蔔；若 `farm[i]` 為 0 ，表示在 index 為 `i` 時，沒有種蘿蔔。(index `i` 與 `i+1` 之間表示間隔為一公尺)

Notice: 請完成 `pull_radish` 函式，並回傳總共拔了幾個蘿蔔，注意參數是什麼！

In happy farm, Ariel planted `N` radishes. The radishes are spaced at one-meter intervals. It is time to harvest, but Ariel doesn't want to pull all radishes. Therefore, she deicides to start with herself (starting point is `y`), and pulls radishes every `x` meters to left and right, respectively. Please output how many radishes Ariel pulls and the situation of farm.

`farm[N]` represents the situation of farm. If `farm[i]` is 1, that means there is a radish at index `i` ; If `farm[i]` is 0, that means there isn't a radish at index `i`.

Notice: Please finish `pull_radish` function, and return the number of radishes that Ariel pulls. Pay attention to the parameter!

Input

There are three number, integer N , y and x, represent the number of radishes, Ariel's position and the interval.

Output

Output the total of radishes which Ariel pulled and the array farm[N]

Loader Code

Your code will be judge using this program:

```
#include<stdio.h>
#include<stdlib.h>

int pull_radish(int *l, int *r, int *pos, int x);

int main() {
    int N, y, x;
    scanf("%d %d %d", &N, &y, &x);
    int *farm = malloc(N * sizeof(int));

    // initialize farm[N]
    for(int i=0; i<N; i++) {
        farm[i] = 1;
    }

    int total = pull_radish(&farm[0], &farm[N-1], &farm[y], x);
    printf("%d\n", total);
    for(int i=0; i<N; i++) {
        printf("%d ", farm[i]);
    }

    return 0;
}
```

Sample1

Input

```
8 2 2
```

Output

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

Sample2

Input

```
15 14 5
```

Output

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

2021_midterm

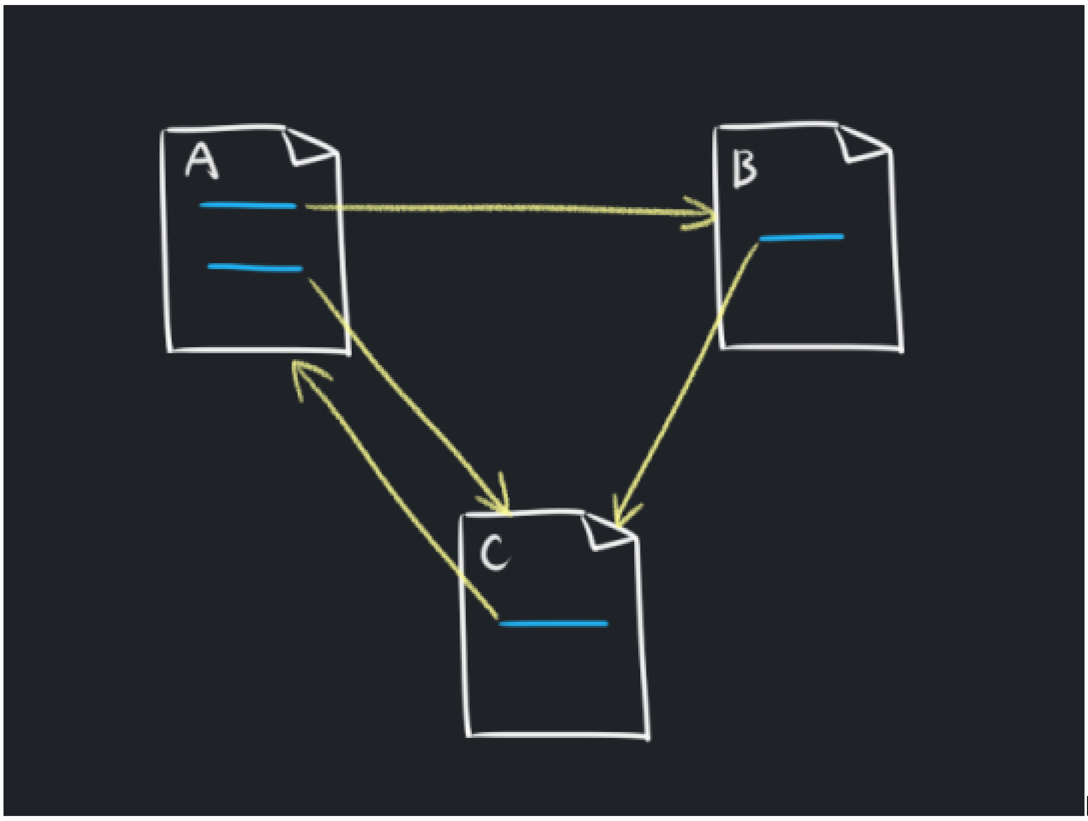
[15 points] PageRank

Description

PageRank is an algorithm that used by Google to sort the searching result. This algorithm calculates the rank of a page via the quality and the number of hyperlinks that point to it. The algorithm supposes that there are many hyperlinks point to a web page if it is important.

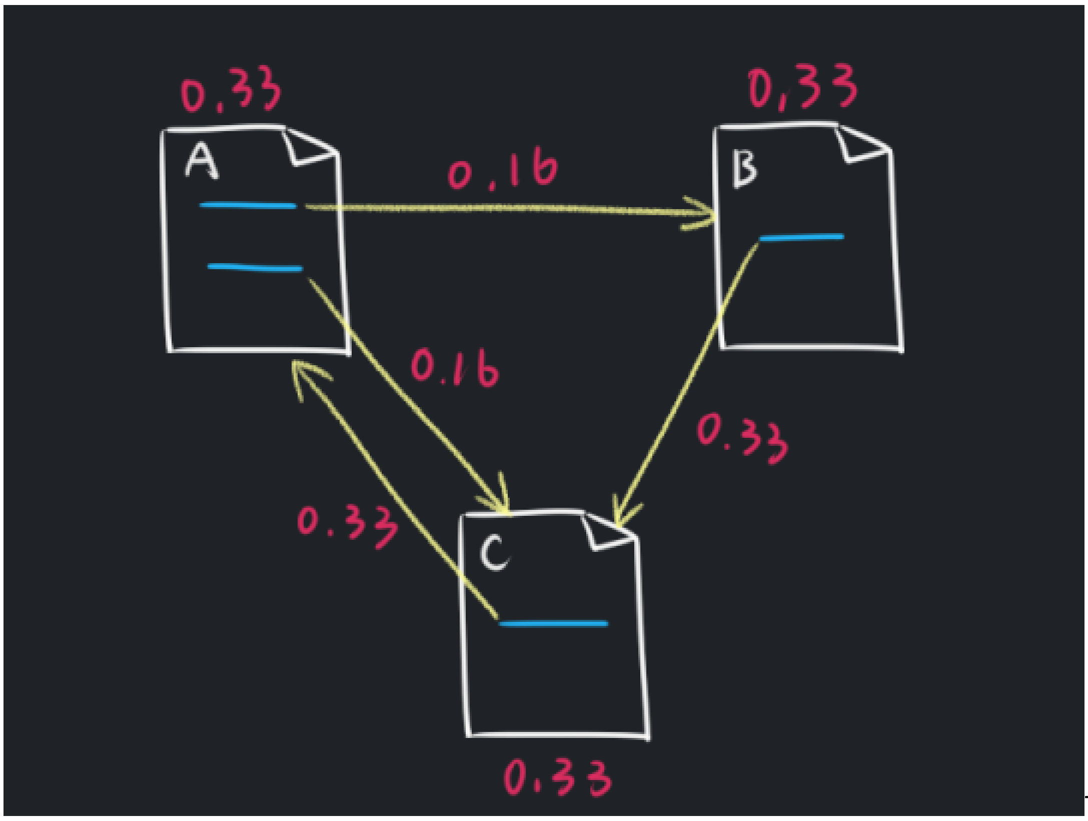
A higher PageRank indicates the website is more important. On the other hand, the PageRank of a web page is the probability of a user who visits a hyperlink on a web page randomly and finally reaches the target one. Thus, the PageRank should be an number lies between 0 and 1, and the sum of PageRank of all web pages should be 1.

The following figure is an example: Given three web pages A, B and C. Web page B and C are referenced by A, C is referenced by B and finally A is referenced by C.



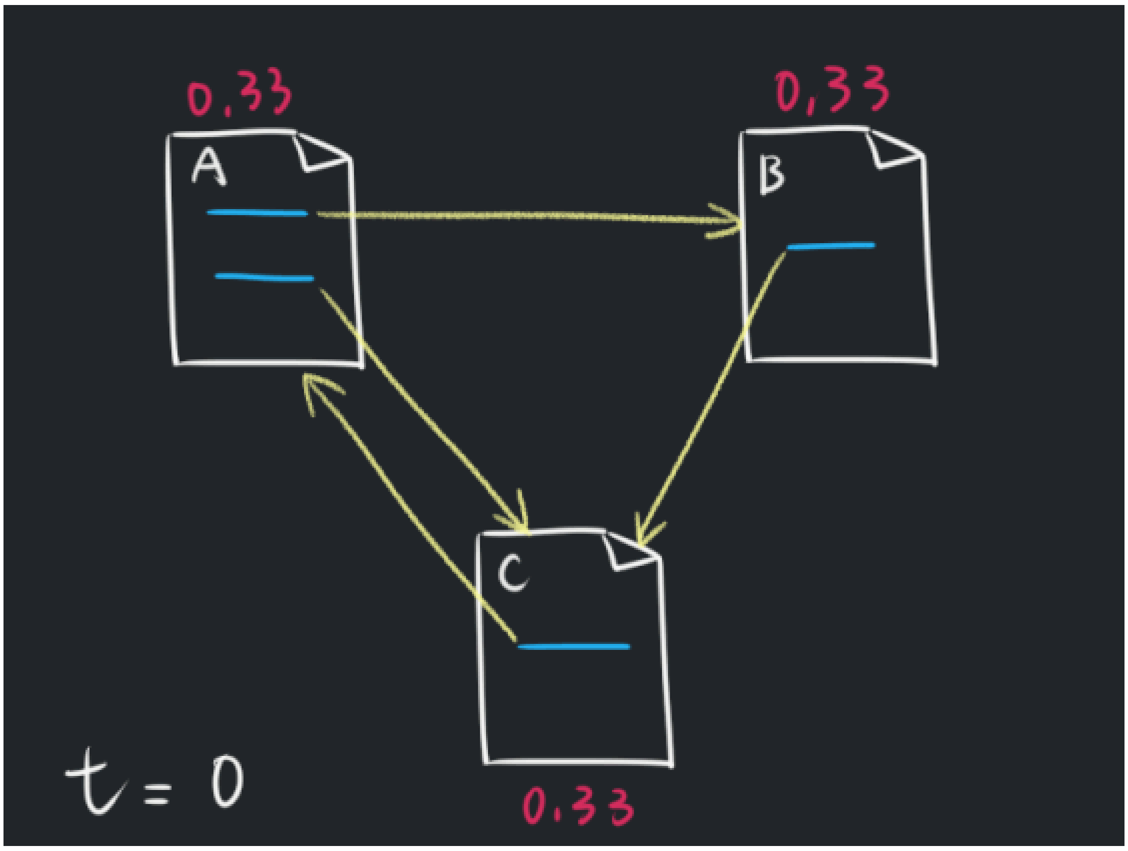
Each web page starts with an equal importance which is as known as PR. Since the sum of all PRs should be 1, the PR of each web page is initialized to 1/N, the N indicates the number of web pages, which is 3 in this case, thus the PR of each web page is initially 1/3.

Next, the PRs are updated by the number of hyperlinks. We may imagine that a web page X references another web page Y as "X votes to Y". If X only has reference to Y, than Y gets 1 point from X; but if both Y and Z are referenced by X, they should share the point voted by X. Thus, each of them gets 1/2 point. Moreover, a vote by an important web page should be concerned more valuable than a vote by a less important web page. To present this thought, the point voted by X should be multiplied by its own PR. As shown in the following figure, A votes to B and C, which makes both B and C gets 1/2 * 1/3 = 1/6 points.



The points voted to a web page are summed up to be the new PR of the web page. For example, as shown in the above figure, the new PR of web page C is 1/6 + 1/3 = 1/2. After several iterations over the steps mentioned above, the PR of a web page will converge to a fixed value which becomes the PageRank of it.

As following animation presents: after repeating the process with enough times (20 iterations in this case), the PR of each web page is 0.4, 0.2 and 0.4 respectively. The values will remain unchanged in the further iterations.



The PR of a web page u at the given time t can be written in to following equation:

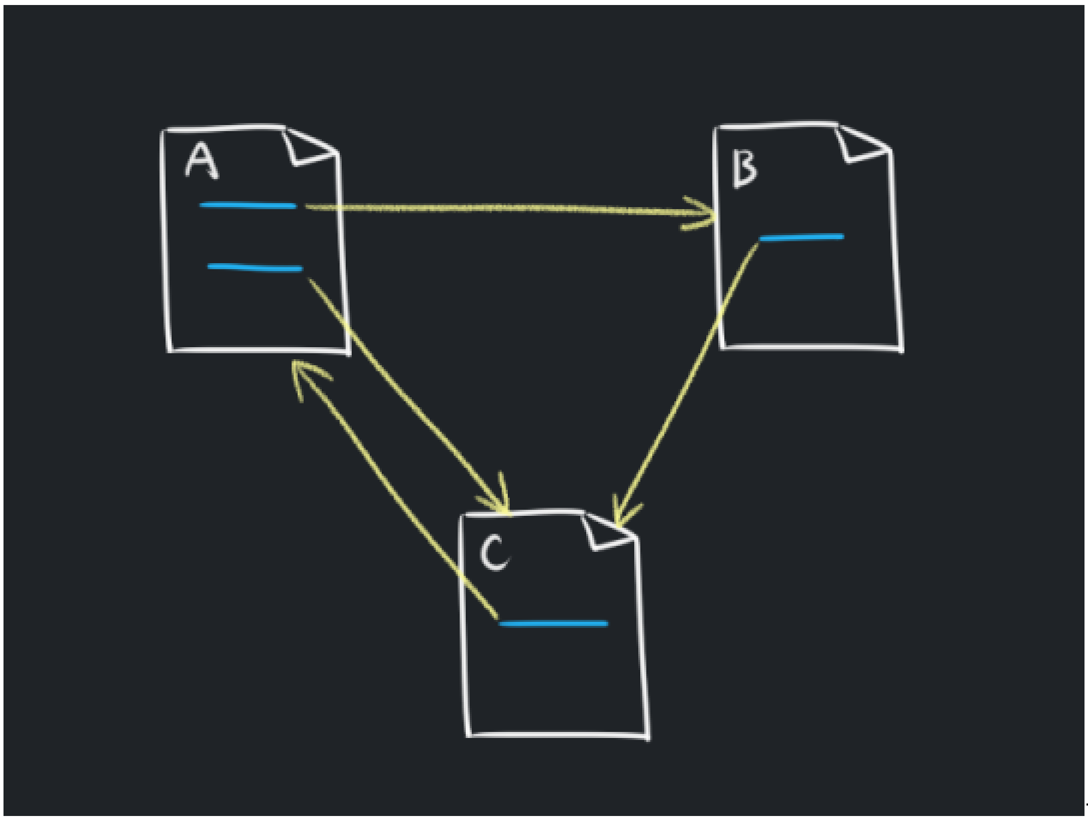
```
PR(t, u) = sum( PR(t - 1, v) / L(v) ) for all v link to u
```

where `v` indicates a web page that references `u`, and `L(v)` indicates the number of hyperlinks in the web page `v`.

Now, given a set of web pages and the hyperlinks of each web page in the set, please calculate the PR of each web page.

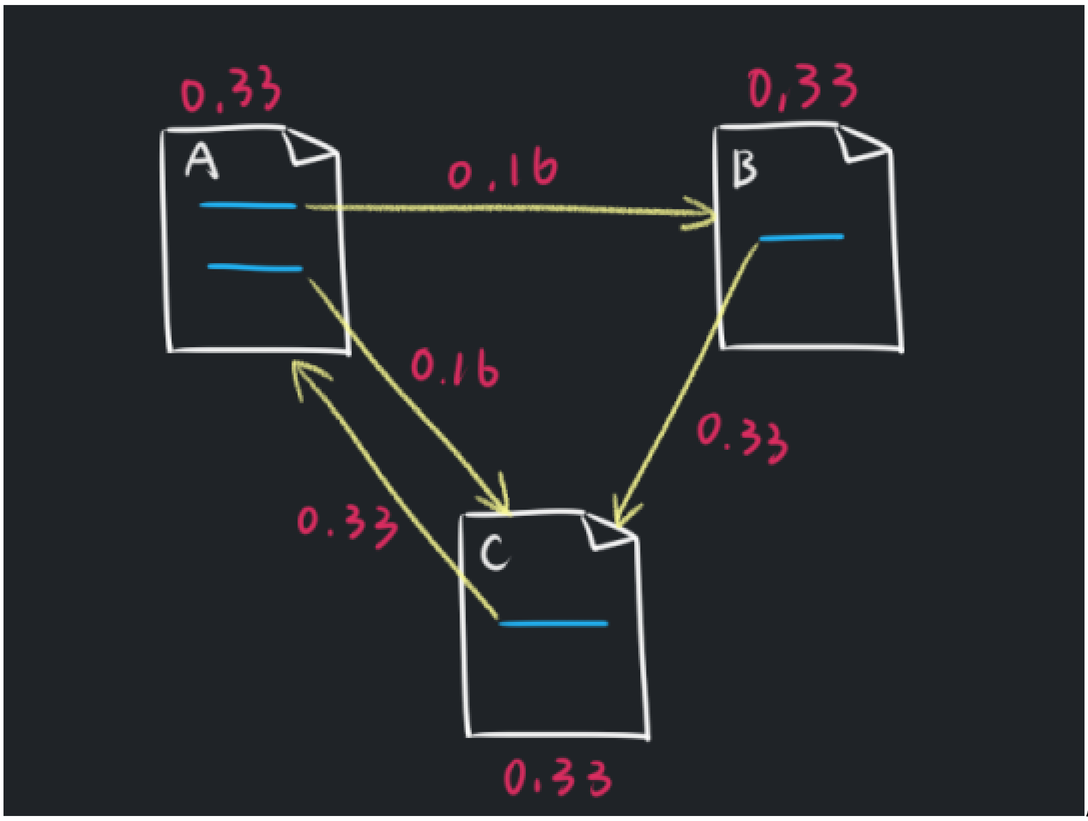
PageRank，又稱**網頁排名**，是Google公司所使用的對其搜尋引擎搜尋結果中的網頁進行排名的一種演算法，他本質上是一種以網頁之間的超連結個數和品質作為主要因素粗略地分析網頁的重要性的演算法。其基本假設是：更重要的頁面往往會有更多其他頁面擁有通向該頁面的超連結，因此我們可以透過「一個網頁有多少很重要的網頁通向他」來計算一個網頁的 PageRank。PageRank 愈高，就表示這個網頁愈重要。在實質意義上，PageRank 可以視為：「有一個使用者不斷在進入一個網站後，隨機點擊頁面上的超連結，最後會抵達某網頁的機率值」。因此，所有網頁的 PageRank 都應該介於 0 - 1 之間，並且總和應該要是 1。

我們來看一個簡單的例子：給定一個網頁的集合，該集合內有 A, B, C 三個網頁。其中 A 當中有通向 B 和 C 的超連結，B 當中有通向 C 的超連結，C 當中有通向 A 的超連結。如下圖所示：

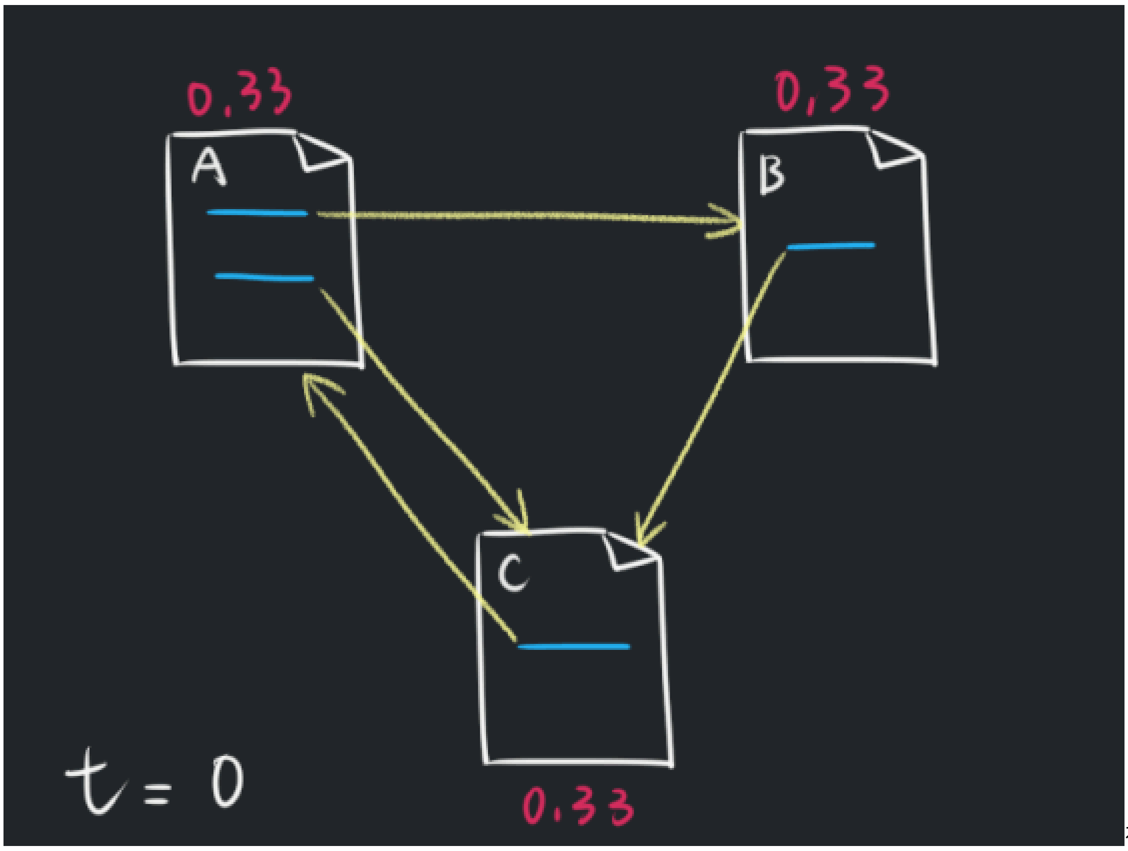


一開始所有人的重要程度都是一樣的，我們把重要程度叫作 PR，並且因為總和要是 1，所以大家的 PR 都要被初始化為 1 / N，其中 N 是這個集合的大小。在這個例子當中 N = 3，所以每個人一開始的 PR 都會是 1/3。

接著我們要透過超連結的數量來更新 PR 值。我們把「X 有一個通向 Y 的超連結」想像成是「X 投了票給 Y」，如果 X 只有投給 Y 的話，那麼 X 就是投了 1 票給 Y；但如果 X 同時投給 Y 跟 Z，那麼就視為 X 各投了 0.5 票給 Y 跟 Z。但不同重要性的網頁投的一票應該有不同的價值，如果一個網頁被一個影響力很高的網頁投了一票，他的重要性應該會比只有被一個不重要的網站投了一票還要更高的。我們必須把 X 的重要性考量進得票分數當中，由於 PR 愈高就表示一個網頁愈重要，因此我們可以把 X 投出的票數乘上他的 PR 值，來反應他的重要程度。在這個例子當中，A 一開始的 PR 值是 1/3，他分別給 B 跟 C 各投了一票，所以 B 跟 C 分別得到了 1/6 分，如下圖所示。



所有人的投票都確定之後，我們可以把一個網頁的得分給加總起來，就是該網頁的新的 PR 值。例如上圖中 C 網頁的 PR 值就是 1/6 + 1/3 = 1/2 = 0.5。透過不斷迭代這個計算過程，最後所有網頁的 PR 值會收斂到某一個數字，我們就把最後收斂到的那個數字當作該網頁的 PageRank。如下方動畫所示：



在重複這個步驟夠多次（以上圖來說是 20 次）之後，這三個網頁的 PR 值分別會是 0.4, 0.2, 0.4，接著就算再重複更多次，三個網頁的 PR 值也會維持在這個數字不動。

我們可以形式化的將上述問題描述成以下格式：在任意時間點 `t` ，一個網頁 `u` 的 PR 值 `PR(t, u)` 為

```
PR(t, u) = sum( PR(t - 1, v) / L(v) ) for all v link to u
```

其中 `v` 是一個有超連結通向網頁 `u` 的網頁，`L(v)` 是 `v` 網頁中的超連結總數（也就是他總共連出去到幾個網頁）。

現在，給定你一個網頁的集合，以及該集合中的每個網頁通向哪些網頁，請你計算出每一個頁面的 PageRank。

Input

The first line is an integer N (3 <= N <= 100), indicates the number of web page in the set. The following N lines contains several integers, which indicate the index of referenced pages. For example: the 1st line contains 1, 2, which means page 0 has hyperlinks that link to page 1 and page 2. Each line will end with -1 that indicates the termination of that line.

Output

N percentage numbers rounded to 2 decimal place that indicate PageRank of each web page. For example, if PageRank of a web page is 0.2, you should print "20.00%" for that page.

Loader Code

Your code will be judge using this program:

Sample1

Input

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

Output

```
40.00% 20.00% 40.00%
```

Sample2

Input

```
7
6 -1
0 -1
1 -1
2 -1
3 -1
```

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

Output

```
14.29% 14.29% 14.29% 14.29% 14.29% 14.29% 14.29%
```

Sample3

Input

```
10
5 -1
0 9 -1
3 -1
4 -1
4 6 -1
4 8 -1
1 2 7 -1
5 -1
6 -1
2 5 9 -1
```

Output

```
3.20% 6.40% 8.00% 8.00% 27.20% 11.20% 19.20% 6.40% 5.60% 4.80%
```

[15 Points] The coordinate to the past

Description

Matsumoto is a brilliant AI that is built and sent to the past to prevent a war between humans and AI. When on his way back to the past, he will receive a set of encoded coordinates indicating where to go.

To decode the coordinates, he has to follow the following steps:

1. split the received line of characters into two groups: characters at odd positions and characters at even positions (index starts by 0 and from left)

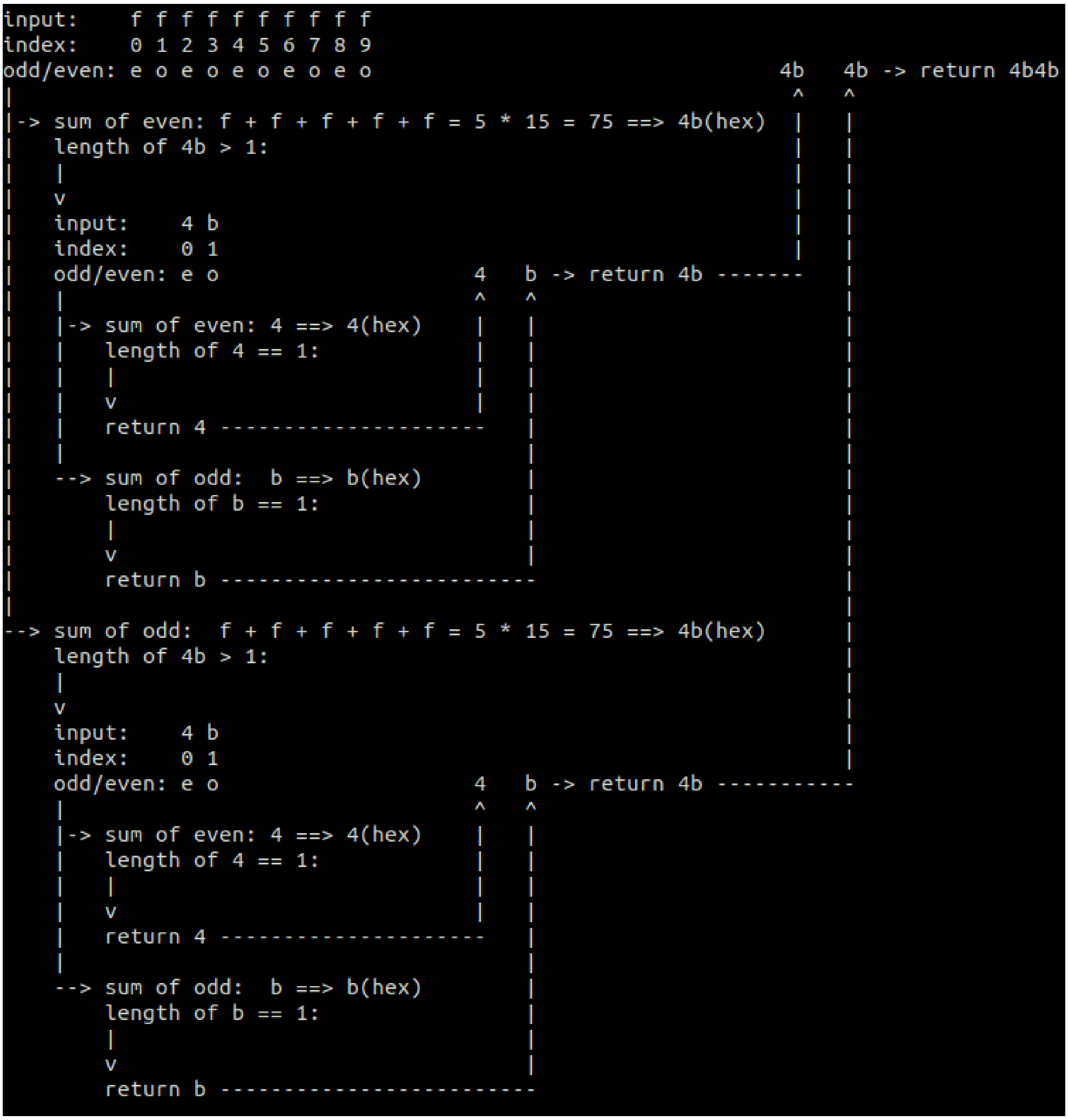
2. sum up the numbers in each group

3. write these two numbers into hexadecimal

4. process the two numbers from step 1 if the length of the number is greater than 1

5. the coordinate is the concatenate of the result of the two numbers (the even one on the left)

The following figure presents an example of the previously shown steps:



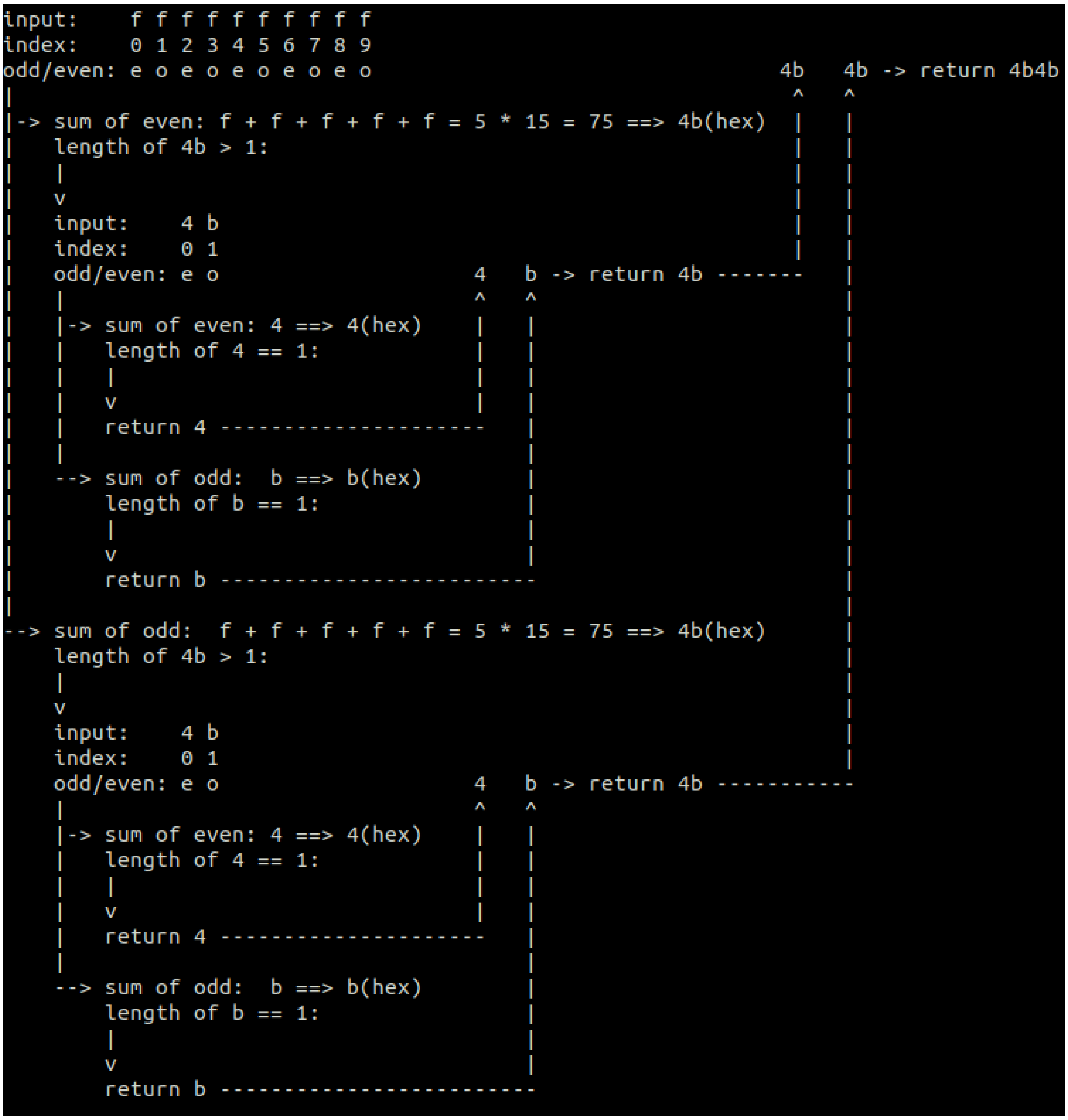
```
input:    f f f f f f f f f f
index:    0 1 2 3 4 5 6 7 8 9
odd/even: e o e o e o e o e o                                4b   4b -> return 4b4b
|                                                             ^   ^
|-> sum of even: f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)  |   |
|   length of 4b > 1:                                         |   |
|   |                                                         |   |
|   v                                                         |   |
|   input:    4 b                                             |   |
|   index:    0 1                                             |   |
|   odd/even: e o                    4   b -> return 4b -------   |
|   |                                ^   ^                        |
|   |-> sum of even: 4 ==> 4(hex)    |   |                        |
|   |   length of 4 == 1:            |   |                        |
|   |   |                            |   |                        |
|   |   v                            |   |                        |
|   |   return 4 ---------------------   |                        |
|   |                                    |                        |
|   --> sum of odd:  b ==> b(hex)        |                        |
|       length of b == 1:                |                        |
|       |                                |                        |
|       v                                |                        |
|       return b -------------------------                        |
|                                                                 |
--> sum of odd:  f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)      |
    length of 4b > 1:                                             |
    |                                                             |
    v                                                             |
    input:    4 b                                                 |
    index:    0 1                                                 |
    odd/even: e o                    4   b -> return 4b -----------
    |                                ^   ^
    |-> sum of even: 4 ==> 4(hex)    |   |
    |   length of 4 == 1:            |   |
    |   |                            |   |
    |   v                            |   |
    |   return 4 ---------------------   |
    |                                    |
    --> sum of odd:  b ==> b(hex)        |
        length of b == 1:                |
        |                                |
        v                                |
        return b -------------------------
```

Please help him to decode the coordinate.

松本是個聰明的人工智慧，創造他的人想要將他送回過去阻止一場人類與人工智慧的戰爭。在他穿越的過程中，他會收到一份經過編碼的目的地座標，想要將座標解碼需要透過以下的步驟：

1. 將收到的所有字元分成奇數位與偶數位兩組（從最左邊的開始編號，並且從0開始）
2. 分別計算兩組數字的和
3. 將兩個相加的結果以16進位表示
4. 若有和的位數大於1的，則重複上述步驟
5. 將兩組數字計算的結果串接起來（偶數組的結果在左邊，奇數組的在右邊）

下面是範例二的示意圖：



```
input:    f f f f f f f f f f
index:    0 1 2 3 4 5 6 7 8 9
odd/even: e o e o e o e o e o                                4b   4b -> return 4b4b
|                                                             ^   ^
|-> sum of even: f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)  |   |
|   length of 4b > 1:                                         |   |
|   |                                                         |   |
|   v                                                         |   |
|   input:    4 b                                             |   |
|   index:    0 1                                             |   |
|   odd/even: e o                    4   b -> return 4b -------   |
|   |                                ^   ^                        |
|   |-> sum of even: 4 ==> 4(hex)    |   |                        |
|   |   length of 4 == 1:            |   |                        |
|   |   |                            |   |                        |
|   |   v                            |   |                        |
|   |   return 4 ---------------------   |                        |
|   |                                    |                        |
|   --> sum of odd:  b ==> b(hex)        |                        |
|       length of b == 1:                |                        |
|       |                                |                        |
|       v                                |                        |
|       return b -------------------------                        |
|                                                                 |
--> sum of odd:  f + f + f + f + f = 5 * 15 = 75 ==> 4b(hex)      |
    length of 4b > 1:                                             |
    |                                                             |
    v                                                             |
    input:    4 b                                                 |
    index:    0 1                                                 |
    odd/even: e o                    4   b -> return 4b -----------
    |                                ^   ^
    |-> sum of even: 4 ==> 4(hex)    |   |
    |   length of 4 == 1:            |   |
    |   |                            |   |
    |   v                            |   |
    |   return 4 ---------------------   |
    |                                    |
    --> sum of odd:  b ==> b(hex)        |
        length of b == 1:                |
        |                                |
        v                                |
        return b -------------------------
```

請幫助松本解碼他收到的座標。

Input

A newline character terminated string indicates the encoded coordinate. The length of the string will be greater than 1 and less than 10001. Each character represents a hexadecimal, the upper case and the lower case are both possible. 一行以換行字元結尾的字串，代表經過編碼的座標。 該字串的長度介於1到10001間（不包含10001）。字串中每個字元代表一個16進位數字，且大小寫都有可能。

Output

A single line that indicates the decoded coordinate, the letter of the coordinate should be shown in lower case (if any). 解碼後的字串，若字串中有出現英文字母一律以小寫表示。

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1111111111
```

Output

```
55
```

Sample2

Input

```
ffffffffff
```

Output

```
4b4b
```

[20 Points] Paper Referencing

Description

It is usual that a paper may have many references. Probably a website, a book, or other papers.

Assume that if we want to understand a specific paper, we need to read the paper and its all references.

(If the referenced paper has other references, all of them should be read.)

In this problem, given some reference numbers,

write a program that lists all papers that should be read for the specific paper.

For the first sample testcase, given a number 5, which means there are five papers (number 1 ~ 5),

then, follow the format「referencingPaper referencedPaper1 referencedPaper2 ...」to input.

We can know that,

the 5th paper references the 1st 2nd 3rd papers the 2nd paper references the 4th paper

the 1st paper references the 2nd paper

the 3rd paper references the 4th paper

until we encounter 0, stop inputting.

Now, if we want to fully understand all these five papers, then

for the 1st paper, we should read the 1st, 2nd, 4th papers (since the 1st paper references the 2nd paper, and the 2nd paper references the 4th paper)

for the 2nd paper, we should read the 2nd, 4th papers

for the 3rd paper, we should read the 3rd, 4th papers

for the 4th paper, we should read the 4th paper

for the 5th paper, we should read the 1st, 2nd, 3rd, 4th, 5th papers

---

在一篇論文中，通常會有許多參考文獻，可能是某個網站、某本書、或是其他相關論文。

假設要完整個理解某篇論文，需要完整看完自身內容，以及其參考文獻（若類推下去仍有參考文獻，亦須將其完整看完）。

本題以此概念為出發，給予許多篇的論文編號，

請寫出一個程式，條列出透徹每篇論文所須看的所有論文編號。

以第一筆範例測資來說，給予一數字 `5`，代表有著五篇論文（編號 1 ~ 5），

接下來以「參考論文 被參考論文1 被參考論文2 ...」的格式進行輸入。

因此可以解讀出以下訊息：

第 5 篇論文參考了第 1、2、3 篇論文

第 2 篇論文參考了第 4 篇論文

第 1 篇論文參考了第 2 篇論文

第 3 篇論文參考了第 4 篇論文

直到遇到 0，代表輸入結束。

那麼若要透徹這五篇論文，則

第 1 篇論文應閱讀完第 1、2、4 篇論文（因為第 1 篇論文參考了第 2 篇論文，而第 2 篇論文又參考了第 4 篇論文）

第 2 篇論文應閱讀完第 2、4 篇論文

第 3 篇論文應閱讀完第 3、4 篇論文

第 4 篇論文應閱讀完第 4 篇論文

第 5 篇論文應閱讀完第 1、2、3、4、5 篇論文。

Input

The first line is an integer N, and it means there are N papers (number 1 ~ N). And 1 <= N <= 100. Then follow the format「referencingPaper referencedPaper1 referencedPaper2 ...」to input until encounter 0. Notice that the referencing paper number on each row won't be ordered. You don't need to consider the cycle. (If the 1st paper references the 3rd paper, and the 3rd paper references the 1st paper, well, it sounds weird!) 一開始輸入一數字 N，代表有 N 篇論文，編號為 1 ~ N，且 1 <= N <= 100。接者以「參考論文 被參考論文1 被參考論文2 ...」的格式進行輸入， 直到讀到 0 代表輸入結束。 注意，每列的參考論文編號並不會按照順序， 且不必考慮 cycle 的狀況（如果說第 1 篇論文參考第 3 篇論文，而第 3 篇論文卻又參考了第 1 篇論文，聽起來超怪的對吧！）。

Output

Print each paper number in ascending order on each row. Then follow the format「referencingPaper -> readPaper1 readPaper2 ...」. Notice that the readPaperN should be printed in ascending order. 依照每個論文號碼順序輸出， 並以「參考論文 -> 需閱讀的論文1 需閱讀的論文2 ...」的格式回答。 注意，須閱讀的論文N 必須由小到大排序。

Loader Code

Your code will be judge using this program:

Sample1

Input

```
5
5 1 2 3
2 4
1 2
3 4
0
```

Output

```
1 -> 1 2 4
2 -> 2 4
3 -> 3 4
4 -> 4
5 -> 1 2 3 4 5
```

Sample2

Input

```
8
6 2 4 5
4 1 7
3 1 5 8
0
```

Output

```
1 -> 1
2 -> 2
3 -> 1 3 5 8
4 -> 1 4 7
5 -> 5
6 -> 1 2 4 5 6 7
7 -> 7
8 -> 8
```

[20 Points] PK high school transfer student

Description

In PK high school, there is a transfer student. **The student number is sorted by their name.**

Following are the student number sorting rules:

- Sorted by the letter of name, if the letter is closer to A, the student number is smaller. (For instance, the student number of `Chloe` is smaller than that of `Sam` because compare with `C` and `S`, `C` is closer to `A`)

- It is no different between lowercase and uppercase (ex. `AMY` and `amy` are the same)

- In this case, `Sam` and `Samuel`, the letters of `Sam` is the same as the first three letters of `Samuel`, but the length of `Sam` is shorter than that of `Samuel`, `Sam`'s student number is smaller than `Samuel`.

Here is an example.

There are two students in class. The name of those students are `nendou` and `saiki`.

The transfer student's name is `saiko`.

Let's compare `nendou` and `saiko` first. Because the order of `n` is smaller than `s`, the student number of `nendou` is smaller than `saiko`.

Next, compare `saiki` and `saiko`. Because the first four letter of their name are the same, compare the 5th letter. Because the order of `i` is smaller than `o`, the student number of `saiki` is smaller than `saiko`.

In this case, transfer student `saiko` 's student number is 3.

Please help transfer student to find his/her student number.

p.s The transfer student's name won't be the same as the student's name.

呀咧呀咧，新學期PK學園又來了個新轉學生，在這裡所有的學生 **座號都是使用名字拼音排序** 的。

排序的規則為

- 從名字拼音的開始排序，比較接近 `a` 的座號會比較前面
- 大小寫字母沒有分別 (ex. `AMY` 與 `amy` 視作完全相同)
- 如果剛好出現轉學生全名與班上學生某姓名的前面字母完全相等，或班上某學生全名與轉學生姓名前面字母完全相等，則名字較短的座號會比較前面。 (ex. `Sam` 與 `Samuel`, `Sam` 的座號會比較小)

現在來舉個例子：

班上有兩個學生，分別是 `nendou` 與 `saiki`,

轉學生的名字叫 `saiko`。

先來比較 `nendou` 與 `saiko` ，按照英文 26 個字母的排序，`n` 比 `s` 來得更前面，所以 `nendou` 的座號在 `saiko` 之前。

再比較 `saiki` 與 `saiko` ，由於前四個字母皆為 `saik` ，比較第五個字母，由於 `i` 在 `o` 之前，所以 `saiki` 的座號也在 `saiko` 之前。

在這個範例裡，新轉學生 `saiko` 的座號就是 3。

每個學期都會出現新的轉學生，可以請你幫轉學生找到他的座號嗎？

p.s 不會出現班上同學與轉學生撞名的狀況

Input

First line is a number (N) represent for how many students are in the class. Second line is transfer student's information. The first number (n) represents for the length of the student's name, following is the student's name. Following N lines are the origin student's information. The first number (n) represents for the length of the student's name, following is the student's name. 1 <= N <= 30 1 <= n <= 16 The students' names are composed of a ~ z and A ~ Z ----------------------- 第 1 行的輸入 N 代表班上目前有幾名學生 第 2 行的輸入是轉學生資訊，有兩個欄位，第一個 n 表示轉學生名字的長度，後面接續著轉學生的名字 後面 N 行是班上學生的資訊，第一個 n 表示學生名字長度，後面接續著班上學生的名字（注意：這 N 行之間並不是已排序好的學生資訊） 1 <= N <= 30 1 <= n <= 16 學生名稱由小寫英文字母 a ~ z 及大寫英文字母 A ~ Z 組成

Output

Transfer student's student number (student number starts from 1) ---------------------- 輸出為轉學生的座號 (座號從 1 開始)

Loader Code

Your code will be judge using this program:

Sample1

Input

```
2
5 saiko
6 nendou
5 saiki
```

Output

```
3
```

Sample2

Input

```
5
3 pat
5 patsy
7 patrick
5 percy
8 philemen
8 penelope
```

Output

```
1
```

[25 Points] Railway Timetable

Description

Ariel is traveling around Taiwan by scooter. Unfortunately, her scooter broke down when she arrived in Taipei. Therefore, Ariel plans to take the train from Taipei to Tainan.
There are `n` trains. Each train has different number `x`, departure time `HH:MM` (0<=`HH`<=23, 0<=`MM`<=59) and running time `y` minutes. Current time is `hh:mm`, please tell Ariel which train that she can take arrives in Tainan earliest.

Note 1. Ariel only takes the train after `hh:mm`(including `hh:mm`)

Note 2. Timetable is sorted from 00:00 to 23:59.

Note 3. Probably train A departs earlier than train B, but train A arrives later than train B ; train A departs later than train B, but train A arrives earlier than train B. (referring to Sample 2)
Note 4. Ariel must can take the train to Tainan.
Note 5. Trains arrive in Tainan at different times. That means, there is only one answer.

Ariel 以台南為起點騎車環島，騎車到了台北後，車子拋錨了，只好改搭火車回家，當天的火車時刻表上共有 `n` 班火車，每班火車都有不相同的編號 `x`，發車時間為 `HH:MM` (0<=`HH`<=23, 0<=`MM`<=59)，需要花費 `y` 分鐘才能抵達台南，此時時間為 `hh:mm`，請問 Ariel 應該要搭哪一班火車才能最早抵達台南?

註1. Ariel 只能搭 `hh:mm` 以後的火車 (包含 `hh:mm`)

註2. 火車時刻表上的發車時間是從 00:00 開始排序下來的

註3. 有可能比較早發車的火車比較晚抵達台南;比較晚發車的火車比較早抵達台南 **(可參考 sample 2)**

註4. 一定有火車可以搭回台南

註5. 火車抵達台南的時間皆不同，即答案只會有一個\

Input

First line, a sequence hh:mm represents current time. (hh and mm are integers, 0<=hh<=23, 0<=mm<=59) Second line, an integer n represents the number of trains.(1<=n<=30) Then, there are n lines, an integer x , a sequence HH:MM and an integer y represent the train's number, departure time and running time, respectively. (0<=x<=9999, 0<=HH<=23, 0<=MM<=59, 60<=y<=180)

Output

Output the train's number that Ariel can arrive in Tainan earliest.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
02:00
8
100 01:11 100
101 01:59 102
111 02:03 100
200 02:59 110
201 03:11 100
211 05:02 90
```

```
300 07:12 80
301 10:22 60
```

Output

```
111
```

Sample2

Input

```
06:00
10
10 01:00 100
1 01:50 102
6749 03:07 200
123 05:59 300
122 06:00 200
2333 06:05 180
777 07:12 100
3012 10:22 300
331 13:20 100
599 19:22 50
```

Output

```
777
```

[25 Points] Trithemius Cipher Encoder

Description

Do you remember Caesar cipher? One similar technique is **Trithemius cipher**, it is a successive shift cipher. In the case of Caesar cipher, each character in the plaintext message is shifted with a fixed offset `k`. As for **Trithemius cipher**, a successive shift (it could be ascending or descending, each time it shift 1) and the offset `k` is added to each character.

For example: `k=1` , ascending, and the message we want to encode is `TAs are so H4ND5OME`. The offset of each character is:

1. `T A s a r e s o H N D O M E`
2. `1 2 3 4 5 6 7 8 9 10 11 12 13 14`

The cipher text becomes `UCv ewk zw Q4XO5AZS`.

Write a program that prints out the result of Trithemius cipher for the given input (described in Input Format).

Note: Transform only **english alphabet (case-preserving)** and preserve other characters such as Arabic numerals or punctuation. The successive shift only occurs on english alphabet.

還記得凱薩密碼嗎？有個類似的加密方式是**特里特米烏斯密碼**。在凱薩密碼中，每個字元都會被平移一個固定的值`k`，而**特里特米烏斯密碼密碼**則會額外多加一個連續的偏移值(會是遞增或遞減，每次移動1)。

舉例來說， `k=1`、遞增的狀況下，我們想加密`TAs are so H4ND5OME`，而每個字元的偏移量為：

1. `T A s a r e s o H N D O M E`
2. `1 2 3 4 5 6 7 8 9 10 11 12 13 14`

加密文字會是`UCv ewk zw Q4XO5AZS`.

請撰寫一支程式來加密給定的訊息。

註：只需轉換英文字母(需保留大小寫)並保留其他字元例如阿拉伯數字、標點符號等等。連續偏移量只會在遇到英文字母時增減。

Input

The first line is an integer k, where -2147483648 <= k <= 2147483647. The second line contains a character 'a' or 'd' which indicates that it is ascending or descending. The third line is the message that you need to encode. Each line is ended with a newline character.

Output

The cipher text.

Loader Code

Your code will be judge using this program:

Sample1

Input

```
1
a
TAs are so H4ND5OME
```

Output

```
UCv ewk zw Q4XO5AZS
```

Sample2

Input

```
4
d
PD1 is sooooo awesome!!
```

Output

```
TG1 kt snmlkj upwjebs!!
```