Week4 (1006)

鍾健雄

UVa490：Rotating Sentences

In "Rotating Sentences," you are asked to rotate a series of input sentences 90 degrees clockwise. So instead of displaying the input sentences from left to right and top to bottom, your program will display them from top to bottom and right to left.

Input

As input to your program, you will be given a maximum of 100 sentences, each not exceeding 100 characters long. Legal characters include: newline, space, any punctuation characters, digits, and lower case or upper case English letters. (NOTE: Tabs are not legal characters.)

Output

The output of the program should have the last sentence printed out vertically in the leftmost column; the first sentence of the input would subsequently end up at the rightmost column.

“Rotating Sentences,” (22)

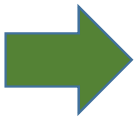
from top to bottom (19)

last sentence printed out (26)

lineNo = 3
max = 26

“	R	o	t	a	t	i	n	g		S	e	n	t	e	n	c	e	s	,	“	\0	
f	r	o	m		t	o	p		t	o		b	o	t	t	o	m	\0				
l	a	s	t		s	e	n	t	e	n	c	e		p	r	i	n	t	e	d		o	u	t	\0

“	R	o	t	a	t	i	n	g		S	e	n	t	e	n	c	e	s	,	“		\0
f	r	o	m		t	o	p		t	o		b	o	t	t	o	m					\0
l	a	s	t		s	e	n	t	e	n	c	e		p	r	i	n	t	e	d		o	u	t	\0



l	f	“
a	r	R
s	o	o
t	M	t
		a
s	t	t
e	o	i
n	p	n
t		g
e	t	
n	o	s
c		e
e	b	n
	o	t
p	t	e
r	t	n
i	o	c
n	m	E
t		S
e		,
d		“
o		
u		
t		

- 定義str[100][100]，定義lineNo紀錄有多少個字串
- 讀取字串
- 找出最長的字串長度(max)
- 對每一個字串，不到max長度的位置補「空格」
- 將m x n的字串陣列轉90度(m=lineNo, n=max)
  - => n x m的字串陣列

Decode the Mad man (UVA10222)

Once in BUET, an old professor had gone completely mad. He started talking with some peculiar words. Nobody could realize his speech and lectures. Finally the BUET authority fall in great trouble. There was no way left to keep that man working in university. Suddenly a student (definitely he was a registered author at UVA ACM Chapter and hold a good rank on 24 hour-Online Judge) created a program that was able to decode that professor's speech. After his invention, everyone got comfort again and that old teacher started his everyday works as before.

So, if you ever visit BUET and see a teacher talking with a microphone, which is connected to a IBM computer equipped with a voice recognition software and students are taking their lecture from the computer screen, don't get thundered! Because now your job is to write the same program which can decode that mad teacher's speech!

Input

The input file will contain only one test case i.e. the encoded message.

The test case consists of one or more words.

Output

For the given test case, print a line containing the decoded words. However, it is not so hard task to replace each letter or punctuation symbol by the two immediately to its left alphabet on your standard keyboard.

Sample Input

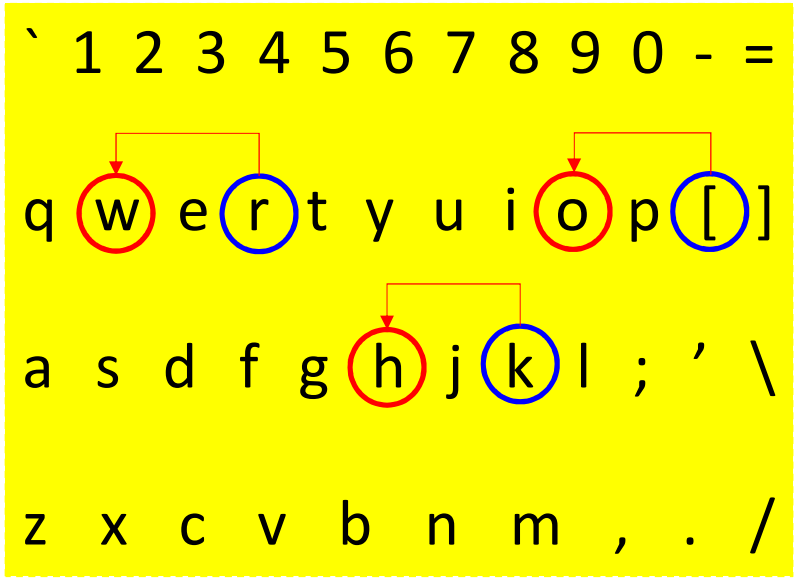
```
k[r dyt I[o
```

Sample Output

```
how are you
```

- 老教授口述的訊息往往有偏差，例如
  - k[r dyt I[o => how are you
- 要將上述訊息轉為正常訊息的方法是
  - 每個字元的鍵盤位置項左移二位
  - 注意：老教授的訊息有「符號」、「大小寫字母」
- 處理過程
  - 空格不需轉換
  - 先將大寫字母轉換為「小寫」
  - 將鍵盤符號字母儲存於一個陣列中

```
Char keys[4][15] = {
  “`1234567890-=“,
  “qwertyuiop[]”,
  “asdfghjkl;’\”,
  “zxcvbnm,./”}
```



Note: I => i => y

TeX

- TeX，是一個由美國電腦教授 Donald Ervin Knuth 編寫的排版軟體。TeX是一款自由軟體。它在學術界特別是數學、物理學和電腦科學界十分流行。普遍認為是一個優秀的排版工具，尤其是對於複雜數學公式的處理。利用LaTeX等終端軟體，TeX就能夠排版出精美的文字以幫助人們辨認和尋找。
- Donald Ervin Knuth 著名電腦科學家，史丹福大學電腦系榮譽退休教授。為現代電腦科學的先驅人物，創造了演算法分析的領域，在數個理論計算機科學的分支做出基石貢獻。在電腦科學及數學領域發表了多部廣泛影響論文和著作。1974年圖靈獎得主。
  - 圖靈獎是ACM於1966年設立的獎項，專門獎勵對電腦事業作出重要貢獻的個人。圖靈獎紀念現代電腦科學的奠基者、電腦科學的先驅、英國科學家、曼徹斯特大學教授艾倫·圖靈（A.M. Turing）。獲獎者必須是在電腦領域具有持久而重大的先進性的技術貢獻。大多數獲獎者是電腦科學家。是電腦界最負盛名的獎項，有「電腦界諾貝爾獎」之稱。
- 高德納所寫的《電腦程式設計藝術》（*The Art of Computer Programming*）是計算機科學界最高度敬重參考書籍之一。也是排版軟體TEX和字型設計系統Metafont的發明人。

TEX Quotes (UVA272)

- “To be or not to be,” quoth the Bard, “that
  is the question”.
  The programming contestant replied: “I must disagree.
  To `C' or not to `C', that is The Question!”
- ``To be or not to be,'' quoth the Bard, ``that
  is the question''.
  The programming contestant replied: ``I must disagree.
  To `C' or not to `C', that is The Question!''
- 將「“」代換為「``」，將「”」代換為「''」

What is the probability? (UVA10056)

Probability has always been an integrated part of computer algorithms. Where the deterministic algorithms have failed to solve a problem in short time, probabilistic algorithms have come to the rescue. In this problem we are not dealing with any probabilistic algorithm. We will just try to determine the winning probability of a certain player.

A game is played by throwing a dice like thing (it should not be assumed that it has six sides like an ordinary dice). If a certain event occurs when a player throws the dice (such as getting a 3, getting green side on top or whatever) he is declared the winner. There can be N such player. So the first player will throw the dice, then the second and at last the N-th player and again the first player and so on. When a player gets the desired event he or she is declared winner and playing stops. You will have to determine the winning probability of one (The I-th) of these players.

Input

Input will contain an integer S ($S \leq 1000$) at first, which indicates how many sets of inputs are there. The next S lines will contain S sets of inputs. Each line contain an integer N ($N \leq 1000$) which denotes the number players, a floating point number p which indicates the probability of the happening of a successful event in a single throw (If success means getting 3 then p is the probability of getting 3 in a single throw. For a normal dice the probability of getting 3 is 1/6), and I ($I \leq N$) the serial of the player whose winning probability is to be determined (Serial no varies from 1 to N). You can assume that no invalid probability (p) value will be given as input.

Output

For each set of input, output in a single line the probability of the I-th player to win. The output floating point number will always have four digits after the decimal point as shown in the sample output.

Sample Input

```
2
2 0.166666 1
2 0.166666 2
```

Sample Output

```
0.5455
0.4545
```

- 有N個人輪流擲骰子，擲到3的人獲勝
- 骰子是公平的，擲到3的機率為1/6，定為 p
  - 失敗的機率 q = 1-p
- N個人中第 i 人獲勝的機率為何？

- 過程
  - 第1回合：
    - 第1人獲勝機率：P
    - 第2人獲勝機率：$q*P$
    - 第3人獲勝機率：q^2*P
    - …
    - 第k人獲勝機率：$q^{k-1}*P$
  - 第2回合：
    - 第1人獲勝機率：$q^N P$
    - 第2人獲勝機率：q^N*q*P
    - 第3人獲勝機率：q^N*q^2*P
    - …
    - 第k人獲勝機率：$q^N*q^{k-1}*P$
  - 第R回合：
    - 第1人獲勝機率：$q^{N(R-1)}P$
    - 第2人獲勝機率：$q^{N(R-1)}*q*P$
    - 第3人獲勝機率：$q^{N(R-1)}*q^2*P$
    - …
    - 第k人獲勝機率：$q^{N(R-1)}*q^{k-1}*P$

$$S = a + ar + ar^2 + ar^3 + \cdots \quad (1)$$
$$rS = ar + ar^2 + ar^3 + \cdots \quad (2)$$

(1)-(2) => (1-r)S = a

S = a/(1-r)

S: 等比級數

a: 首項

r: 公比

第k人獲勝機率:

$q^{k-1}*P+q^N*q^{k-1}*P+\ldots+q^{N(R-1)}*q^{k-1}*P$

首項：$q^{k-1}*P$

公比：q^N

$S = q^{k-1}*P/(1-q^N)$