字元與字串

資料來源： http://www.tcgs.tc.edu.tw/~sagit/cpp/q5.htm

一、字元與 ASCII 碼

一個英文字母、數字或其他的符號，我們稱它為字元。要表示一個字元，我們可以用一對單引號 ' 把該字元夾起來，例如：

```
char c='a';
```

而要輸入一個字元，有下列四種方式：

```
char c;
scanf("%c", &c); // C
c=getc(stdin); // C
cin >> c; // C++
cin.get(c); // C++
```

上面的 getc() 函數可以從檔案讀取一個字元，而 stdin 則是標準輸入(Standard Input)，也就是從鍵盤讀取。(可被重新導向)

而要印出一個字元，則可以用下面的方式：

```
char c='a';
printf("%c", c); // C
putc(c, stdout); // C
cout << c; // C++
cout.put(c); // C++
```

上面的 stdout 同樣是標準輸出(Standard Output)，也就是從螢幕輸出。(可被重新導向)

事實上，字元在電腦中是以一個八位元的整數來儲存(即 1 Byte)，而這個符號與數字的對應關係我們稱為 ASCII 碼 (American Standard Code for Information Interchange)，例如：

數字	符號	數字	符號	數字	符號
48	0	65	A	97	a
49	1	66	B	98	b
50	2	67	C	99	c
51	3	68	D	100	d
52	4	69	E	101	e
…	…	…	…	…	…

也就是說，其實字元也是一個數字，而大小寫字母相差 32 (小寫字母比較大)。而我們也可以把字元拿來做加減乘除等四則運算。例如

```
char c='a';
c+=3;
cout << c;
```

上面的例子會印出字元 d。至於下面的程式可以印出 A 到 Z 及它們的 ASCII 碼值：

```
char c;
for(c='A'; c<='Z'; c++) {
   cout << c << " " << int(c) << endl; // C++
   printf("%c %d\n", c, c); // C
}
```

除此之外，C/C++ 裡還有一些特殊字元，通常以反斜線 \ 開頭：

'\0' — 空字元，用於字串的結束，它的值也就是 0
'\n' — New Line，換行符號
'\r' — Carriage Return，回歸鍵(即 Enter 鍵)
'\t' — Tab，跳格
'\b' — Backspace，倒退鍵
'\a' — Bell，嗶一聲
'\\' — 反斜線 \
'\'' — 單引號 '
'\"' — 雙引號 "

至於要判斷一個字元是數字或大小寫字母，除了用 c>='A' && c<='Z' 之類的寫法外，C/C++ 提供了一些字元處理的函數，不過要使用它們要先 #include <ctype.h>，這些函數如下：

isdigit(c) — 判斷 c 是否為數字
islower(c) — 判斷 c 是否為小寫字母
isupper(c) — 判斷 c 是否為大寫字母
isalpha(c) — 判斷 c 是否為字母
isalnum(c) — 判斷 c 是否為字母或數字

二、字串

接下來，我們討論到字串，字串就是一段文字，我們可以用一對雙引號 " 把該段文字夾起來。由於 C 中並沒有字串的變數型態(PS. C++ 中有 string 的字串型態)，而是用字元的陣列來儲存一個字串，我們先宣告字串的長度：

```
#define LEN 80
```

再來宣告字串變數：

```
char s[LEN], t[LEN]="TCGS";
```

上面的例子宣告了 s、t 為字串，長度為 LEN 個字元。我們在宣告字串的時候要注意它的長度，以免位數不夠造成程式錯誤。事實上，每個字串後面都有一個 '\0' 的字元，也就是說，上面的 "TCGS" 字串，事實上總共用了 5 Bytes，這一點要特別注意。我們用下面的表格說明：

t[0]	t[1]	t[2]	t[3]	t[4]	t[5]	t[6]	t[7]	t[8]	t[9]
'T'	'C'	'G'	'S'	'\0'	?	?	?	?	?

而我們要輸入和輸出一個字串可以用下面的方式：

```
char s[LEN];
scanf("%s", s); // C
printf("%s\n", s); // C
cin >> s; // C++
cout << s << endl; // C++
```

比較特別的是，因為 s 本身就是字串的位址，所以使用 scanf() 讀取字串時在 s 前面不用加上 & 符號。

要注意的是，上面的方式讀到空白就會停止，例如我們輸入的是 "Hello, TCGS!" 則只有 "Hello," 被讀進字串 s 裡。而要避免這種情況，我們可以使用整行讀取的方式：

```
fgets(s, LEN, stdin); // C
cin.getline(s, LEN); // C++
```

上面的方式， fgets() 會保留後面的換行符號 ('\n')，而 cin.getline() 則不會，這點要注意。

因為字串是字元的陣列，所以如果我們要取得某一個字元，可以用 s[i] 的方式取得，例如下面的程式可以輸入一行文字後把它從後面反過來輸出：

```
char s[LEN];
int i, len;
cin.getline(s, LEN);
```

```
len=strlen(s);
for(i=len-1; i>=0; i--) cout << s[i];
cout << endl;
```

上面我們用了一個 strlen() 的函數，它的功能是傳回該字串的長度。要使用這個函數要先 #include <string.h>。而 string.h 裡還有許多常用的字串函數，如下：

strlen(s1) — 傳回 s1 的長度(不含 '\0' 字元)

strcpy(s1, s2) — 將 s2 的內容複製到 s1

strcmp(s1, s2) — 比較 s1、s2 的內容，如果相等傳回 0

strcat(s1, s2) — 將 s2 串接到 s1 後面

strstr(s1, s2) — 傳回 s2 字串在 s1 字串中第一次出現的位置

strrev(s1) — 將 s1 字串倒置 (非標準)

三、編碼問題

在第二次世界大戰中，德軍的通訊編碼被美國破解，以致於機密被美國竊聽而慘敗。德軍的編碼規則(假)如下：將訊息每個字母往後推兩位再傳出去，例如 A→C、B→D，而後面的 Y→A、Z→B，所有的訊息都是大寫字母，而空白以及其他的符號則維持不變。例如 "I LOVE TCGS." 則會變成 "K NQXG VEIU."。

假設你是美軍的情報員，收到德軍編碼過的訊息，要快速地將它解碼回原來的訊息，你可以完成這個任務嗎？

輸入 1：K NQXG VEIU.
輸出 1：I LOVE TCGS.
輸入 2：PRUE 2005 KU EQOKPI UQQP.
輸出 2：NPSC 2005 IS COMING SOON.

我們可以一個字元一個字元讀進來，如果是大寫字母，則把它減 2，如果減 2 之後比 A 還小，則加上 26，然後把它印出來。如果不是字母，則直接輸出即可。程式如下：

```
char c;
while (1) {
   cin.get(c);
   if( cin.fail() ) break;
   if( (c>='A') && (c<='Z') ) {
      c-=2;
      if( c<'A' ) c+=26;
   }
   cout << c;
}
```

輸入結束請按 Ctrl+Z 再按 Enter。上面使用 cin.fail() 判斷是不是讀到結束了。如果是使用 C 的話，可以改成：

```
c = getc(stdin);
if( c==EOF ) break;
```

你也可以把上面的程式改成輸入原本的文字，而輸出編碼過的文字。

四、練習題

1. 基礎：Q458 — The Decoder
2. 基礎：Q10082 — WERTYU
3. 基礎：Q10222 — Decode the Mad man
4. 基礎：Q272 — TeX Quotes
5. 綜合：Q494 — Kindergarten Counting Game
6. 綜合：Q10340 — All in All

458 - The Decoder

Write a complete program that will correctly decode a set of characters into a valid message. Your program should read a given file of a simple coded set of characters and print the exact message that the characters contain. The code key for this simple coding is a one for one character substitution based upon a *single arithmetic manipulation* of the printable portion of the ASCII character set.

Input and Output

For example: with the input file that contains:

1JKJ'pz'{ol'{yhklthyr'vm'{ol'Jvu{yvs'Kh{h'Jvywvyh{pvu5
1PIT'pz'h'{yhklthyr'vm'{ol'Pu{lyuh{pvuhs'I|zpulzz'Thjopul'Jvywvyh{pvu5
1KLJ'pz'{ol'{yhklthyr'vm'{ol'Kpnp{hs'Lx|pwtlu{'Jvywvyh{pvu5

your program should print the message:

*CDC is the trademark of the Control Data Corporation.
*IBM is a trademark of the International Business Machine Corporation.
*DEC is the trademark of the Digital Equipment Corporation.

Your program should accept all sets of characters that use the same encoding scheme and should print the actual message of each set of characters.

Sample Input

1JKJ'pz'{ol'{yhklthyr'vm'{ol'Jvu{yvs'Kh{h'Jvywvyh{pvu5
1PIT'pz'h'{yhklthyr'vm'{ol'Pu{lyuh{pvuhs'I|zpulzz'Thjopul'Jvywvyh{pvu5
1KLJ'pz'{ol'{yhklthyr'vm'{ol'Kpnp{hs'Lx|pwtlu{'Jvywvyh{pvu5

Sample Output

*CDC is the trademark of the Control Data Corporation.
*IBM is a trademark of the International Business Machine Corporation.
*DEC is the trademark of the Digital Equipment Corporation.

MyNote: **Finished Date:**

說明：類似上面的編碼，但是除了換行（'\n'）維持不變外，其他文字包含空白以及符號都要加減某個數，至於那個數是多少，請自行從上面的輸入、輸出推算。

10082 - Problem C: WERTYU

A common typing error is to place the hands on the keyboard one row to the right of the correct position. So "Q" is typed as "W" and "J" is typed as "K" and so on. You are to decode a message typed in this manner.

Input consists of several lines of text. Each line may contain digits, spaces, upper case letters (except Q, A, Z), or punctuation shown above [except back-quote (`)]. Keys labelled with words [*Tab, BackSp, Control,* etc.] are not represented in the input. You are to replace each letter or punction symbol by the one immediately to its left on the QWERTY keyboard shown above. Spaces in the input should be echoed in the output.

Sample Input

O S, GOMR YPFSU/

Output for Sample Input

I AM FINE TODAY.

MyNote~ **Finished Date:**

說明：有一個奇怪的鍵盤，按下任何一個鍵，都會輸出它右邊那個鍵的符號，例如按下 W 則印出 E，按下 P 則印出 [。不過 Enter、空白鍵、倒退鍵、Tab、Ctrl、Alt、Shift 這些鍵是不受影響的。現在請你由印出來的文字，推算回去原本輸入的文字。(PS. 使用 switch)

10222 - Decode the Mad man

The Problem

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!

The Input

The input file will contain only one test case i.e. the encoded message.
The test case consists of one or more words.

The 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

MyNote~ Finished Date:
說明：和上一題類似，只不過是改成輸出左邊第二個鍵的符號即可。
PS. 以上三題，也可以使用整行讀取的方式，處理完之後再整行輸出。

272 - TEX Quotes

TeX is a typesetting language developed by Donald Knuth. It takes source text together with a few typesetting instructions and produces, one hopes, a beautiful document. Beautiful documents use `` and '' to delimit quotations, rather than the mundane " which is what is provided by most keyboards. Keyboards typically do not have an oriented double-quote, but they do have a left-single-quote ` and a right-single-quote '. Check your keyboard now to locate the left-single-quote key ` (sometimes called the ``backquote key'') and the right-single-quote key ' (sometimes called the ``apostrophe'' or just ``quote''). Be careful not to confuse the left-single-quote ` with the ``backslash'' key \. TeX lets the user type two left-single-quotes `` to create a left-double-quote `` and two right-single-quotes '' to create a right-double-quote ''. Most typists, however, are accustomed to delimiting their quotations with the un-oriented double-quote ".

If the source contained

"To be or not to be," quoth the bard, "that is the question."

then the typeset document produced by TeX would not contain the desired form:

``To be or not to be," quoth the bard, ``that is the question."

In order to produce the desired form, the source file must contain the sequence:

``To be or not to be,'' quoth the bard, ``that is the question.''

You are to write a program which converts text containing double-quote (") characters into text that is identical except that double-quotes have been replaced by the two-character sequences required by TeX for delimiting quotations with oriented double-quotes. The double-quote (") characters should be replaced appropriately by either `` if the " opens a quotation and by '' if the " closes a quotation. Notice that the question of nested quotations does not arise: The first " must be replaced by ``, the next by '', the next by ``, the next by '', the next by ``, the next by '', and so on.

Input and Output

Input will consist of several lines of text containing an even number of double-quote (") characters. Input is ended with an end-of-file character. The text must be output exactly as it was input except that:

- the first " in each pair is replaced by two ` characters: `` and
- the second " in each pair is replaced by two ' characters: ''.

Sample Input

```
"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!"
```

Sample Output

```
``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!''
```

MyNote:將第奇數個的 " 改成以 `` 印出來，第偶數個的 " 則改成以 '' 印出來，而其他字元則照原本的印出來。

Finished Date:

494 - Kindergarten Counting Game

Everybody sit down in a circle. Ok. Listen to me carefully.

``Woooooo, you scwewy wabbit!''

Now, could someone tell me how many words I just said?

Input and Output

Input to your program will consist of a series of lines, each line containing multiple words (at least one). A ``word'' is defined as a consecutive sequence of letters (upper and/or lower case).

Your program should output a word count for each line of input. Each word count should be printed on a separate line.

Sample Input

```
Meep Meep!
I tot I taw a putty tat.
I did! I did! I did taw a putty tat.
Shsssssssssh ... I am hunting wabbits. Heh Heh Heh Heh ...
```

Sample Output

```
2
7
10
9
```

MyNote- Finished Date:

說明：計算每一行出現多少個英文字(連續出現的英文字母)。

注意，本題字串長度至少為 101。

10340 - All in All

You have devised a new encryption technique which encodes a message by inserting between its characters randomly generated strings in a clever way. Because of pending patent issues we will not discuss in detail how the strings are generated and inserted into the original message. To validate your method, however, it is necessary to write a program that checks if the message is really encoded in the final string.

Given two strings *s* and *t*, you have to decide whether *s* is a subsequence of *t*, i.e. if you can remove characters from *t* such that the concatenation of the remaining characters is *s*.

Input Specification

The input contains several testcases. Each is specified by two strings *s, t* of alphanumeric ASCII characters separated by whitespace. Input is terminated by EOF.

Output Specification

For each test case output, if *s* is a subsequence of *t*.

Sample Input

sequence subsequence
person compression
VERDI vivaVittorioEmanueleReDiItalia
caseDoesMatter CaseDoesMatter

Sample Output

Yes
No
Yes
No

MyNote~ Finished Date:

說明：每一行有兩個字串 s、t，如果 s 是 t 的子字串，也就是將 t 去掉某些字母之後就會和 s 一樣，則印出 "Yes"，否則印出 "No" 。

注意，本題字串長度至少為 90001。

string 與 stringstream

資料來源：http://www.tcgs.tc.edu.tw/~sagit/cpp/q6.htm

一、string

string 是 C++ 提供的字串型態，和 C 的字串相比，除了有不限長度的優點外，還有其他許多方便的功能。要使用 string， 必須先加入這一行：

#include <string>

接下來要宣告一個字串變數，可以寫成：

string s;

我們也可以在宣告的同時讓它設成某個字串：

string s="TCGS";

而要取得其中某一個字元，和傳統 C 的字串一樣是用 s[i] 的方式取得。比較不一樣的是如果 s 有三個字元，傳統 C 的字串的 s[3] 是 0 字元，但是 C++ 的 string 則是只到 s[2] 這個字元而已。

下面我們把 string 與 字元陣列的語法做一個對照：

操作	string	字元陣列
宣告字串	string s;	char s[100];
取得第 i 個字元	s[i]	s[i]
字串長度	s.length() 或 s.size()	strlen(s)
讀取一行	getline(cin, s);	gets(s);
設成某字串	s="TCGS";	strcpy(s, "TCGS");
字串相加	s=s+"TCGS"; s+="TCGS"	strcat(s, "TCGS");
字串比較	s=="TCGS"	strcmp(s, "TCGS")

從上面的表格，我們可以發現 string 的用法比較直覺，因此如果沒有特別的需要，儘量使用 string 會比較方便。

二、stringstream

stringstream 是 C++ 提供的另一個字串型的串流(stream)物件，和之前學過的 iostream、fstream 有類似的操作方式。要使用 stringstream， 必須先加入這一行：

```
#include <sstream>
```

stringstream 主要是用在將一個字串分割，可以先用 clear( ) 以及 str( ) 將指定字串設定成一開始的內容，再用 >> 把個別的資料輸出，例如：

```
string s;
stringstream ss;
int a, b, c;
getline(cin, s);
ss.clear();
ss.str(s);
ss >> a >> b >> c;
```

下面我們看到一個使用 stringstream 的例子：

題目：輸入的第一行有一個數字 N 代表接下來有 N 行資料，每一行資料裡有不固定個數的整數(最多 20 個，每行最大 200 個字元)，請你寫一個程式將每行的總和印出來。

輸入：

```
3
1 2 3
20 17 23 54 77 60
111 222 333 444 555 666 777 888 999
```

輸出：

```
6
251
4995
```

程式如下：

```
string s;
stringstream ss;
int n, i, sum, a;
cin >> n;
getline(cin, s); // 讀取換行
for (i=0; i<n; i++)
{
    getline(cin, s);
```

```
        ss.clear();
        ss.str(s);
        sum=0;
        while (1)
        {
                ss >> a;
                if ( ss.fail() ) break;
                sum+=a;
        }
        cout << sum << endl;
}
```

三、C 字串分割函數 — strtok( )

strtok( ) 函數是另一個字串分割的選擇，它可以將原本的 C 傳統字串依指定的字元把它們分割成兩個子字串，我們可以用 atoi(s) 、 atol(s) 、 atof(s) 等函數將切出來的字串轉換成整數、長整數、浮點數(double)等 (PS.要 #include <stdlib.h>)。

我們先宣告字串長度：

```
#define LEN 200
```

主程式：

```
int n, i, sum;
char s[LEN], *p;
cin.getline(s, LEN);
n=atoi(s);
for(i=0; i<n; i++) {
   cin.getline(s, LEN);
   for (sum=0, p=strtok(s, " "); p; p=strtok(0, " "))
         sum+=atoi(p);
   cout << sum << endl;

}
```

strtok( ) 函數的第一個參數是要切割的字串，第二個參數則是分割字元。如果成功，則傳回切割出來的第一個字串的位址，如果失敗，則傳回 0。而第二次使用 strtok() 時，則第一個參數要用 NULL (0) ，這樣它就會繼續上一次的字串切割，說明如下：

1.呼叫 strtok() 之前（s="I LOVE U"）：

s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]	s[7]	s[8]
'I'	' '	'L'	'O'	'V'	'E'	' '	'U'	'\0'

2.執行 p=strtok(s, " ")：

s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]	s[7]	s[8]
'I'	'\0'	'L'	'O'	'V'	'E'	' '	'U'	'\0'

p 為 s[0] 的位址，strtok() 記錄 s[2] 的位址

3.再執行 p=strtok(0, " ")：

s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]	s[7]	s[8]
'I'	'\0'	'L'	'O'	'V'	'E'	'\0'	'U'	'\0'

p 為 s[2] 的位址，strtok() 記錄 s[7] 的位址

4.再執行 p=strtok(0, " ")：

s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]	s[7]	s[8]
'I'	'\0'	'L'	'O'	'V'	'E'	'\0'	'U'	'\0'

p 為 s[7] 的位址，strtok() 記錄 s[8] 的位址

從上面的過程中，我們可以知道原本的字串 s 會被切割成數個子字串，而這些子字串中間以 '\0' 分隔開。

四、練習題：Q482 — Permutation Arrays

輸入：

```
3 1 2
32.0 54.7 -2
```

輸出：

```
54.7
-2
32.0
```

說明：輸入兩行文字，第一行有 N 個整數，代表第二行中 N 個浮點數的順序，請把第二行的浮點數依序印出來。

482 - Permutation Arrays

In many computer problems, it is necessary to permute data arrays. That is, the data in an array must be re-arranged in some specified order. One way to permute arbitrary data arrays is to specify the permutations with an index array to point out the position of the elements in the new array. Let x be an array that is to be permuted and let x' be the permuted array. Then, we have the relationship between x and x' that $x'_{p_i} = x_i$.

Input

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

Each input set will contain two lines of numbers. The first line will be an index array p containing the integers 1...n, where n is the number of integers in the list. The numbers in the first line will have been permuted in some fashion. The second line will contain a list numbers in floating point format.

Output

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.

The output for this program will be the list of floating point numbers from the input set, ordered according to the permutation array from the input file. The output numbers must be printed one per line in the same format in which they each appeared in the input file.

Sample Input

```
1

3 1 2
32.0 54.7 -2
```

Sample Output

```
54.7
-2
32.0
```

MyNote~ Finished Date:

說明：輸入兩行文字，第一行有 N 個整數，代表第二行中 N 個浮點數的順序，請把第二行的浮點數依序印出來。