

**The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College,
Springfield, MA
October 25, 2003**

Problem #1: Lingo

In the game of Lingo, there is a word to guess. The player makes a guess and is shown which letters match in the same positions and which appear in the word but in a different position.

Write a program whose input consists of the word to guess and a guess, of the same length. The output is a string representing how the two strings match. There should be an asterisk (*) in each position of the guess that matches the corresponding letter of the word to guess. There should be a pound sign (#) in each position of the guess that matches a letter in the word to guess but not in the right position. There should be a period (.) in all other positions of the guess. For example, given the word to guess HOUSE and the guess SOUTH, the output should be #**.# because O and U match and are in the right positions while S and H match but are out of place. Multiple letters match only once. For example, given the words CARRY and ERROR, the output should be .#*.., since one R matches in the right position and another matches but is out of place. In such a case, it is the leftmost letters that are considered to match. Thus, given CARRY and ERROR, it is the first R in ERROR (and not the third one) that matches the second R in CARRY. Also, matches in the right position take precedence over ones out of place. For example, given the words COLORS and REVERT, the output should be*. since one of the Rs matches in the right position.

Sample input:

HOUSE
SOUTH

Sample output:

#**.#

Sample input:

CARRY
ERROR

Sample output:

.#*..

Sample input:

COLORS
REVERT

Sample output:

....*.

**The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College, Springfield, MA
October 25, 2003**

Problem #2: Find Word

In the *Find Word* puzzle, you are given a square array of letters. You are also given a word that you are supposed to find inside that array of letters, horizontally, vertically, diagonally, in either direction. An example of such an array is given below.

```
W Y O W R E S T F G
T R J E O F Q U L R
M P L E O F E U D Z
A V G K M W P R I Y
P R T H L O G X N T
L H L P S Y H L E B
E R O O D N C P G W
D G W U I E G A P P
C O A D U T R G P O
B H G P M B V E W S
```

You might be asked to find words like DOOR, PAGE, TRIP, ROOM, WEEKLY, HOME, and QUIT in this array.

Write a program to read in an array and a word and find all occurrences of the word in the array. The input is the number of rows in the array (up to 20), followed by the rows, followed by the word to find. If the word is not in the array, the phrase "NOT FOUND" should be displayed. If it is found, the starting position (row and column number, counting from 1) and the direction should be displayed. The direction could be

- "right"
- "left"
- "down"
- "up"
- "diagonally up to the right"
- "diagonally up to the left"
- "diagonally down to the right"
- "diagonally down to the left"

Sample input:

10
WYOWRESTFG
TRJEOFQULR
MPLEOFEUDZ
AVGKMWPRIY
PRTHLOGXNT
LHLPSYHLEB
ERODNCPGW
DGWUIEGAPP
COADUTRGPO
BHGPMBVEWS
PAGE

Sample output:

row 7, column 8, down
row 8, column 9, left
row 10, column 4, diagonally up to the left

**The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College, Springfield, MA
October 25, 2003**

Problem #3: The Basics of Bases

A place value numeration system

- has a specific base
- has place value positions that are powers of the base
- has digits representing the values 0, 1, ..., (base - 1) where uppercase letters are used as needed for digits greater than 9 (e.g., A represents 10, B represents 11, etc.)

For example, in our base 10 system, the place values are powers of 10 and the available digits are 0, 1, 2, ..., 9. Using the place values, the numeral 627 is expanded as

$$627 = 6 \cdot 10^2 + 2 \cdot 10^1 + 7 \cdot 10^0.$$

A numeral in any base can be expanded similarly and converted to its equivalent base 10 value - multiply each digit by the value of its position and then add the products. The following table provides some examples.

base	digits	numeral	converted to base 10
5	0,1,2,3,4	2304	$2 \cdot 5^3 + 3 \cdot 5^2 + 0 \cdot 5^1 + 4 \cdot 5^0 = 329$
2	0,1	110101	$1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 53$
12	0,1,2,...,9,A,B	8A3	$8 \cdot 12^2 + 10 \cdot 12^1 + 3 \cdot 12^0 = 1275$

You will be presented with a sequence of pairs of numerals without knowing their corresponding bases. For a given pair of numerals, say X and Y, you must find the smallest base for X and the smallest base for Y such that the numerals X and Y have the same base-ten value when converted from their respective bases. Notice that since the available digits for any numeral are 0, 1, 2, ..., (base - 1), the base chosen for a numeral must be

larger than each of the digits in the numeral. For example, the numeral 14 cannot be in base 2, 3, or 4.

Suppose we are given the numerals 14 and 21.

14 in base 5 is equal to 9 in base 10

21 in base 4 is equal to 9 in base 10

So 14 (base 5) = 21 (base 4).

The input will consist of a sequence of pairs of numerals, X and Y, one pair per line. The numerals consist of digits 0 - 9 and uppercase letters A - Z (representing digits 10 - 35, respectively). Each numeral contains at most 16 characters. A pair of zeros signals the end of the input.

For each pair of numerals, display the smallest bases for which the pair of numerals has the same base-ten value. The possible bases are 2 - 36. Format each line as shown in the sample output. If the two numerals cannot be equal for any bases, then print a message as shown in the sample output.

Sample Input:

```
14 21
13 D
135 468
100 4
25B 2A6
0 0
```

Sample Output:

```
14 (base 5) = 21 (base 4)
13 (base 10) = D (base 14)
135 is not equal to 468 for any base
100 (base 2) = 4 (base 5)
25B (base 13) = 2A6 (base 12)
```

**The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College, Springfield, MA
October 25, 2003**

Problem #4: Reading, Rating, and 'Rithmetic

The Flesch Readability Index is a simple tool to gauge the readability of a document without linguistic analysis. The index is calculated as follows:

- Count all words in the file. A *word* is any sequence of characters delimited by white space (spaces, tabs, or blank lines), whether or not it is an actual English word.
- Count all syllables in each word. To make this simple, use the following rules (in order of precedence):
 1. Each *group* of adjacent vowels (a, e, i, o, u, y) counts as one syllable. For example, the "ea" in "real" counts as one syllable, but the "e" and "a" in "regal" count as two syllables since they are separated by a consonant.
 2. An "e" at the end of a word does not count as a syllable. For example, the "e" in "cake" does not count as a syllable. However, the "eye" in "buckeye" does count as a syllable according to rule 1.
 3. Each word has at least one syllable, even if the previous rules give a count of 0. For example, the word "the" counts as one syllable, and the word "plbtbrbrtt" also counts as one syllable.
- Count all sentences. A sentence is counted when a word ends with one of the following punctuation marks: period, colon, semicolon, question mark, or exclamation mark.
- The index is computed using the following formula:

$$\begin{aligned} \text{Index} &= 206.835 \\ &\quad - 84.6 * (\text{number of syllables} / \text{number of words}) \\ &\quad - 1.015 * (\text{number of words} / \text{number of sentences}) \end{aligned}$$

rounded to the nearest integer.

The index is an integer, usually between 0 and 100, that indicates how difficult the text is to read. Note that lower numbers indicate a more difficult reading level. The table on the next page translates the index into various educational levels.

Index	Educational Level
91-100	5 th grader
81-90	6 th grader
71-80	7 th grader
66-70	8 th grader
61-66	9 th grader
51-60	High school student
31-50	College student
0-30	College graduate
Less than 0	Law school graduate

Your program should read in a text file and compute the readability index. Each line of the input contains at most 80 characters. A single line containing only the letters ZZZ, beginning at column 1, signals the end of the input. You may assume that the index computed will be at most 100.

Output for the program consists of the number of syllables, number of words, number of sentences, readability index and the corresponding educational level, formatted as indicated in the sample output.

Sample Input:

While much attention has been focused on teacher shortages in classrooms nationwide, the need for principals and school administrators is increasing too. Professors are working to meet the demand.

ZZZ

Sample Output:

51 syllables
29 words

2 sentences
Index = 43
College student

**The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College, Springfield, MA
October 25, 2003**

Problem #5: Parentheses Nightmare

Professor Parenthesis just had a horrible nightmare. All of Professor Parenthesis's nightmares are programmed in Psil, a language which consists entirely of various sorts of parentheses. P.P., as he is known around campus, knows that he has a certain number of each of the following types of parentheses:

left parenthesis	(
right parenthesis)
left brace	{
right brace	}
right superparenthesis]

His nightmare consists of finding each legal Psil expression that these various parentheses can form. Then P.P. wakes up. P.P. wants to know how long his next nightmare will be.

A legal expression in Psil is defined as

1. An empty string OR
2. (P), where P is a legal Psil expression OR
3. {P}, where P is a legal Psil expression OR
4. (...(P], where P is a legal Psil expression such that
 - P does not begin with a (
 - P is immediately preceded by some positive number of left parentheses
5. PQ, where P and Q are legal Psil expressions

For example, ((]) is not a legal Psil expression according to rule 4 because) is not a legal Psil expression. However, (((({})) is legal according to rule 4, where P is the legal Psil expression {}.

Each line of input will consist of 5 integers. The first will be the number of left parentheses, the second the number of right

parentheses, the third the number of left braces, the fourth the number of right braces, and the fifth the number of right superparentheses. You must print out the number of legal Psil expressions that use ALL of the various parentheses. If all five input numbers are 0, the line is not processed and the input is terminated.

Sample Input:

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

Sample Output:

```
3
([ ])
[] ( )
() ( ]
```

```
1
{ }
```

```
0
```

The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College, Springfield, MA
October 25, 2003

Problem #6: Art Appreciation

Trees are powerful data structures in computer programming. There are many forms of trees, such as free trees, rooted trees, ordered trees, binary trees, AVL-trees, and B-trees. In this problem we will consider binary trees. Recall that a **binary tree** is either empty, or it contains a finite set of elements of which one is called the **root** of the tree and the remaining elements are divided into two disjoint subsets, each of which is itself a binary tree. These two subsets are called the **left** and **right subtrees** of the original tree. If A is the root of a binary tree and B is the root of its left or right subtree, then B is called the **left** or **right child** of A. Each element of the tree is called a **node**.

There are many ways to represent binary trees. In languages that support dynamic structures, such as C++ and Java, binary trees can be defined by pointers or objects. A method supported by all programming languages is to use a table to represent a binary tree. Each row of the table contains an element of the tree, and indices of its left and right children. If an element does not have a left or right child, a zero is used. For example,

1	B	0	0
2	D	0	0
3	A	6	2
4	F	5	3
5	R	0	1
6	C	0	0

represents the following tree:

```

      F
      *
*****
*       *
R       A
*       *
***   *****
     * * *

```

B C D

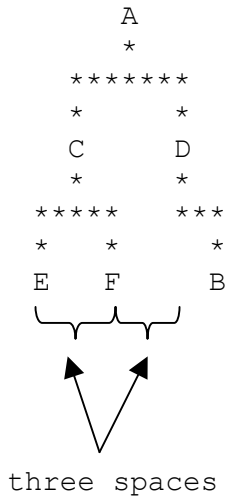
Note the root of the tree may not be the first element of the table.

You are to write a program to display a tree as above, given as input a tree in the table format. The input data begins with a line containing a positive integer N representing the number of elements in the tree. The next N lines give the table representation of the tree. Assume that each tree node is represented by an uppercase letter, and that a tree can contain 1 to 26 nodes. Your program should read the binary tree and draw a graph of the tree. Use an asterisk (*) to represent a vertical twig. If a node, say A, has only one child, say B, it should be represented as follows:



depending on whether B is a left child or a right child.

If a node has two children, then you should use a series of asterisks to draw a horizontal twig so that there are exactly three vertical spaces between the leftmost descendant of the right subtree and the rightmost descendant of the left subtree of that node, such as



Sample Input:

```
8
B 0 0
D 0 0
```

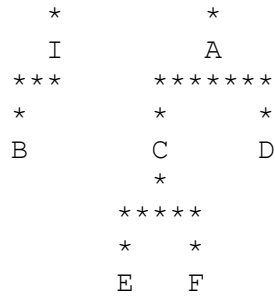
Sample output:

```
      H
      *
*****
```

```

A 7 2
F 0 0
H 6 3
I 1 0
C 8 4
E 8 0

```



Note that the leftmost node of the tree should be on column 1 of your output. Also, if a node has two children, it should be displayed directly above the middle of the horizontal twig to provide a balanced look. More specifically, assuming there are N asterisks in a horizontal twig, if N is odd, then the root is on top of the $(N+1)/2$ -st asterisk (see node A above); if N is even, then the root is on top of the $N/2$ -st asterisk (see node H above).

**The Northeast Regional Competition
of the
2003-2004 ACM International Collegiate Programming Contest
sponsored by IBM
Eastern Preliminary: Western New England College, Springfield, MA
October 25, 2003**

Problem #7: The Price to Write

A text editor has several available operations that can be used to transform a source string into a target string. The operations are described as follows:

- \c copies the next character of the source string to the target string
- \d deletes the next character of the source string
- \ix inserts the character `x` into the target string
- \rx replaces the next character of the source string with the character `x` and copies it to the target string
- \t "twiddles" or interchanges the next two characters of the source string and copies them to the target string
- \k kills the remainder of the source string

As an example, the source string **algorithmic** can be transformed into the target string **glisten** using the following sequence of operations.

<u>operation</u>	<u>source string</u>	<u>target string</u>
\d	lgorithmic	
\t	orithmic	gl
\d	rithmic	gl
\d	ithmic	gl
\c	thmic	gli
\is	thmic	glis
\c	hmic	glist
\re	mic	gliste
\rn	ic	glisten
\k		glisten

Each of the operations will be assigned an associated cost. The cost of a sequence of transformations is the sum of the costs of the individual operations in the sequence. The previous example of converting the source string **algorithmic** to the target string **glisten** would have total cost

$$3 * \text{cost}(\text{delete}) + \text{cost}(\text{twiddle}) + 2 * \text{cost}(\text{copy}) \\ + \text{cost}(\text{insert}) + 2 * \text{cost}(\text{replace}) + \text{cost}(\text{kill})$$

You will be given a source string and a target string and the cost of each operation. You are required to find a least expensive transformation sequence.

The input will consist of 6 integers, one per line, that correspond to the costs of the operations copy, delete, insert, replace, twiddle and kill, respectively. Lines 7 and 8 of the input contain the source string and target string, respectively. Each string has a maximum of 30 characters.

On the first line of the output, display the minimum total cost for a least expensive transformation sequence. The remaining lines of the output must contain a transformation sequence with one operation per line. If there is more than one sequence of minimum cost, any transformation sequence of minimum cost will be judged correct.

Sample Input:

```
1
4
4
3
2
1
contestant
offset
```

Sample Output:

```
15
\d
\c
\rf
```

\rf
\t
\c
\k