235 Typesetting

Proportional fonts are so called because characters require varying amounts of space on the printed line. The size in which text is "set," usually measured in points, also affects the space required for each character. In this problem you are given a number of paragraphs of text to set. Each paragraph may include special "words" to select the font and point size.

Input

The input starts with the font width table. These data give the widths of 10- point characters in six different fonts. The first line contains the number of characters in the table, N ($0 \leq N \leq 100$). Each of the next N lines contain a character in column 1 and then 6 integers representing the width of that character in each of the 6 different fonts. Widths are given in an arbitrary measurement called "units." The width of each 10-point character will be greater than zero units, and less than 256 units. Character widths scale linearly with point size. Thus if a 10-point "A" is 12 units wide, a 20-point "A" is 24 units wide.

The remainder of the input consists of paragraphs to be typeset. Each paragraph begins with a line containing two integers, L and W. L is the number of input lines of text for the paragraph (these immediately follow the first line), and W is the width allowed for each typeset line, in units. The initial font at the beginning of each paragraph is always font 1, and the initial point size in which characters are to be set is 10. Fonts are numbered 1 through 6, corresponding to columns 1 through 6 in the font width table. An empty paragraph (one for which L is 0) will mark the end of the input data. No output is to be produced for this empty paragraph.

The words in each paragraph are sequences of no more than 8 non-blank characters separated by spaces (that is, blanks — no tab characters will appear in the input). Spaces at the ends of input lines are irrelevant, and spaces between words are significant only to the extent that they separate words. Each character in each word will appear in the width table. Case is significant for all characters in the input data.

The special tokens '`*f1`', '`*f2`', '`*f3`', '`*f4`', '`*f5`', and '`*f6`' are used to select a particular font to be used in setting the text that follows it. The token '`*s`N', where N is an integer in the range 1 to 99 indicates that N point characters are to be used in setting the following text. These tokens will always be separated from words and other tokens by at least one blank. Note that style and size changes made in one paragraph do not carry over to the next paragraph, and that many such changes may appear in a single paragraph.

For each paragraph, try to set as many words per line as possible, ensuring that each word is followed by at least the width of a blank (which will always appear in the font width table) with the same point size and style as the characters in the preceding word, except for the last word on the line. The last word in a typeset line must not have any following space.

When scaling fonts, round the scaled character widths to the nearest integer, rounding upward in cases where the rounded value is half way between two consecutive integers. Thus, if a particular 10 point character occupies 9 units of space, a 15 point character would occupy 14 units of space, as would a 16 point character. A 14 point character, however, would occupy only 13 units of space.

Output

For each paragraph, first display the paragraph number (1, 2, ...). Then, for each typeset line in the paragraph, display the line number, the first and last words on that line, and the total number of units

of white space that follow the last character printed on the line. (This is just the number of units of space available on the line not occupied by characters or spaces between characters.)

If a single word exceeds the width of a line, set it on a line by itself. In the output for that line, show only that single word, and a negative amount of white space equal to the excess width of the word.

Sample Input

```
4
A 10 20 30 12 22 32
B 1 2 3 4 5 6
C 9 10 8 3 5 2
  2 4 6 3 5 7
2 80
*f2  AAA BBB CCC
  ABC *s15 CBA AABC CACA
3 100
AAA
AAA BBB CCC
ABC CBA AABC CACA
0 0
```

Sample Output

```
Paragraph 1
Line 1: AAA ... BBB (10 whitespace)
Line 2: CCC ... ABC (14 whitespace)
Line 3: CBA ... CBA (32 whitespace)
Line 4: AABC ... AABC (2 whitespace)
Line 5: CACA (-10 whitespace)
Paragraph 2
Line 1: AAA ... CCC (4 whitespace)
Line 2: ABC ... AABC (26 whitespace)
Line 3: CACA ... CACA (62 whitespace)
```