Problem E. TextColumns

Problem Statement

An OCR program (optical character recognition) needs to perform other things than just recognizing letters. One of these things is mapping out the page layout. In this problem, you should solve the similar, but easier task, of concatenating text spread across several text columns.

The input text will be given as a String[] and split up into several text columns. Consider for instance the text below: (quotes are for clarity only)

```
{"  This is       put  it       that       ",
 " a text          all        is           ",
 "in  several    together       nicely     ",
 "columns.       into  a       formatted.  ",
 "               single                    ",
 " Your  job        text                   ",
 "is     to                    Good luck!  "}
```

The text above obviously has three text columns. Formally, text columns are always separated where there are three or more consecutive empty character columns (i.e. the only character in those columns are space). Furthermore, text columns can't be empty (see example 3).

You should also find all paragraphs. The paragraph splits occur on blank lines within a text column. Only blank lines between the first and last row used in a text column should be considered. In the text above, the last line in column two is empty, but this is not a paragraph split because there is no text below this line in the same text column.

The return value should contain the same text, in a String[], but where each element corresponds to one paragraph. Empty paragraphs should be ignored (that is, there should be no empty strings). Leading and trailing spaces should be removed in each paragraph, and consecutive spaces should be replaced by a single space. Line breaks within a paragraph should also be replaced by a single space so the last word on a line is not concatenated with the first word on the next line.

Create a class TextColumns containing the method formatText which takes a String[] **text** containing the text and which returns a String[], containing the formatted text in the format described above. See the examples for further explanations.

Definition

Class: TextColumns

Method: formatText

Parameters: String[]

Returns: String[]

Method signature: String[] formatText(String[] text)

(be sure your method is public)

Constraints

- **text** will contain between 1 and 50 elements inclusive.
- Each element in **text** will contain between 1 and 50 characters, inclusive.
- All elements in **text** will contain the same number of characters.
- **text** will only contain the characters 'A'-'Z', 'a'-'z', '0'-'9', '.', ',', '!', '?' and space.
- At least one character in **text** will not be a space.

Examples

0)

```
{"  This is       put  it      that       ",
 " a text          all       is           ",
 "in  several    together      nicely     ",
 "columns.       into  a      formatted.  ",
 "               single                   ",
 " Your  job        text                  ",
 "is     to                   Good luck!  "}
```

```
Returns:
{ "This is a text in several columns.",
 "Your job is to put it all together into a single text that is nicely formatted.",
 "Good luck!" }
```

This is the example in the problem statement.

1)

```
{"This is                                    ",
 "just one                                   ",
 "column                                     ",
 "                                           ",
 "a                 l                        ",
 "   b                 m                     ",
 "      c                 n                  ",
 "         d                 o               ",
 "            e                 p            ",
 "               f                 q         ",
 "            g                 r            ",
 "         h                 s               ",
 "      i                 t                  ",
 "   j                 u                     ",
 "k                 v                        "}

Returns: { "This is just one column",  "a l b m c n d o e p f q g r h s i t j u k v" }
```

2)

```
 {"                                    ",
  "                                    ",
  " Some text                          ",
  " here and                           ",
  "  some                              ",
  "                                    ",
  "                            new     ",
  "                                over",
  "                             here   ",
  "                text                ",
  "                  there             ",
  "                                    ",
  "                                    ",
  "                   And              ",
  "             something              ",
  "                                    ",
  "                                    ",
  "                             Finito ",
  "                                    "}

Returns:
{ "Some text here and some text there",
 "And something new over here",
 "Finito" }
```

3)

```
 {"    Column 1                                      ",
  "                   Column 2                       ",
  "                                   Column 3       "}

Returns: { "Column 1 Column 2 Column 3" }
```