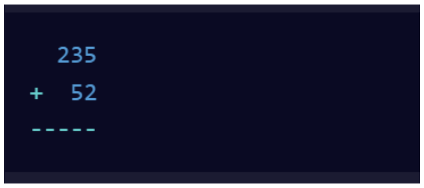
Build an Arithmetic Formatter Project

Students in primary school often arrange arithmetic problems vertically to make them easier to solve. For example, "235 + 52" becomes:

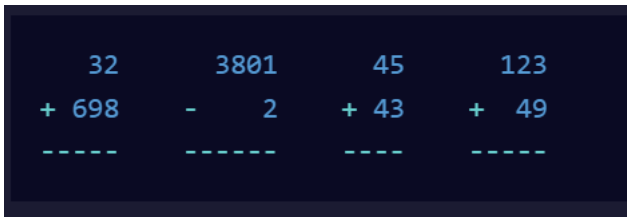


```
  235
+  52
-----
```

Finish the arithmetic_arranger function that receives a list of strings which are arithmetic problems, and returns the problems arranged vertically and side-by-side. The function should optionally take a second argument. When the second argument is set to True, the answers should be displayed.

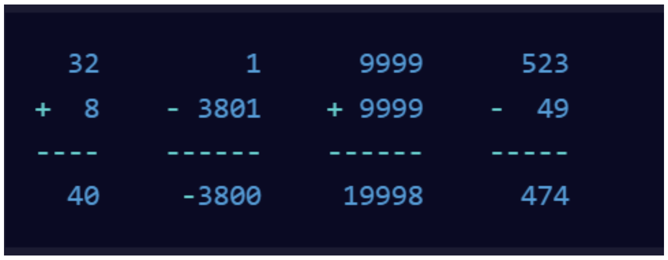
Example

Function Call:
arithmetic_arranger(["32 + 698", "3801 - 2", "45 + 43", "123 + 49"])
Output:



```
   32      3801      45      123
+ 698    -    2    + 43    +  49
-----    ------    ----    -----
```

Function Call:
arithmetic_arranger(["32 + 8", "1 - 3801", "9999 + 9999", "523 - 49"], True)
Output:



```
  32         1      9999      523
+  8    - 3801    + 9999    -  49
----    ------    ------    -----
  40     -3800     19998      474
```

Rules

The function will return the correct conversion if the supplied problems are properly formatted, otherwise, it will return a string that describes an error that is meaningful to the user.

Situations that will return an error:

- If there are too many problems supplied to the function. The limit is five, anything more will return: 'Error: Too many problems.'
- The appropriate operators the function will accept are addition and subtraction. Multiplication and division will return an error. Other operators not mentioned in this bullet point will not need to be tested. The error returned will be: "Error: Operator must be '+' or '-'."
- Each number (operand) should only contain digits. Otherwise, the function will return: 'Error: Numbers must only contain digits.'
- Each operand (aka number on each side of the operator) has a max of four digits in width. Otherwise, the error string returned will be: 'Error: Numbers cannot be more than four digits.'

If the user supplied the correct format of problems, the conversion you return will follow these rules:

- There should be a single space between the operator and the longest of the two operands, the operator will be on the same line as the second operand, both operands will be in the same order as provided (the first will be the top one and the second will be the bottom).
- Numbers should be right-aligned.
- There should be four spaces between each problem.
- There should be dashes at the bottom of each problem. The dashes should run along the entire length of each problem individually. (The example above shows what this should look like.)

Tests

1. arithmetic_arranger(["3801 - 2", "123 + 49"]) should return   3801   123\n-   2   + 49\n------   -----.
2. arithmetic_arranger(["1 + 2", "1 - 9380"]) should return   1   1\n+ 2   - 9380\n---   ------.
3. arithmetic_arranger(["3 + 855", "3801 - 2", "45 + 43", "123 + 49"]) should return   3   3801   45   123\n+ 855   -   2   + 43   + 49\n-----   ------   ----   -----.
4. arithmetic_arranger(["11 + 4", "3801 - 2999", "1 + 2", "123 + 49", "1 - 9380"]) should return   11   3801   1   123   1\n+ 4   - 2999   + 2   + 49   - 9380\n----   ------   ---   -----   ------.
5. arithmetic_arranger(["44 + 815", "909 - 2", "45 + 43", "123 + 49", "888 + 40", "653 + 87"]) should return 'Error: Too many problems.'.
6. arithmetic_arranger(["3 / 855", "3801 - 2", "45 + 43", "123 + 49"]) should return "Error: Operator must be '+' or '-'.".
7. arithmetic_arranger(["24 + 85215", "3801 - 2", "45 + 43", "123 + 49"]) should return 'Error: Numbers cannot be more than four digits.'.
8. arithmetic_arranger(["98 + 3g5", "3801 - 2", "45 + 43", "123 + 49"]) should return 'Error: Numbers must only contain digits.'.
9. arithmetic_arranger(["3 + 855", "988 + 40"], True) should return   3   988\n+ 855   + 40\n-----   -----\n 858   1028.

10. arithmetic_arranger(["32 - 698", "1 - 3801", "45 + 43", "123 + 49", "988 + 40"], True) should return   32   1   45   123   988\n- 698   - 3801   + 43   + 49   + 40\n-----   ------   ----   -----   -----\n -666   -3800   88   172   1028