Modeling Challenge III: Calibron 12

Eckel, TJHSST AI1, Fall 2021

Background & Explanation

To finish off Unit 2, you will be solving a famous problem called the Calibron 12. More specifically, you'll be solving a variety of Calibron-12-like puzzles; the Calibron 12 itself is *quite* challenging! There is a link on the website to a site where you can see this famous puzzle; read up before continuing. As with Rush Hour, you can declare success or failure for BLUE credit. Unlike Rush Hour, RED credit is available for a good optimization (see Unit 2 Extension Options).

Required Task

You will be given the size of a large containing rectangle and then the dimensions of a number of smaller rectangles. The goal is to fit all the little rectangles into the big one. The following input string specifies the Calibron 12 puzzle itself:

```
"56 56 28x14 32x11 32x10 21x18 21x18 21x14 21x14 17x14 28x7 28x6 10x7 14x4"
```

The first number is the height of the puzzle, the second the width. After that are the dimensions of each rectangle. Note that you may get two (or more!) different rectangles with the same dimensions, and note also rectangles can be rotated (so `21x18` might be `18x21` in the solution.)

On the website, there's a tiny amount of shell code for you to download (really just code to read in and unpack the puzzle statement) that also contains specific instructions about input and output formatting. Be sure you read them carefully. ***Note, please, that your code will receive one puzzle directly on the command line (a single long string specified on in quotes on the command line so it arrives as one argument with spaces, not many separate values), not a filename as usual.***

You can solve this puzzle any way you want. There are no rules about data structure or method (except for the usual stipulations that you can't google a solution or import any non-standard libraries). It just has to work.

Test Cases

This test case should output "Containing rectangle incorrectly sized."

```
"10 13 3x6 2x5 4x10 7x9 1x1"
```

This test case should output "No solution." The rectangles are the right total area, but no good arrangement is possible.

```
"8 5 4x2 4x2 4x2 4x2 2x4"
```

Here are some standard test cases. To get full credit on this assignment, you should be able to solve each of the following test cases in no more than 15 seconds per test case:

```
"4 7 7x4"
"2 3 1x2 2x2"
"18 9 3x11 5x7 4x8 6x10 1x2"
"4 8 4x1 1x6 1x3 3x1 1x3 1x3 6x1 1x4"
"11 12 3x6 2x5 4x10 7x9 1x1"
"9 18 3x8 5x10 4x11 6x7 1x2"
"13 14 4x5 3x8 6x11 7x10 2x1"
"19 19 1x19 1x12 6x9 9x15 15x3 10x6 3x12"
```

A later RED assignment in this unit allows you to choose between several options; one of those options will be to optimize this assignment. If you declare SUCCESS and then want to dive deeper, you can get RED credit for that assignment by solving each of the following test cases in no more than 1 minute per test case (in particular, the final test case is the actual Calibron 12):

```
"24 24 4x7 7x4 7x5 11x4 9x3 2x8 22x3 8x4 14x11 8x10 2x12 5x3 9x3"
"56 56 28x14 32x11 32x10 21x18 21x18 21x14 21x14 17x14 28x7 28x6 10x7 14x4"
```

Specification if you Declare Success

If you believe you have succeeded in the task, submit a single Python script to the link on the course website marked "success".

This assignment is **complete** if:

- The "Name" field on the Dropbox submission form contains your **class period**, then your **last name**, then your **first name**, in that order.
- Your code does all of the following:
    - Accept a single command line argument specifying one puzzle (see shell code).
    - Output a solution to the puzzle according to the specification in the shell code. (Please follow it **exactly**. Any extraneous print statements, including any empty lines, will cause my grader to fail.)
- Total runtime is less than 2 minutes. See timing hints on the front to check yourself.

Specification if you Declare Failure

If you decide that your code can't be completed, submit a single document file (doc, docx, or pdf preferred) to the link on the course website marked "failure".

The assignment is **complete** if:

- The "Name" field on the Dropbox submission form contains your **class period**, then your **last name**, then your **first name**, in that order.
- You write a 500 – 1000 word analysis of your code including what you tried to do, what the code actually does, and why it isn't working.

Extension: Calibron-12 Optimization

As mentioned above, this unit features a choice of several options for a RED credit, and a second choice for BLACK credit. One option is optimizing the Calibron-12. Look at the "Unit 2 Extensions" document for more details.