BYU IDAHO CSE 110 | Programming Building Blocks

08 Teach: Team Activity

Loops of Loops

Instructions

Face-to-face students will complete this activity in class.

Online students should arrange for a one hour synchronous meeting to work through the activity together.

Overview

When learning multiplication in school, students often write out multiplication tables where numbers are written along the top of each column and on the left side of each row. Then, the student fills in each spot in the table as the product of the number at top of the column and the left of the row. This might produce something like the following:

```
  1   2   3   4   5   6   7   8   9  10  11  12
  2   4   6   8  10  12  14  16  18  20  22  24
  3   6   9  12  15  18  21  24  27  30  33  36
  4   8  12  16  20  24  28  32  36  40  44  48
  5  10  15  20  25  30  35  40  45  50  55  60
  6  12  18  24  30  36  42  48  54  60  66  72
  7  14  21  28  35  42  49  56  63  70  77  84
  8  16  24  32  40  48  56  64  72  80  88  96
  9  18  27  36  45  54  63  72  81  90  99 108
 10  20  30  40  50  60  70  80  90 100 110 120
 11  22  33  44  55  66  77  88  99 110 121 132
 12  24  36  48  60  72  84  96 108 120 132 144
```

In this activity you'll write a program that uses loops within loops to generate multiplication tables.

Hint from Instructor:

This can be a challenging assignment. Please be aware that in addition to the sample solution code, there is also a video walk-through posted at the bottom of this page that steps through the creation of the sample solution line by line, which may be helpful if you feel nervous about this topic or activity.

Assignment

Write a program that asks the user for the number of rows and columns in the table and then generates the associated multiplication table. The following are examples of the finished assignment:

```
How many columns and rows do you want in your multiplication table? 5
  1   2   3   4   5
  2   4   6   8  10
  3   6   9  12  15
  4   8  12  16  20
  5  10  15  20  25
```

```
How many columns and rows do you want in your multiplication table? 12
  1   2   3   4   5   6   7   8   9  10  11  12
  2   4   6   8  10  12  14  16  18  20  22  24
  3   6   9  12  15  18  21  24  27  30  33  36
  4   8  12  16  20  24  28  32  36  40  44  48
  5  10  15  20  25  30  35  40  45  50  55  60
  6  12  18  24  30  36  42  48  54  60  66  72
  7  14  21  28  35  42  49  56  63  70  77  84
  8  16  24  32  40  48  56  64  72  80  88  96
  9  18  27  36  45  54  63  72  81  90  99 108
 10  20  30  40  50  60  70  80  90 100 110 120
 11  22  33  44  55  66  77  88  99 110 121 132
 12  24  36  48  60  72  84  96 108 120 132 144
```

When complete, this program doesn't require too many lines of code, but they can be conceptually difficult to combine together when you are first learning to work with loops, so please work through the steps of this assignment, rather than writing the whole program at once.

To complete the assignment, you need a few other tools.

I. PRINTING NUMBERS THAT TAKE UP A CERTAIN AMOUNT OF SPACE

In previous lessons you've used formatting strings to display numbers with a certain number of decimal places "**{the_number:.2f}**". In a similar way, you can specify the number of spaces to take up by including that number after the colon as follows:

```
the_number = 128
print(f"{the_number}")
print(f"{the_number:1}")
print(f"{the_number:3}")
print(f"{the_number:5}")
print(f"{the_number:10}")
```

This outputs the following:

```
128
128
128
  128
      128
```

Notice that if the number you specify is less than or equal to the number of digits in the number, it will simply display the number normally.

II. PRINTING WITHOUT NEW LINES

To this point, whenever we have used a print statement, it has always been on it's own line, so that the next line starts on a new line. If you *do not* want the print statement to end with a new line, you can specify the **end** as follows:

```
print("This is line one.", end="yyy")
print("This is line two.")
```

This outputs the following:

```
This is line one.yyyThis is line two.
```

Along the same lines, if you do not want anything at the end, you can specify **`end=""`** as follows:

```
print("This is line one.", end="")
print("This is line two.")
```

This outputs the following:

```
This is line one.This is line two.
```

Notice, that because you told the first print statement to end with nothing (by using `""`), it does not end with a newline and the next line prints directly following it.

III. CORE REQUIREMENTS

01. Ask the user for the number of rows and columns (note: you are only asking for a single number that will be used for both the rows and the columns).

    Display the numbers 1 up to *and including* that number, each on their own line at this point.

    The output should look as follows:

```
How many columns and rows do you want in your multiplication table? 5
1
2
3
4
5
```

02. Change the program so that the numbers are printed on the same line with a space between them.

The output should look as follows:

```
How many columns and rows do you want in your multiplication table? 5
1 2 3 4 5
```

03. You now have code that can print out each column you'll need on a line. Now, you need to repeat it for a second line, and a third line, and so forth up until the number of rows that you need.

    To accomplish this, add another loop for each row that has your previous code as its body. Start by simply having it duplicate the same row over and over again. Keep in mind that you'll need to display a newline after the numbers for each row are finished.

    Then, once this is working change it to display the product of the row and column.

The output at the end of the core requirements should look as follows:

```
How many columns and rows do you want in your multiplication table? 5
1 2 3 4 5
2 4 6 8 10
3 6 9 12 15
4 8 12 16 20
5 10 15 20 25
```

IV. STRETCH CHALLENGE

01. Change your program so that each number that is displayed takes up 3 spaces. This should look as follows:

```
How many columns and rows do you want in your multiplication table? 5
  1   2   3   4   5
  2   4   6   8  10
  3   6   9  12  15
  4   8  12  16  20
  5  10  15  20  25
```

```
How many columns and rows do you want in your multiplication table? 12
  1   2   3   4   5   6   7   8   9  10  11  12
  2   4   6   8  10  12  14  16  18  20  22  24
  3   6   9  12  15  18  21  24  27  30  33  36
  4   8  12  16  20  24  28  32  36  40  44  48
  5  10  15  20  25  30  35  40  45  50  55  60
  6  12  18  24  30  36  42  48  54  60  66  72
  7  14  21  28  35  42  49  56  63  70  77  84
  8  16  24  32  40  48  56  64  72  80  88  96
  9  18  27  36  45  54  63  72  81  90  99 108
 10  20  30  40  50  60  70  80  90 100 110 120
 11  22  33  44  55  66  77  88  99 110 121 132
 12  24  36  48  60  72  84  96 108 120 132 144
```

02. Notice that in the example of 5 rows and columns, you don't actually need three spaces per number, but instead two will do. Change your code so that it determines the number of spaces to use automatically, based on the largest number that will be produced.

    For a first approach to this, consider using an if statement to see if the largest number would be greater than or equal to 100.

03. For a second approach to the problem of determining the number of spaces per number, consider mathematically determining the number of digits so that your table could work for any number of digits. (You can now do multiplication tables that go into the thousands and beyond!)

After the final stretch challenge you program can produce the following:

```
How many columns and rows do you want in your multiplication table? 5
 1  2  3  4  5
 2  4  6  8 10
 3  6  9 12 15
 4  8 12 16 20
 5 10 15 20 25
```

```
How many columns and rows do you want in your multiplication table? 12
  1   2   3   4   5   6   7   8   9  10  11  12
  2   4   6   8  10  12  14  16  18  20  22  24
  3   6   9  12  15  18  21  24  27  30  33  36
```

```
  4   8  12  16  20  24  28  32  36  40  44  48
  5  10  15  20  25  30  35  40  45  50  55  60
  6  12  18  24  30  36  42  48  54  60  66  72
  7  14  21  28  35  42  49  56  63  70  77  84
  8  16  24  32  40  48  56  64  72  80  88  96
  9  18  27  36  45  54  63  72  81  90  99 108
 10  20  30  40  50  60  70  80  90 100 110 120
 11  22  33  44  55  66  77  88  99 110 121 132
 12  24  36  48  60  72  84  96 108 120 132 144
```

```
How many columns and rows do you want in your multiplication table? 32
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17
   2    4    6    8   10   12   14   16   18   20   22   24   26   28   30   32   34
   3    6    9   12   15   18   21   24   27   30   33   36   39   42   45   48   51
   4    8   12   16   20   24   28   32   36   40   44   48   52   56   60   64   68
   5   10   15   20   25   30   35   40   45   50   55   60   65   70   75   80   85
   6   12   18   24   30   36   42   48   54   60   66   72   78   84   90   96  102  1
   7   14   21   28   35   42   49   56   63   70   77   84   91   98  105  112  119  1
   8   16   24   32   40   48   56   64   72   80   88   96  104  112  120  128  136  1
   9   18   27   36   45   54   63   72   81   90   99  108  117  126  135  144  153  1
  10   20   30   40   50   60   70   80   90  100  110  120  130  140  150  160  170  1
  11   22   33   44   55   66   77   88   99  110  121  132  143  154  165  176  187  1
  12   24   36   48   60   72   84   96  108  120  132  144  156  168  180  192  204  2
  13   26   39   52   65   78   91  104  117  130  143  156  169  182  195  208  221  2
  14   28   42   56   70   84   98  112  126  140  154  168  182  196  210  224  238  2
  15   30   45   60   75   90  105  120  135  150  165  180  195  210  225  240  255  2
  16   32   48   64   80   96  112  128  144  160  176  192  208  224  240  256  272  2
  17   34   51   68   85  102  119  136  153  170  187  204  221  238  255  272  289  3
  18   36   54   72   90  108  126  144  162  180  198  216  234  252  270  288  306  3
  19   38   57   76   95  114  133  152  171  190  209  228  247  266  285  304  323  3
  20   40   60   80  100  120  140  160  180  200  220  240  260  280  300  320  340  3
  21   42   63   84  105  126  147  168  189  210  231  252  273  294  315  336  357  3
  22   44   66   88  110  132  154  176  198  220  242  264  286  308  330  352  374  3
  23   46   69   92  115  138  161  184  207  230  253  276  299  322  345  368  391  4
  24   48   72   96  120  144  168  192  216  240  264  288  312  336  360  384  408  4
  25   50   75  100  125  150  175  200  225  250  275  300  325  350  375  400  425  4
  26   52   78  104  130  156  182  208  234  260  286  312  338  364  390  416  442  4
  27   54   81  108  135  162  189  216  243  270  297  324  351  378  405  432  459  4
  28   56   84  112  140  168  196  224  252  280  308  336  364  392  420  448  476  5
  29   58   87  116  145  174  203  232  261  290  319  348  377  406  435  464  493  5
  30   60   90  120  150  180  210  240  270  300  330  360  390  420  450  480  510  5
  31   62   93  124  155  186  217  248  279  310  341  372  403  434  465  496  527  5
  32   64   96  128  160  192  224  256  288  320  352  384  416  448  480  512  544  5
```

Sample Solution

When your program is finished, please view the sample solution for this program to compare it to your approach.

You should work to complete this team activity for the one hour period first, without looking at the sample solution. However, if you have worked on it for at least an hour and are still having problems, you may feel free to use the sample solution to help you finish your program.

- Sample solution (Core requirements): teach08_sample.py
- Sample solution (Stretch challenges): teach08_stretch_sample.py

Also, there is a video walk-through of this sample solution:

- Sample Solution Part 1: The first for loop (6 mins)
- Sample Solution Part 2: Loops of loops (7 mins)

Submission

When complete, please report your progress in the associated I-Learn quiz.

If you decided to do additional work on the program after your team activity, either by yourself or with others, feel free to include that additional work when you report on your progress in I-Learn.