

Homework 2 - Multiplication Tables

Due: Monday, March 7th @ 11:59pm

Your sister is such a pain. She is just learning basic math but can't quite grasp the idea of multiplication. Every few minutes she will come up to you and ask you what the product of two numbers is. Finally you decide to give her ALL the numbers.

Using your amazing Visual Basic skills, you will make a program that displays the multiplication table for a range of numbers. To do this you will need to use at least one loop.

For example if the user chooses 2 as the lower bound and 5 as the upper bound the table should look like:

4	6	8	10
6	9	12	15
8	12	16	20
10	15	20	25

Recreate the GUI below.

How To Do Math

Lower Bound: Upper Bound: Calculate

Don't display when sums to:

Label1

Which when you are done will create something like this:

The screenshot shows a Windows application window titled "How To Do Math". It features three input fields at the top: "Lower Bound:" with the value "0", "Upper Bound:" with the value "15", and "Don't display when sums to:" with the value "7". A "Calculate" button is located to the right of these fields. Below the input fields is a 16x16 multiplication table. The table is displayed in a monospaced font, with the first column and first row containing values from 0 to 15. The values in the table are calculated as the product of the row and column indices, with the exception of the diagonal elements which are zero.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
0	3	6	9	12	15	18	21	24	27	30	33	36	39	42	45
0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75
0	6	12	18	24	30	36	42	48	54	60	66	72	78	84	90
0	7	14	21	28	35	42	49	56	63	70	77	84	91	98	105
0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120
0	9	18	27	36	45	54	63	72	81	90	99	108	117	126	135
0	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150
0	11	22	33	44	55	66	77	88	99	110	121	132	143	154	165
0	12	24	36	48	60	72	84	96	108	120	132	144	156	168	180
0	13	26	39	52	65	78	91	104	117	130	143	156	169	182	195
0	14	28	42	56	70	84	98	112	126	140	154	168	182	196	210
0	15	30	45	60	75	90	105	120	135	150	165	180	195	210	225

Things of note:

- Make sure Option Strict is On and Option Explicit is On.
- There are only 4 labels total. Three as labels for the text boxes and one to display the multiplication table.
- In order to use only one label like this you will need to "concatenate" strings together. This is like adding but for strings. $1 + 2 = 3$ but $"1" \& "2" = "12"$
- At the end of each row concatenate on a `vbNewLine` to start the next row. Try this command: `Label1.Text = "First Line" & vbNewLine & "Second Line"`
- You **must** set the font on the multiplication table label to a monospaced font such as `Courier New`. A monospaced font is one where every character is the same width so that they line up in columns nicely. Monospaced fonts are ubiquitous in programming, the Visual Studio code editor uses a monospace font.

- You should try to get the columns to line up nicely. The best way to do this is to use padding. For example, the string "Q" is exactly one character long. `myQString.PadRight(4)` is at MINIMUM four characters long. If spaces were pound signs, this function would return "Q####" making the total length 4. I would suggest you use `PadRight(4)` so that you have 3 spots for digits and one extra space as the actual space between the columns.
 - For extra credit, pad the correct amount based on the length of maximum product. For example, for $15 \times 15 = 225$ so the correct padding is 4 (3 digits + 1 for spacing). For $100 \times 100 = 10,000$ so the correct padding is 6 (5 digits + 1 for spacing).

Why are there blank spots?

Because your sister is picky (or something, just do it).

You will **NOT** print numbers whose components sum to some other number you input. For example if your "don't print when sums to this" number was **9** then you wouldn't print numbers such as 9, 90, 18, 81, 225, 180, etc.

Things to explore to figure this out:

Modulo operator. This allows you to find the remainder. (VB uses `mod`)

Ex. $25 \bmod 10 = 5$

Integer division. This allows you to ignore decimals. (VB uses `\`)

Ex. $25 \backslash 10 = 2$

You only have to handle "not printing" up to three digit numbers. For extra credit try to handle numbers of arbitrary length. If you DO decide to do the extra credit, please comment your code explaining how you did it.