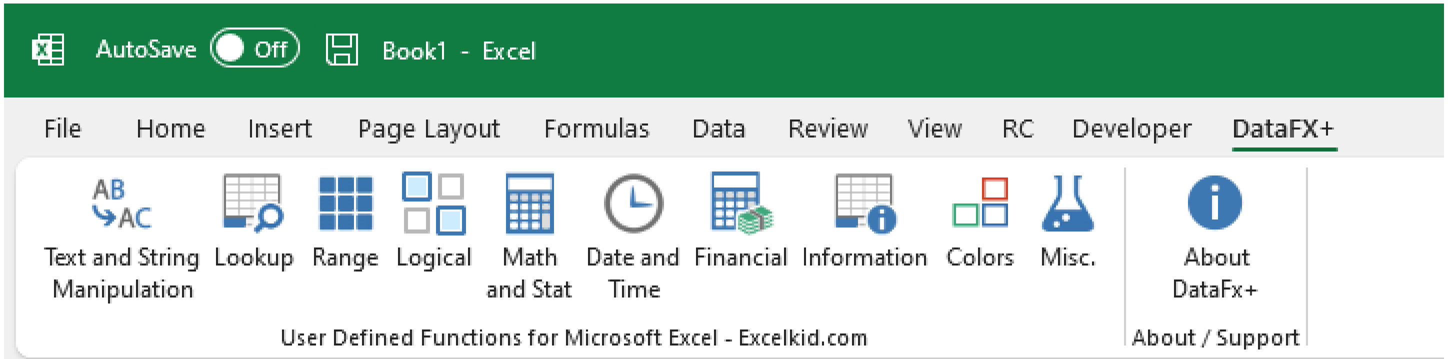
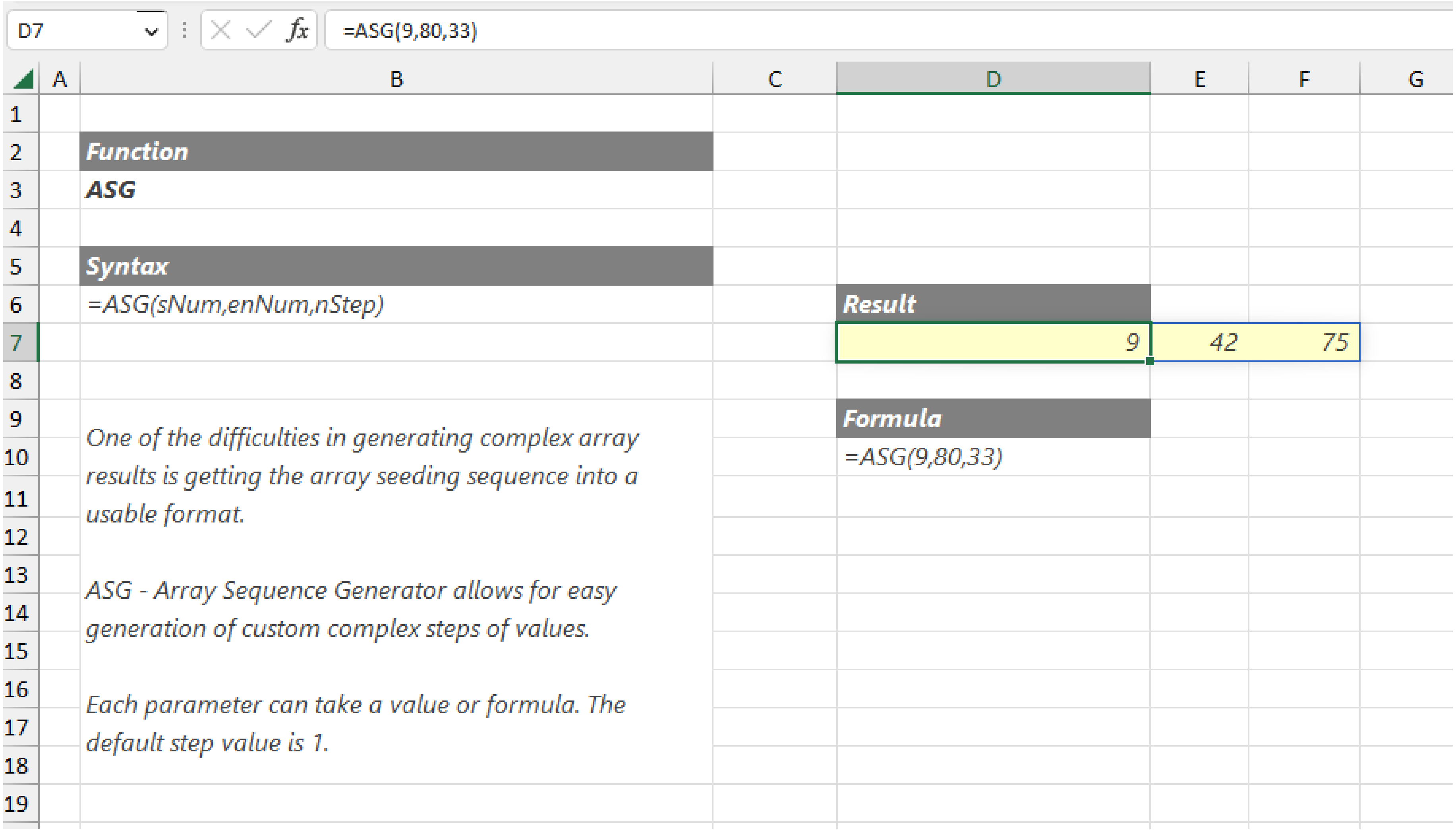
DataFX for Excel

Range functions



ASG



=ASG(9,80,33)

Function

ASG

Syntax

=ASG(sNum,enNum,nStep)

Result

9	*42*	*75*

Formula

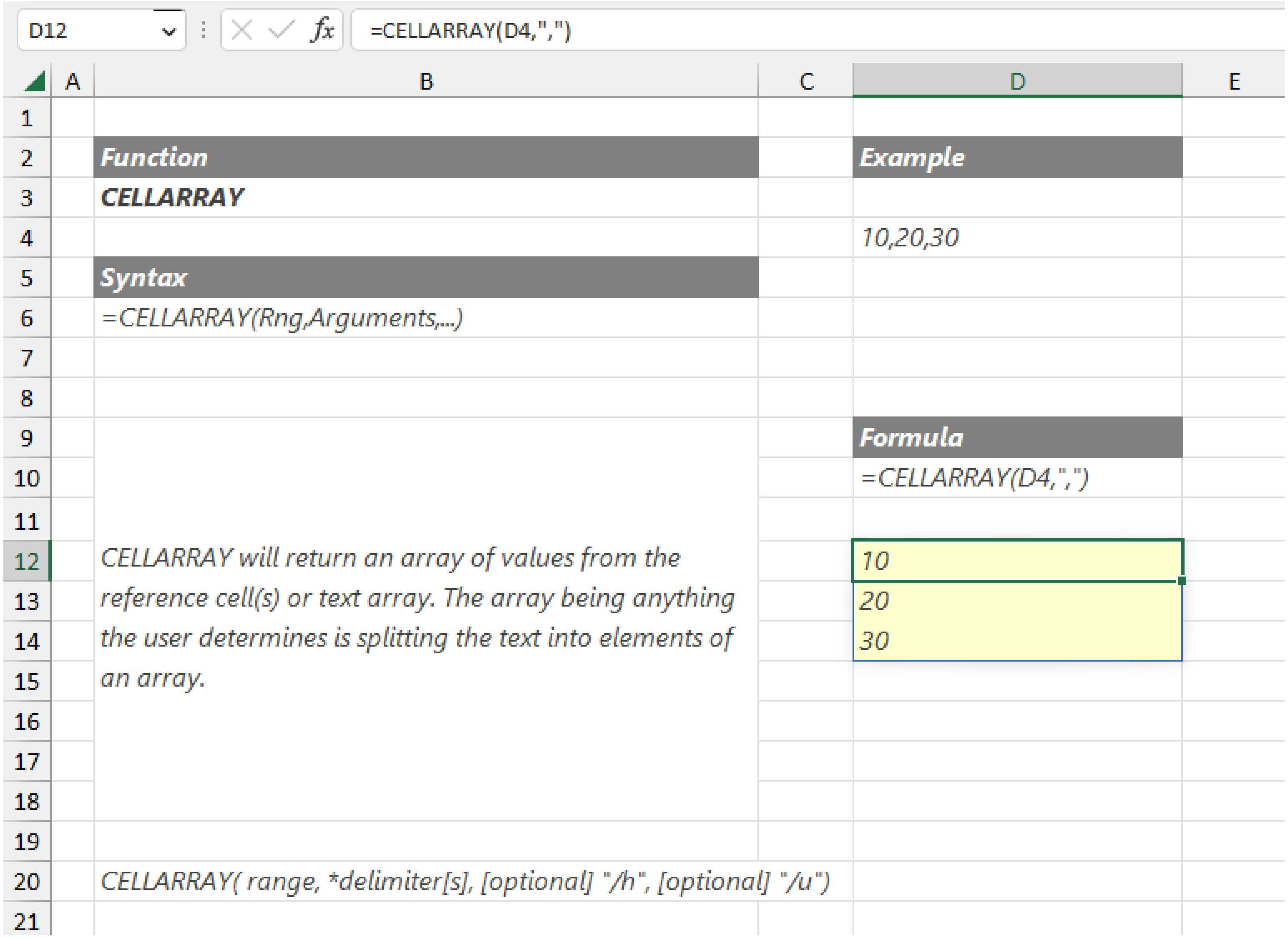
=ASG(9,80,33)

One of the difficulties in generating complex array results is getting the array seeding sequence into a usable format.

ASG - Array Sequence Generator allows for easy generation of custom complex steps of values.

Each parameter can take a value or formula. The default step value is 1.

CELLARRAY



D12

=CELLARRAY(D4,",")

Function

CELLARRAY

Syntax

=CELLARRAY(Rng,Arguments,...)

CELLARRAY will return an array of values from the reference cell(s) or text array. The array being anything the user determines is splitting the text into elements of an array.

CELLARRAY(range, *delimiter[s], [optional] "/h", [optional] "/u")

Example

10,20,30

Formula

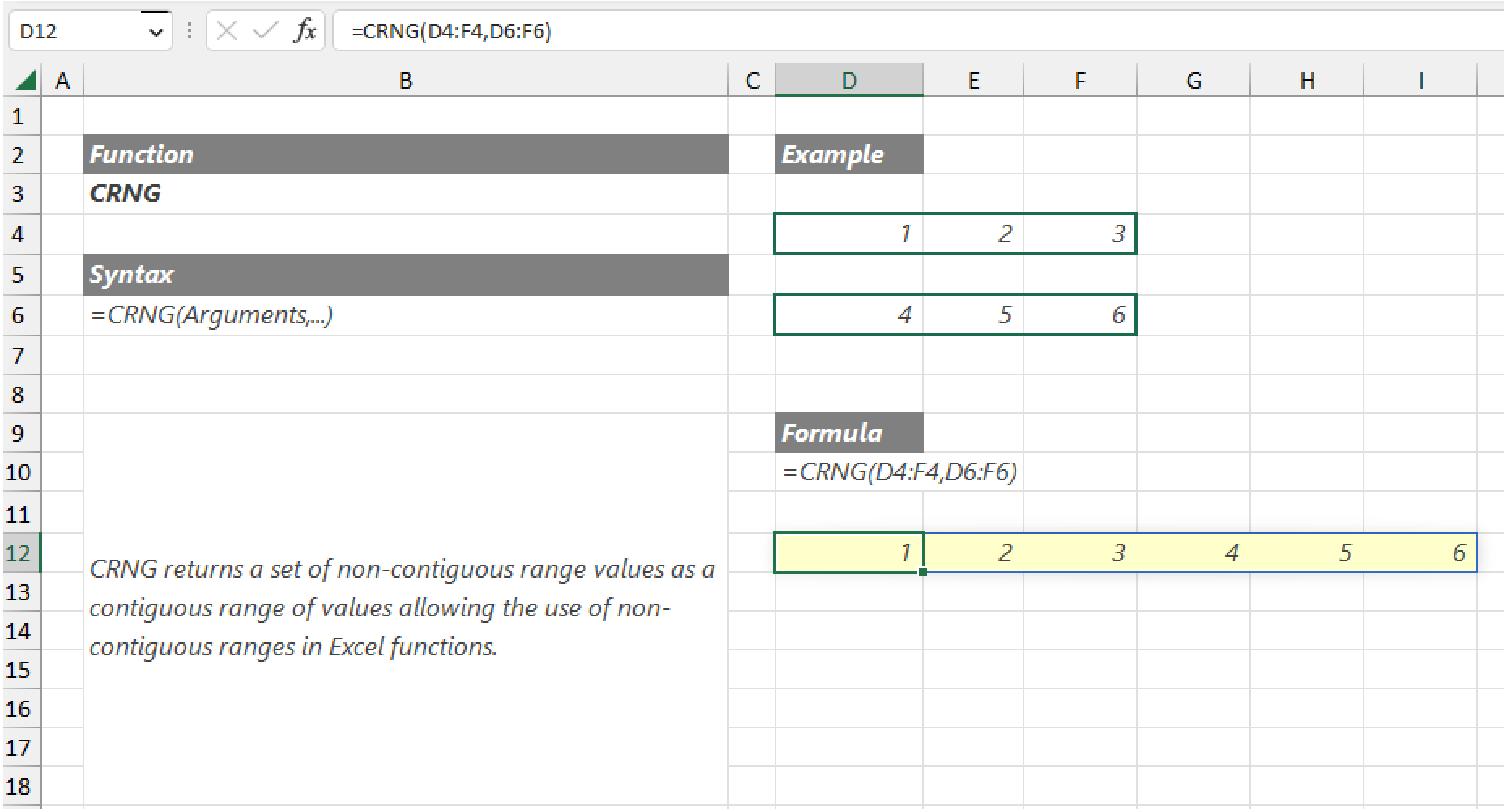
=CELLARRAY(D4,",")

10

20

30

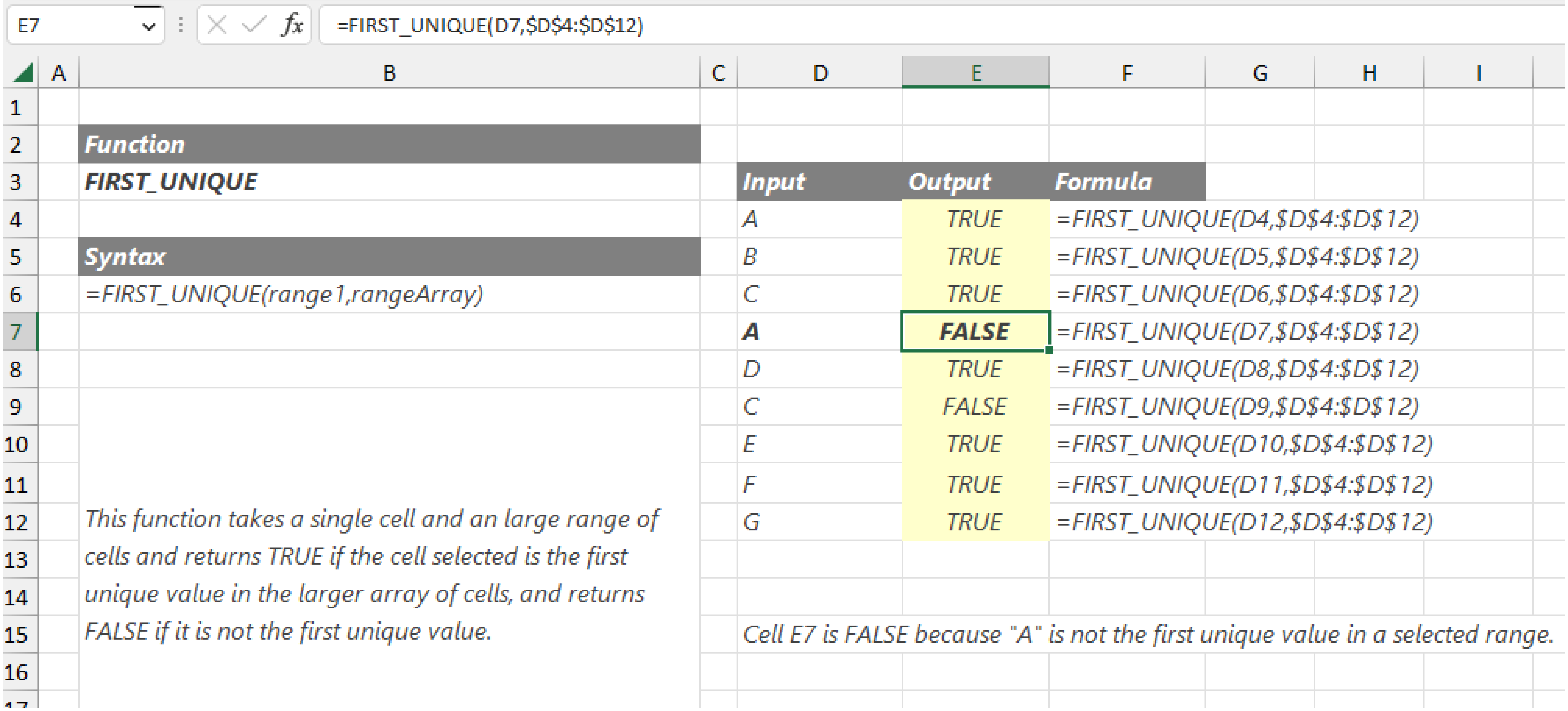
CRNG



D12 | =CRNG(D4:F4,D6:F6)

	A	B	C	D	E	F	G	H	I
1									
2		***Function***		***Example***					
3		***CRNG***							
4				*1*	*2*	*3*			
5		***Syntax***							
6		*=CRNG(Arguments,...)*		*4*	*5*	*6*			
7									
8									
9				***Formula***					
10				*=CRNG(D4:F4,D6:F6)*					
11									
12		*CRNG returns a set of non-contiguous range values as a contiguous range of values allowing the use of non-contiguous ranges in Excel functions.*		*1*	*2*	*3*	*4*	*5*	*6*
13									
14									
15									
16									
17									
18									

FIRST_UNIQUE



E7 =FIRST_UNIQUE(D7,$D$4:$D$12)

Function

FIRST_UNIQUE

Syntax

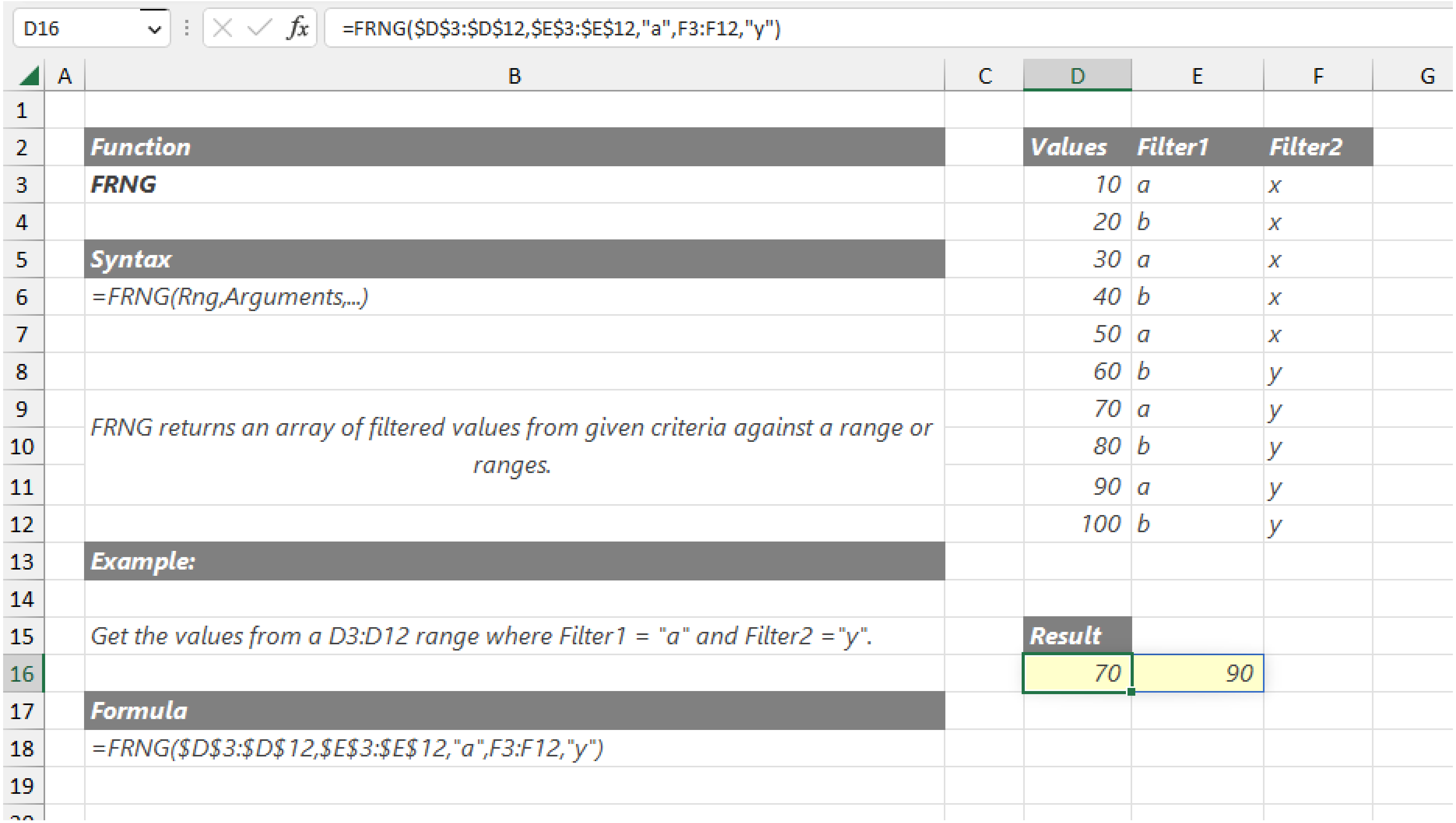
=FIRST_UNIQUE(range1,rangeArray)

This function takes a single cell and an large range of cells and returns TRUE if the cell selected is the first unique value in the larger array of cells, and returns FALSE if it is not the first unique value.

Input	Output	Formula
A	TRUE	=FIRST_UNIQUE(D4,$D$4:$D$12)
B	TRUE	=FIRST_UNIQUE(D5,$D$4:$D$12)
C	TRUE	=FIRST_UNIQUE(D6,$D$4:$D$12)
A	**FALSE**	=FIRST_UNIQUE(D7,$D$4:$D$12)
D	TRUE	=FIRST_UNIQUE(D8,$D$4:$D$12)
C	FALSE	=FIRST_UNIQUE(D9,$D$4:$D$12)
E	TRUE	=FIRST_UNIQUE(D10,$D$4:$D$12)
F	TRUE	=FIRST_UNIQUE(D11,$D$4:$D$12)
G	TRUE	=FIRST_UNIQUE(D12,$D$4:$D$12)

Cell E7 is FALSE because "A" is not the first unique value in a selected range.

FRNG

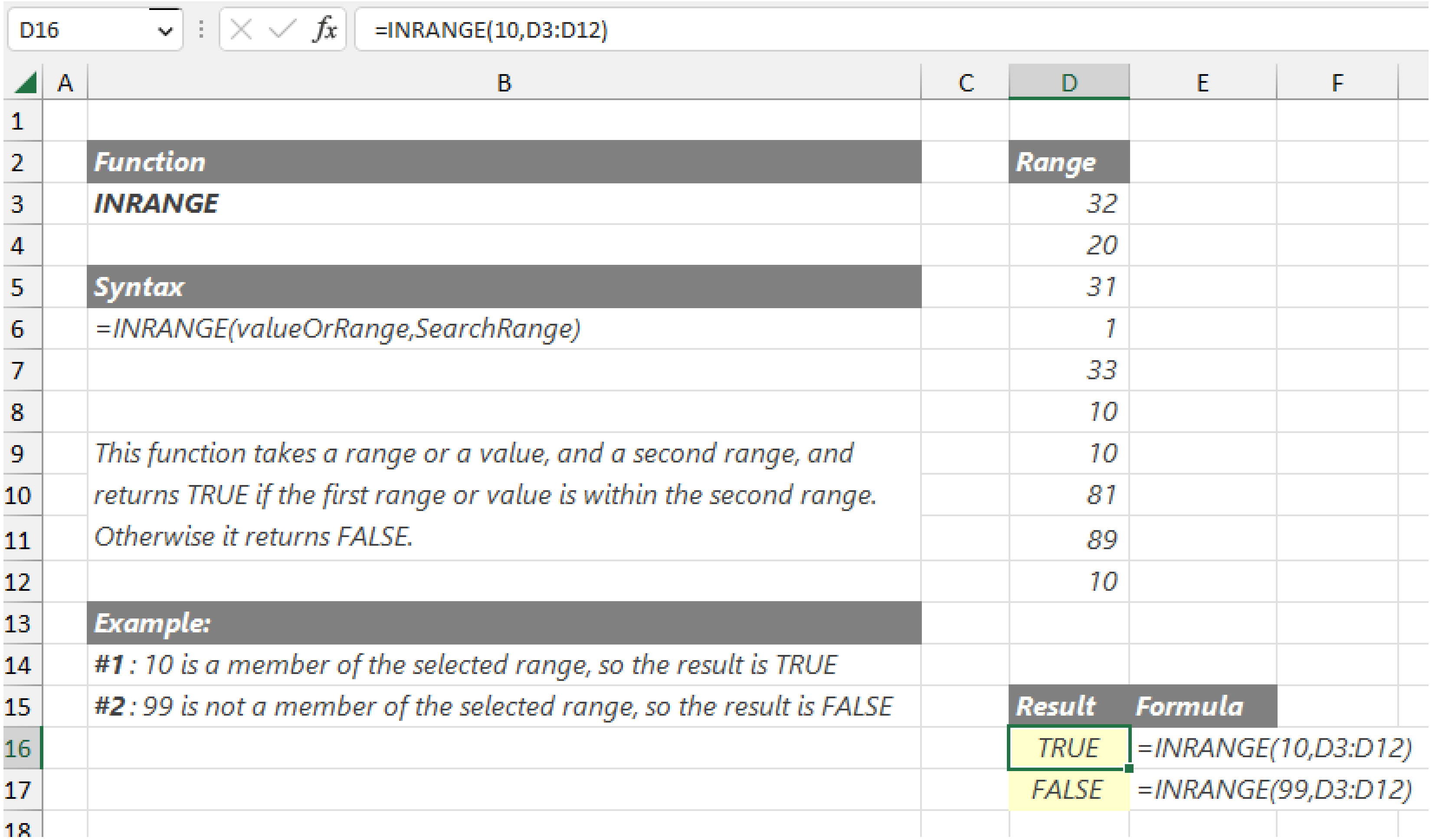


D16

=FRNG($D$3:$D$12,$E$3:$E$12,"a",F3:F12,"y")

	A	B	C	D	E	F	G
1							
2		***Function***		***Values***	***Filter1***	***Filter2***	
3		***FRNG***		*10*	*a*	*x*	
4				*20*	*b*	*x*	
5		***Syntax***		*30*	*a*	*x*	
6		*=FRNG(Rng,Arguments,...)*		*40*	*b*	*x*	
7				*50*	*a*	*x*	
8				*60*	*b*	*y*	
9		*FRNG returns an array of filtered values from given criteria against a range or ranges.*		*70*	*a*	*y*	
10				*80*	*b*	*y*	
11				*90*	*a*	*y*	
12				*100*	*b*	*y*	
13		***Example:***					
14							
15		*Get the values from a D3:D12 range where Filter1 = "a" and Filter2 ="y".*		***Result***			
16				*70*	*90*		
17		***Formula***					
18		*=FRNG($D$3:$D$12,$E$3:$E$12,"a",F3:F12,"y")*					
19							

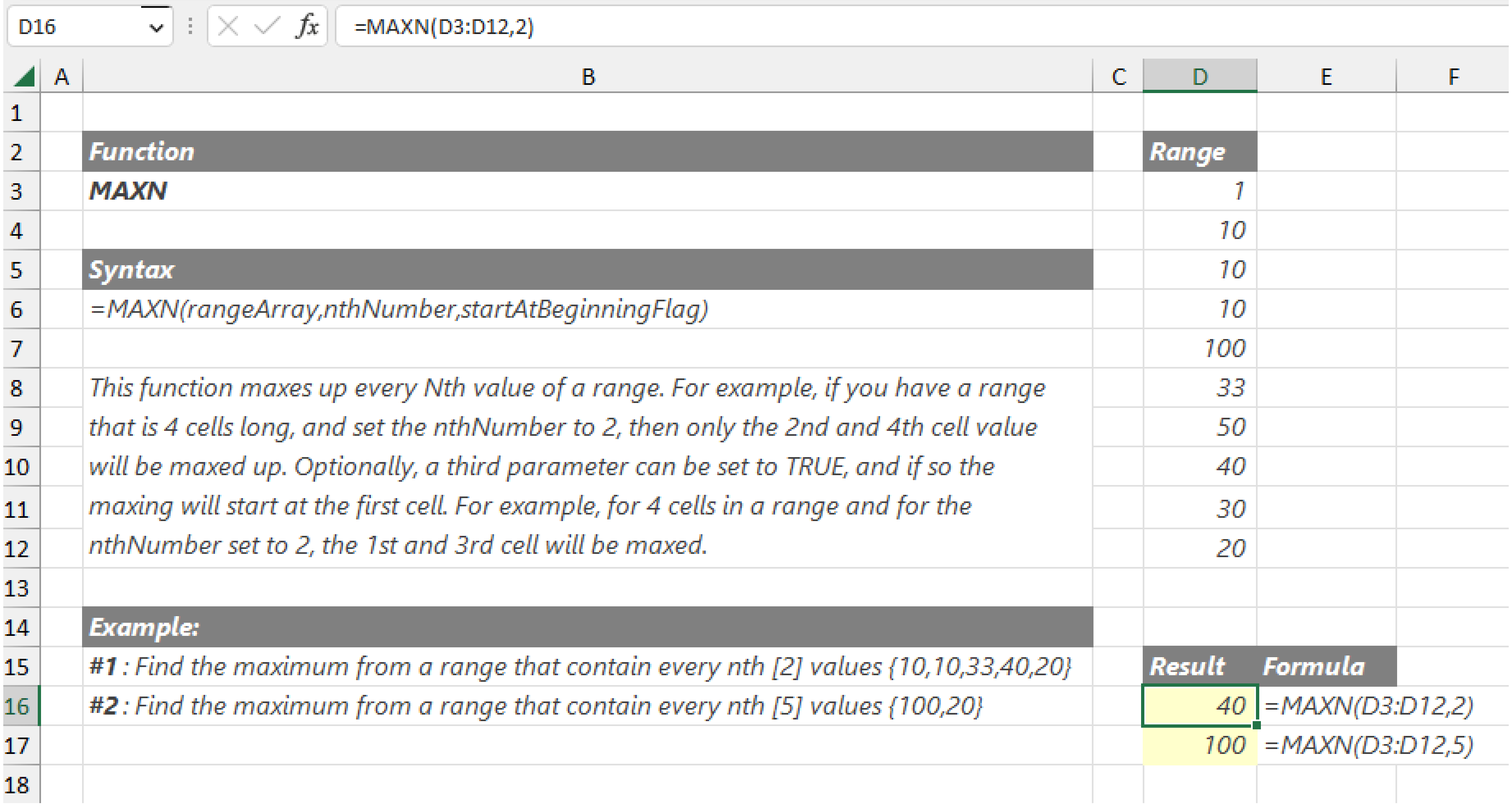
INRANGE



D16 =INRANGE(10,D3:D12)

	A	B	C	D	E	F
1						
2		***Function***		***Range***		
3		***INRANGE***		*32*		
4				*20*		
5		***Syntax***		*31*		
6		*=INRANGE(valueOrRange,SearchRange)*		*1*		
7				*33*		
8				*10*		
9		*This function takes a range or a value, and a second range, and*		*10*		
10		*returns TRUE if the first range or value is within the second range.*		*81*		
11		*Otherwise it returns FALSE.*		*89*		
12				*10*		
13		***Example:***				
14		***#1** : 10 is a member of the selected range, so the result is TRUE*				
15		***#2** : 99 is not a member of the selected range, so the result is FALSE*		***Result***	***Formula***	
16				*TRUE*	*=INRANGE(10,D3:D12)*	
17				*FALSE*	*=INRANGE(99,D3:D12)*	
18						

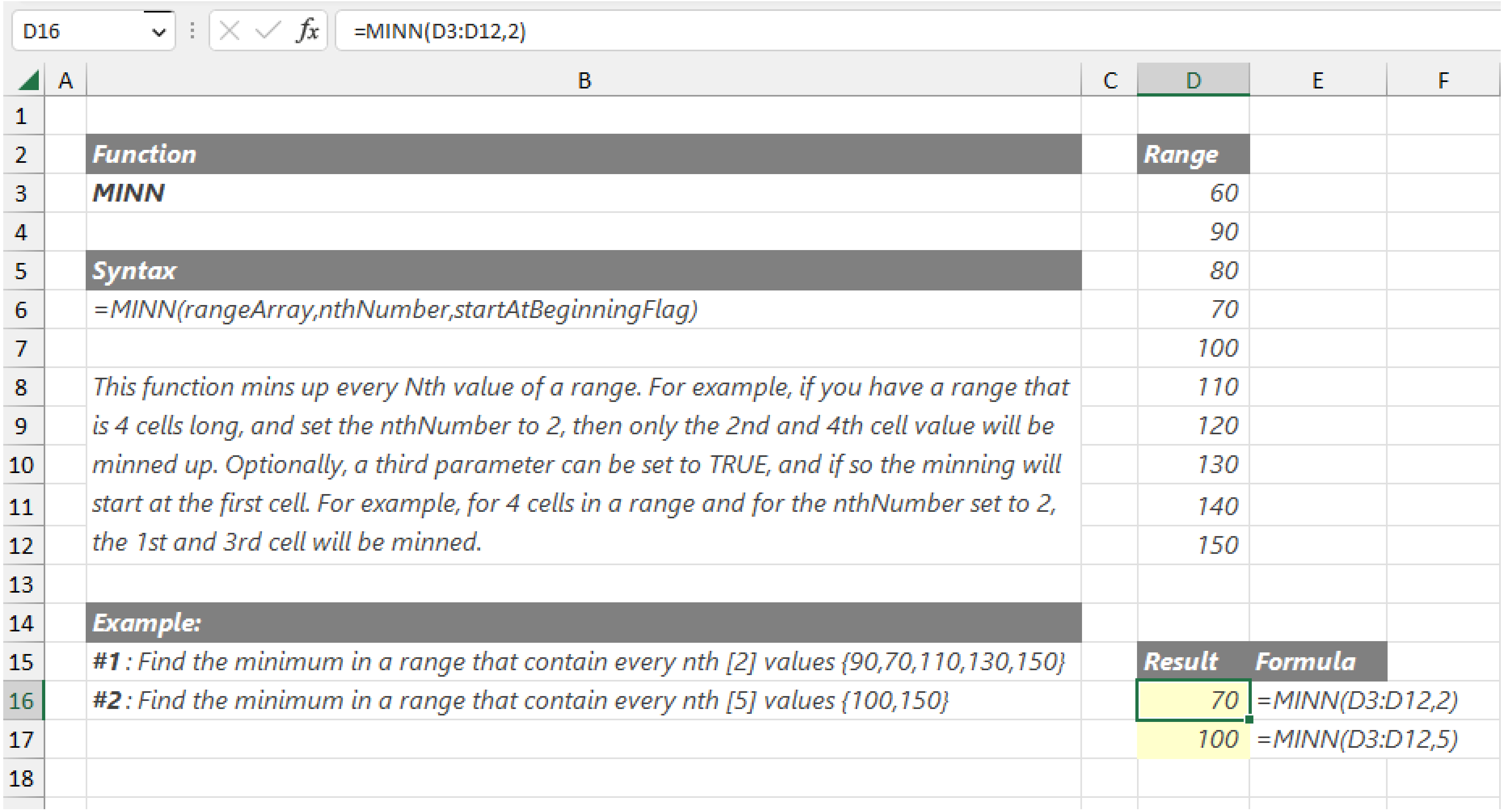
MAXN



D16 =MAXN(D3:D12,2)

	A	B	C	D	E	F
1						
2		***Function***		***Range***		
3		***MAXN***		*1*		
4				*10*		
5		***Syntax***		*10*		
6		*=MAXN(rangeArray,nthNumber,startAtBeginningFlag)*		*10*		
7				*100*		
8		*This function maxes up every Nth value of a range. For example, if you have a range*		*33*		
9		*that is 4 cells long, and set the nthNumber to 2, then only the 2nd and 4th cell value*		*50*		
10		*will be maxed up. Optionally, a third parameter can be set to TRUE, and if so the*		*40*		
11		*maxing will start at the first cell. For example, for 4 cells in a range and for the*		*30*		
12		*nthNumber set to 2, the 1st and 3rd cell will be maxed.*		*20*		
13						
14		***Example:***				
15		***#1** : Find the maximum from a range that contain every nth [2] values {10,10,33,40,20}*		***Result***	***Formula***	
16		***#2** : Find the maximum from a range that contain every nth [5] values {100,20}*		*40*	*=MAXN(D3:D12,2)*	
17				*100*	*=MAXN(D3:D12,5)*	
18						

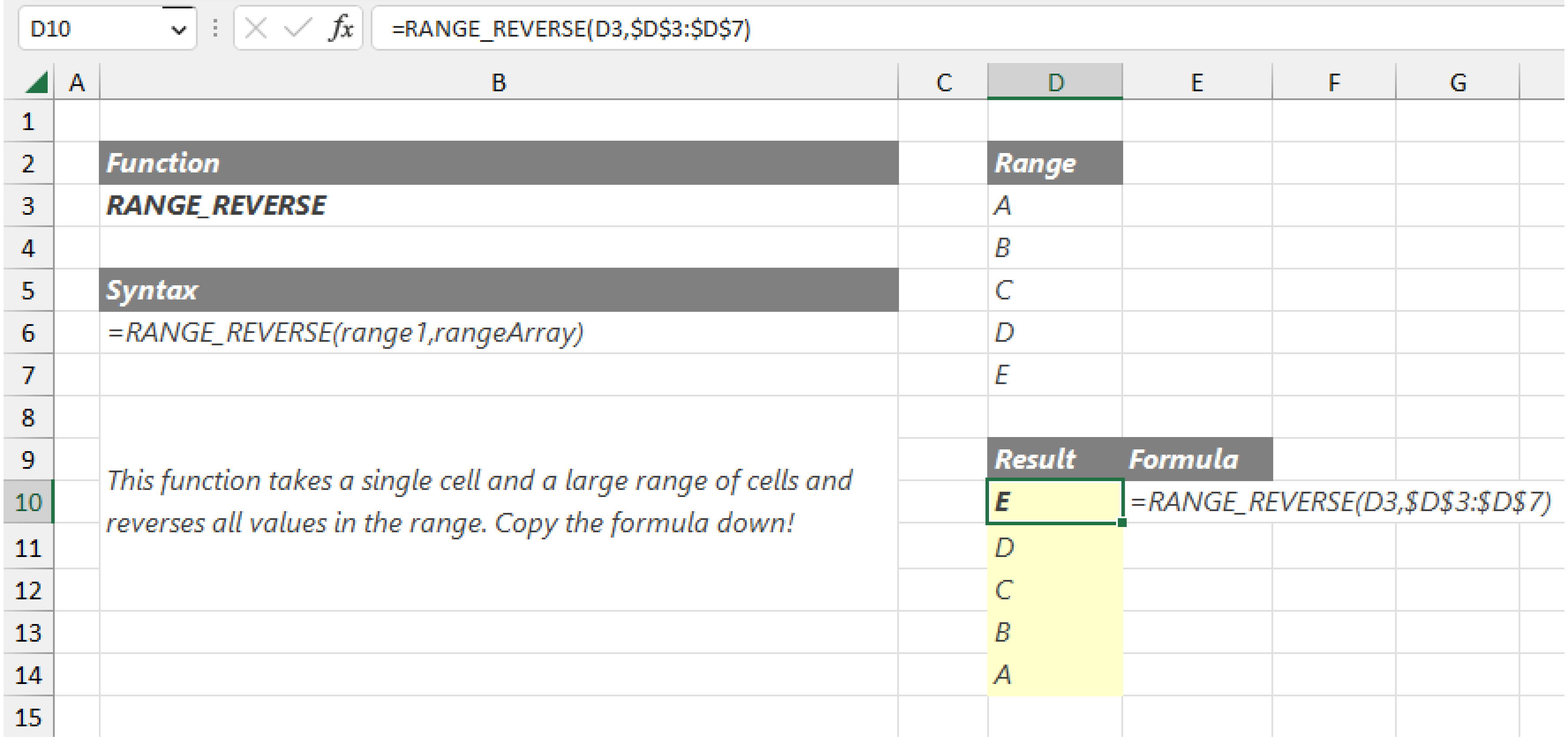
MINN



D16 =MINN(D3:D12,2)

	A	B	C	D	E	F
1						
2		***Function***		***Range***		
3		***MINN***		*60*		
4				*90*		
5		***Syntax***		*80*		
6		*=MINN(rangeArray,nthNumber,startAtBeginningFlag)*		*70*		
7				*100*		
8		*This function mins up every Nth value of a range. For example, if you have a range that is 4 cells long, and set the nthNumber to 2, then only the 2nd and 4th cell value will be minned up. Optionally, a third parameter can be set to TRUE, and if so the minning will start at the first cell. For example, for 4 cells in a range and for the nthNumber set to 2, the 1st and 3rd cell will be minned.*		*110*		
9				*120*		
10				*130*		
11				*140*		
12				*150*		
13						
14		***Example:***				
15		***#1** : Find the minimum in a range that contain every nth [2] values {90,70,110,130,150}*		***Result***	***Formula***	
16		***#2** : Find the minimum in a range that contain every nth [5] values {100,150}*		*70*	*=MINN(D3:D12,2)*	
17				*100*	*=MINN(D3:D12,5)*	
18						

RANGE_REVERSE



=RANGE_REVERSE(D3,$D$3:$D$7)

Function

RANGE_REVERSE

Syntax

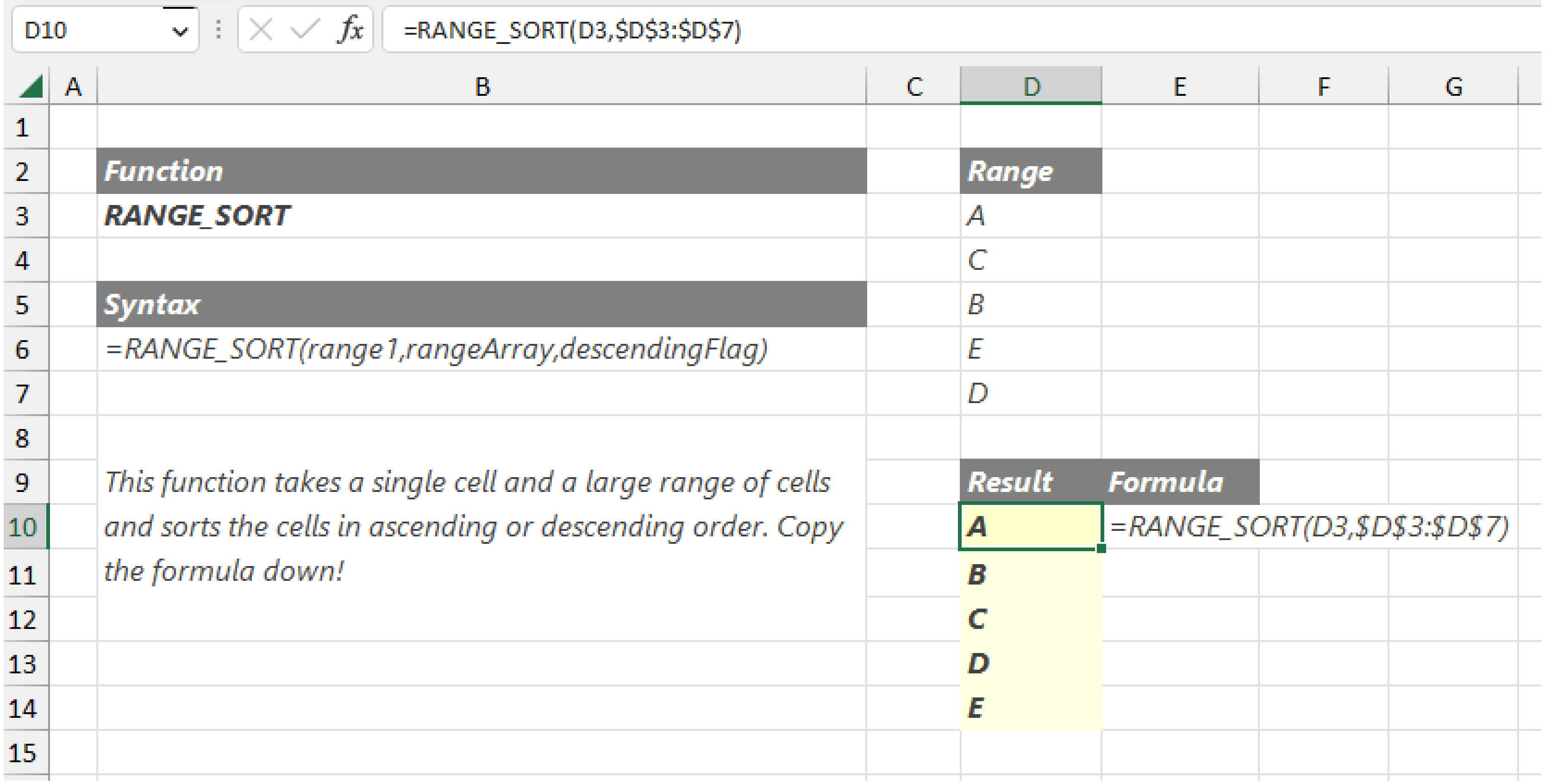
=RANGE_REVERSE(range1,rangeArray)

This function takes a single cell and a large range of cells and reverses all values in the range. Copy the formula down!

Range
A
B
C
D
E

Result	Formula
E	=RANGE_REVERSE(D3,$D$3:$D$7)
D	
C	
B	
A	

RANGE_SORT

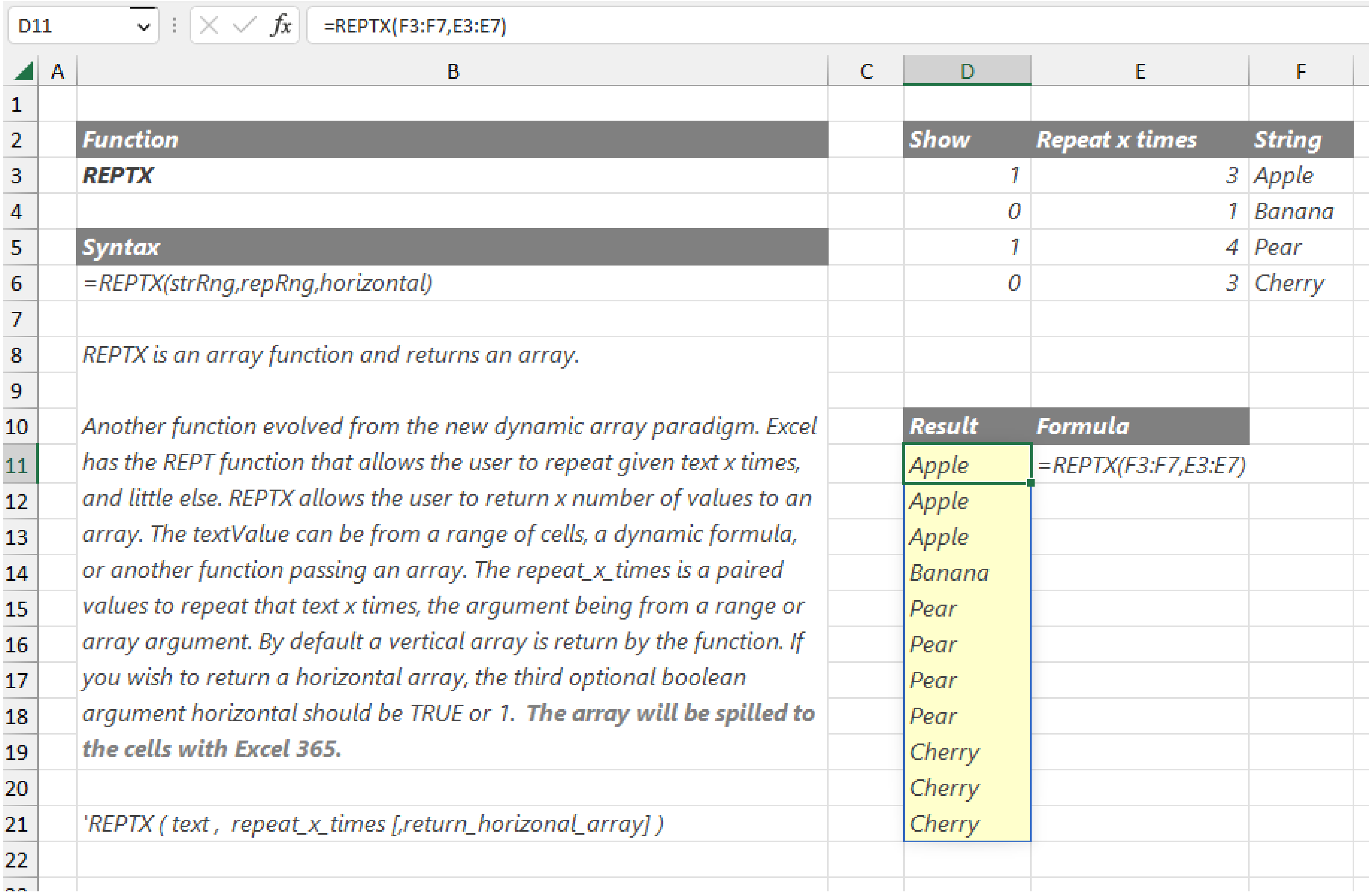


D10

=RANGE_SORT(D3,$D$3:$D$7)

	A	B	C	D	E	F	G
1							
2		***Function***		***Range***			
3		***RANGE_SORT***		*A*			
4				*C*			
5		***Syntax***		*B*			
6		*=RANGE_SORT(range1,rangeArray,descendingFlag)*		*E*			
7				*D*			
8							
9		*This function takes a single cell and a large range of cells and sorts the cells in ascending or descending order. Copy the formula down!*		***Result***	***Formula***		
10				***A***	*=RANGE_SORT(D3,$D$3:$D$7)*		
11				***B***			
12				***C***			
13				***D***			
14				***E***			
15							

REPTX



D11

=REPTX(F3:F7,E3:E7)

Function

REPTX

Syntax

=REPTX(strRng,repRng,horizontal)

REPTX is an array function and returns an array.

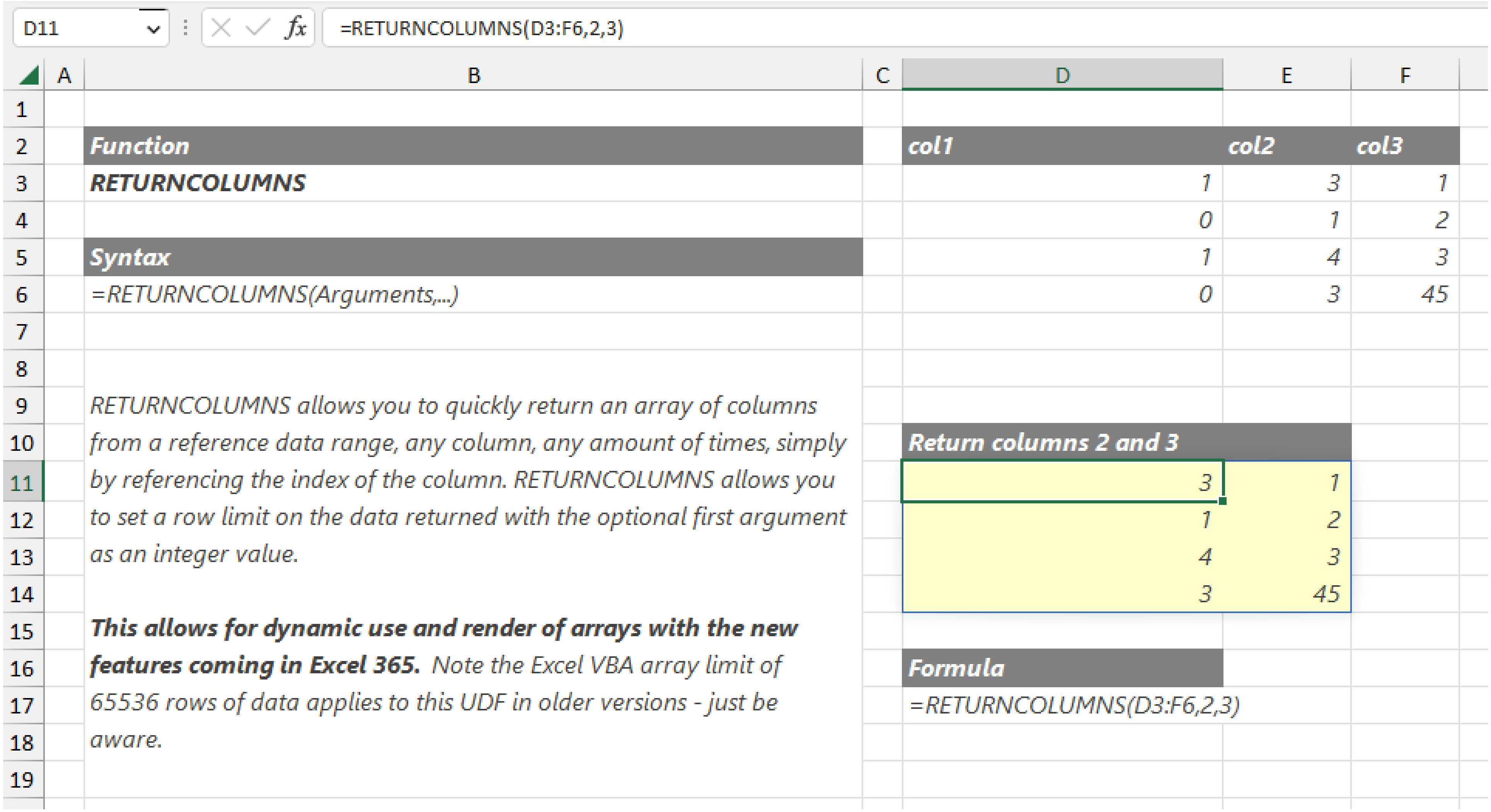
Another function evolved from the new dynamic array paradigm. Excel has the REPT function that allows the user to repeat given text x times, and little else. REPTX allows the user to return x number of values to an array. The textValue can be from a range of cells, a dynamic formula, or another function passing an array. The repeat_x_times is a paired values to repeat that text x times, the argument being from a range or array argument. By default a vertical array is return by the function. If you wish to return a horizontal array, the third optional boolean argument horizontal should be TRUE or 1. ***The array will be spilled to the cells with Excel 365.***

'REPTX (text , repeat_x_times [,return_horizonal_array])

Show	***Repeat x times***	***String***
1	*3*	*Apple*
0	*1*	*Banana*
1	*4*	*Pear*
0	*3*	*Cherry*

Result	***Formula***
Apple	*=REPTX(F3:F7,E3:E7)*
Apple	
Apple	
Banana	
Pear	
Pear	
Pear	
Pear	
Cherry	
Cherry	
Cherry	

RETURNCOLUMNS



=RETURNCOLUMNS(D3:F6,2,3)

Function

RETURNCOLUMNS

Syntax

=RETURNCOLUMNS(Arguments,...)

RETURNCOLUMNS allows you to quickly return an array of columns from a reference data range, any column, any amount of times, simply by referencing the index of the column. RETURNCOLUMNS allows you to set a row limit on the data returned with the optional first argument as an integer value.

This allows for dynamic use and render of arrays with the new features coming in Excel 365. *Note the Excel VBA array limit of 65536 rows of data applies to this UDF in older versions - just be aware.*

col1	*col2*	*col3*
1	*3*	*1*
0	*1*	*2*
1	*4*	*3*
0	*3*	*45*

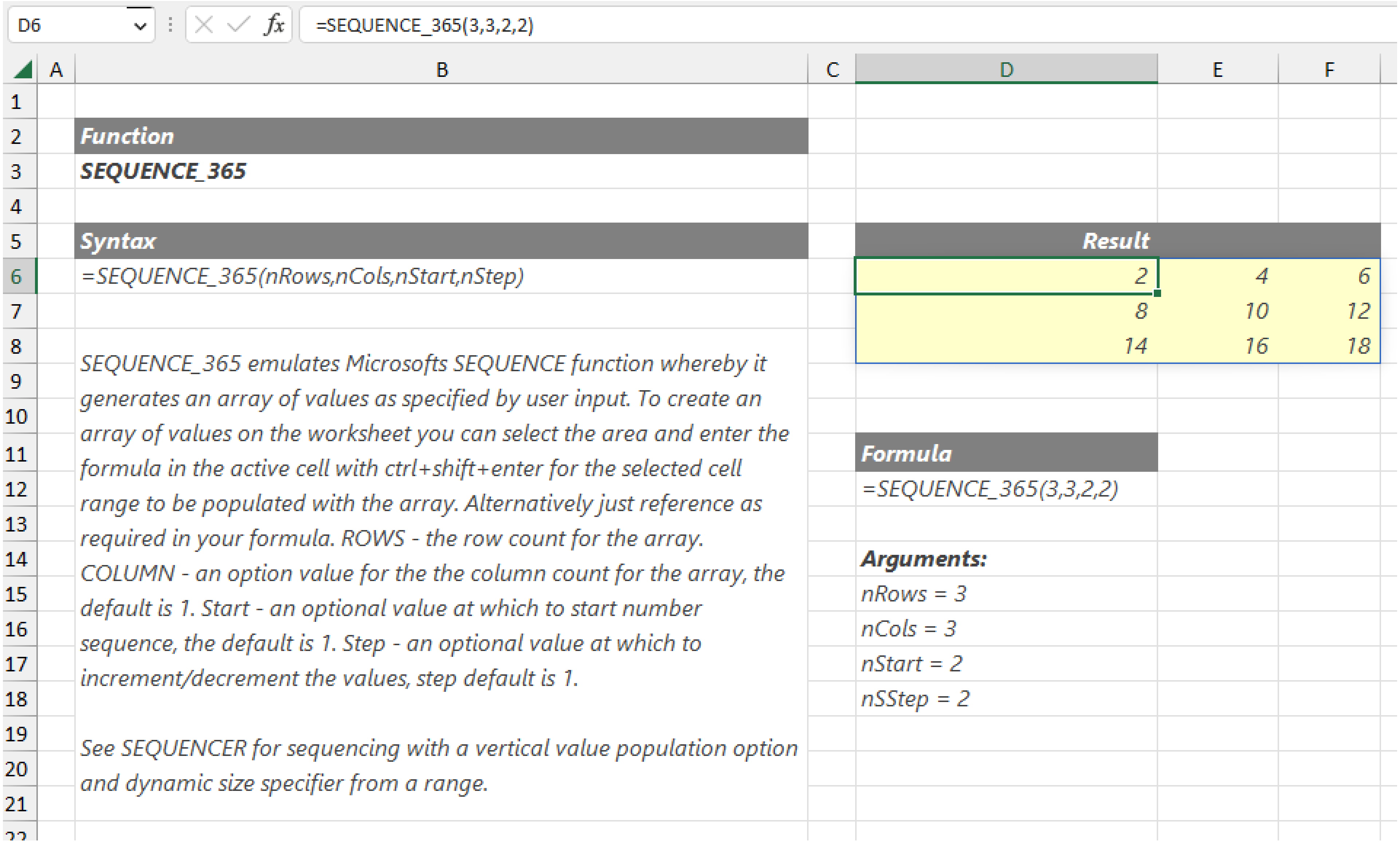
Return columns 2 and 3

3	*1*
1	*2*
4	*3*
3	*45*

Formula

=RETURNCOLUMNS(D3:F6,2,3)

SEQUENCE_365



=SEQUENCE_365(3,3,2,2)

Function

SEQUENCE_365

Syntax

=SEQUENCE_365(nRows,nCols,nStart,nStep)

SEQUENCE_365 emulates Microsofts SEQUENCE function whereby it generates an array of values as specified by user input. To create an array of values on the worksheet you can select the area and enter the formula in the active cell with ctrl+shift+enter for the selected cell range to be populated with the array. Alternatively just reference as required in your formula. ROWS - the row count for the array. COLUMN - an option value for the the column count for the array, the default is 1. Start - an optional value at which to start number sequence, the default is 1. Step - an optional value at which to increment/decrement the values, step default is 1.

See SEQUENCER for sequencing with a vertical value population option and dynamic size specifier from a range.

Result		
2	*4*	*6*
8	*10*	*12*
14	*16*	*18*

Formula

=SEQUENCE_365(3,3,2,2)

Arguments:

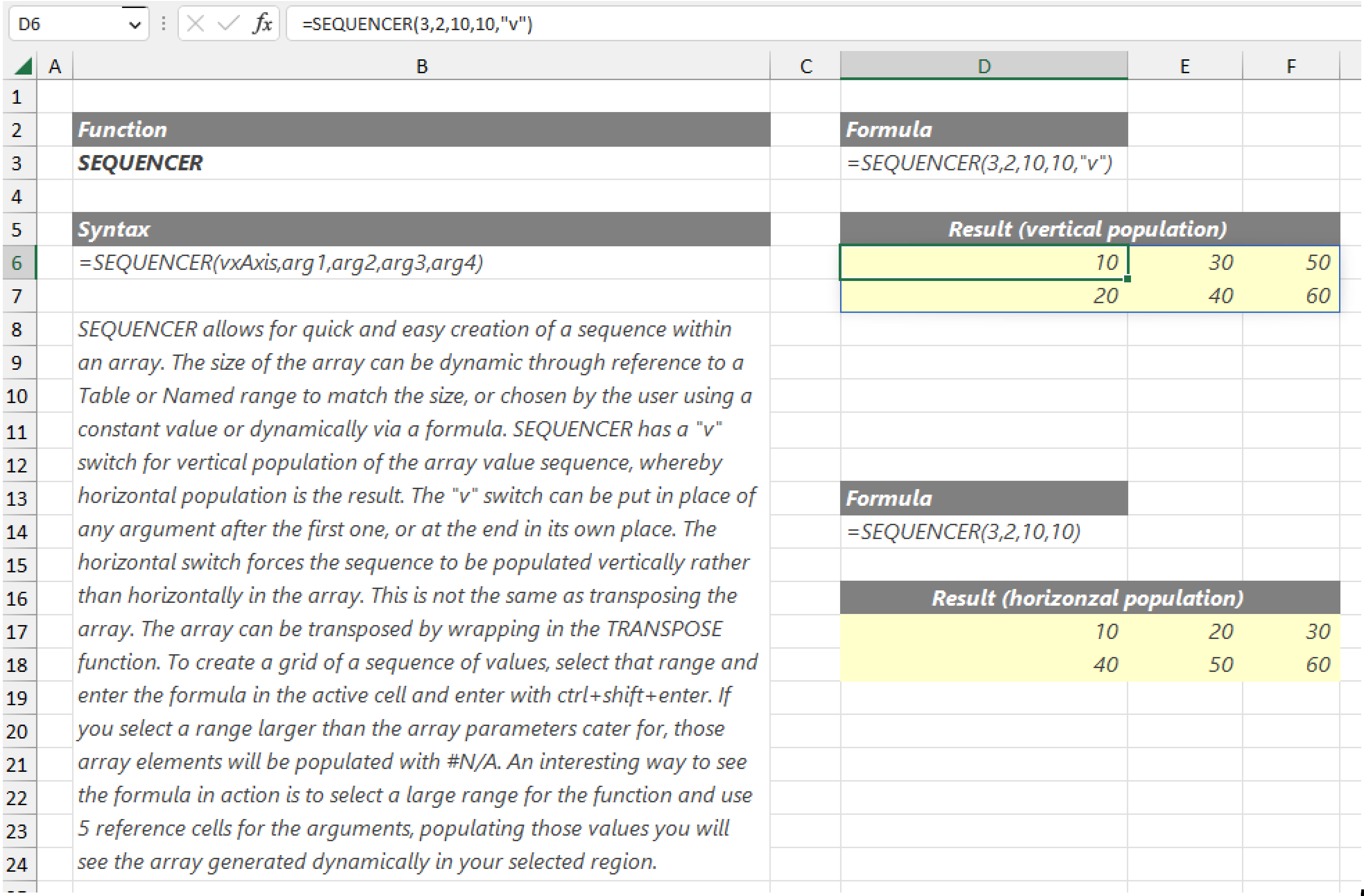
nRows = 3

nCols = 3

nStart = 2

nSStep = 2

SEQUENCER



=SEQUENCER(3,2,10,10,"v")

Function

SEQUENCER

Syntax

=SEQUENCER(vxAxis,arg1,arg2,arg3,arg4)

SEQUENCER allows for quick and easy creation of a sequence within an array. The size of the array can be dynamic through reference to a Table or Named range to match the size, or chosen by the user using a constant value or dynamically via a formula. SEQUENCER has a "v" switch for vertical population of the array value sequence, whereby horizontal population is the result. The "v" switch can be put in place of any argument after the first one, or at the end in its own place. The horizontal switch forces the sequence to be populated vertically rather than horizontally in the array. This is not the same as transposing the array. The array can be transposed by wrapping in the TRANSPOSE function. To create a grid of a sequence of values, select that range and enter the formula in the active cell and enter with ctrl+shift+enter. If you select a range larger than the array parameters cater for, those array elements will be populated with #N/A. An interesting way to see the formula in action is to select a large range for the function and use 5 reference cells for the arguments, populating those values you will see the array generated dynamically in your selected region.

Formula

=SEQUENCER(3,2,10,10,"v")

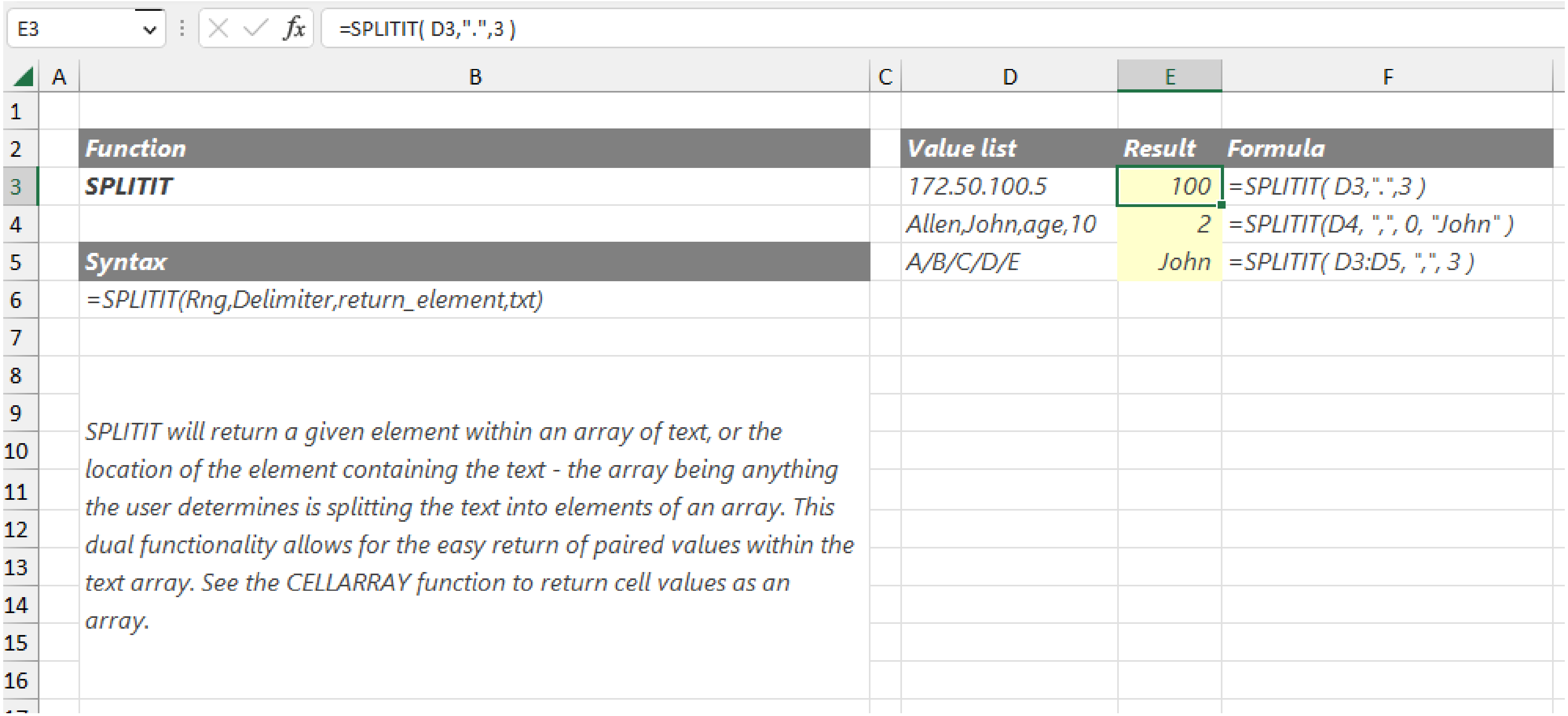
Result (vertical population)		
10	*30*	*50*
20	*40*	*60*

Formula

=SEQUENCER(3,2,10,10)

Result (horizonzal population)		
10	*20*	*30*
40	*50*	*60*

SPLITIT



E3 =SPLITIT(D3,".",3)

Function

SPLITIT

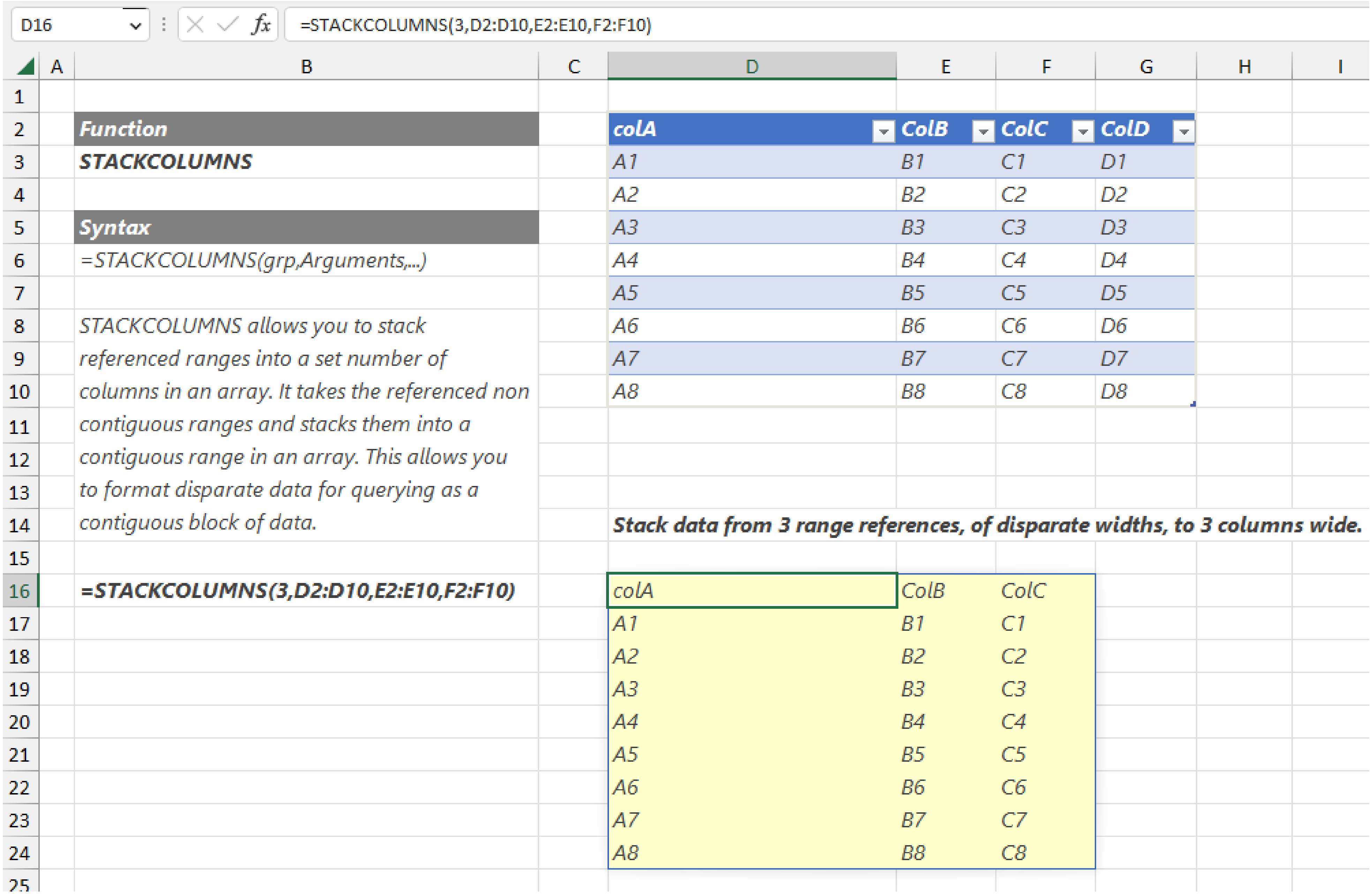
Value list	Result	Formula
172.50.100.5	100	=SPLITIT(D3,".",3)
Allen,John,age,10	2	=SPLITIT(D4, ",", 0, "John")
A/B/C/D/E	John	=SPLITIT(D3:D5, ",", 3)

Syntax

=SPLITIT(Rng,Delimiter,return_element,txt)

SPLITIT will return a given element within an array of text, or the location of the element containing the text - the array being anything the user determines is splitting the text into elements of an array. This dual functionality allows for the easy return of paired values within the text array. See the CELLARRAY function to return cell values as an array.

STACKCOLUMNS



=STACKCOLUMNS(3,D2:D10,E2:E10,F2:F10)

Function

STACKCOLUMNS

Syntax

=STACKCOLUMNS(grp,Arguments,...)

STACKCOLUMNS allows you to stack referenced ranges into a set number of columns in an array. It takes the referenced non contiguous ranges and stacks them into a contiguous range in an array. This allows you to format disparate data for querying as a contiguous block of data.

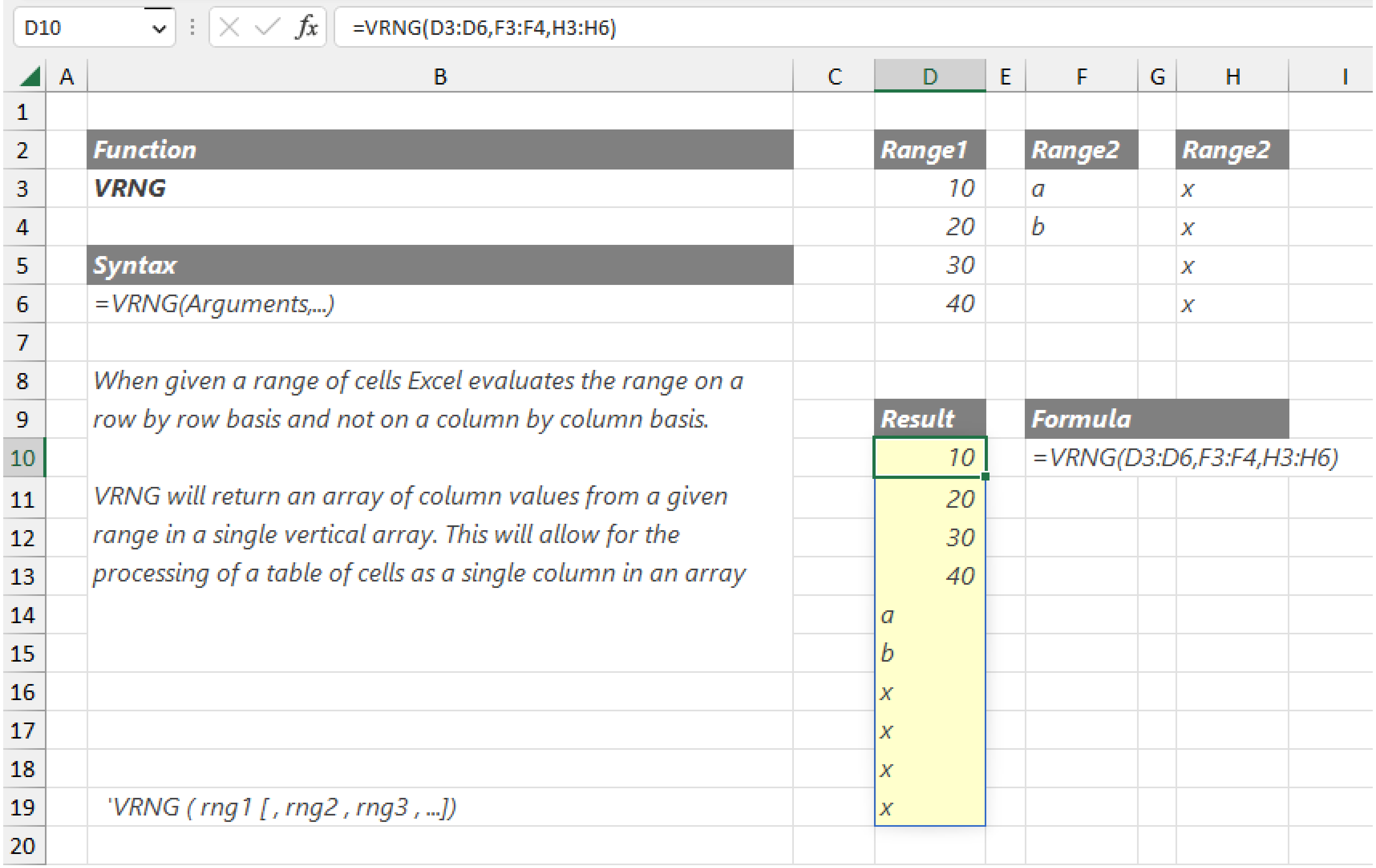
=STACKCOLUMNS(3,D2:D10,E2:E10,F2:F10)

colA	ColB	ColC	ColD
A1	B1	C1	D1
A2	B2	C2	D2
A3	B3	C3	D3
A4	B4	C4	D4
A5	B5	C5	D5
A6	B6	C6	D6
A7	B7	C7	D7
A8	B8	C8	D8

Stack data from 3 range references, of disparate widths, to 3 columns wide.

colA	ColB	ColC
A1	B1	C1
A2	B2	C2
A3	B3	C3
A4	B4	C4
A5	B5	C5
A6	B6	C6
A7	B7	C7
A8	B8	C8

VRNG



D10 =VRNG(D3:D6,F3:F4,H3:H6)

	A	B	C	D	E	F	G	H	I
1									
2		**Function**		**Range1**		**Range2**		**Range2**	
3		***VRNG***		10		a		x	
4				20		b		x	
5		**Syntax**		30				x	
6		=VRNG(Arguments,...)		40				x	
7									
8		*When given a range of cells Excel evaluates the range on a row by row basis and not on a column by column basis.*							
9				**Result**		**Formula**			
10				10		=VRNG(D3:D6,F3:F4,H3:H6)			
11		*VRNG will return an array of column values from a given range in a single vertical array. This will allow for the processing of a table of cells as a single column in an array*		20					
12				30					
13				40					
14				a					
15				b					
16				x					
17				x					
18				x					
19		'VRNG (rng1 [, rng2 , rng3 , ...])		x					
20									