[3.2] Calculate table indices instead of searching

Many functions require searching a table by an index value, then returning the corresponding value, or the two values that bracket an input value. Suppose we have this table:

index	x	y
1	0.5	0.11
2	0.7	0.22
3	0.9	0.33
4	1.1	0.44

If x = 0.7, the search routine should return 0.22. If x = 0.6, then the search routine should return 0.22, 0.33, or both, depending on the purpose of the program. For example, an interpolation routine would need 0.22 and 0.33.

In the most general case, the program searches the x-column data to bracket the desired x-value. Even for short tables and a fast search routine, this is slow. But if the x-data is evenly spaced and the first x-value is known, then it is faster just to calculate the index, like this:

$$\text{index} = \text{int}\left[\frac{x - x1}{dx} + 1\right]$$

where *x1* is the first x-value, *x* is the search target, and *dx* is the x-data spacing. For the table above, x1 = 0.5 and dx = 0.2. *int* is the 89/92+ *int()* function. To find the index for x = 0.8, the calculation is

$$\text{index} = \text{int}\left[\frac{0.8 - 0.5}{0.2} + 1\right] = 2$$

Note that this equation returns the index pointing to the x-value less than the target value.

If the x-data is in ascending order, *dx* is positive. If the x-data is in descending order, the formula works correctly if *dx* is negative.

Once you have found the index, you can find the corresponding table x-value from

$$x = dx(\text{index} - 1) + x1$$

If the target *x* is less than the first x-value in the table, the equation returns zero. If the target *x* is greater than on equal to the last x-value in the table, the equation returns an index larger than the table size. Your program should check for these conditions and handle them appropriately.