Gerald's Column
by Gerald Fitton

This month I shall take a short break from "What will you use your next computer for?" and address a spreadsheet problem which, suddenly, has become very 'popular'!

The Problem

Douglas Groom is the most recent of many people who, over the years, have asked me whether Fireworkz can solve this particular kind of problem for them. Other people have asked for solutions using different spreadsheets including PipeDream, Eureka and Schema.

Typically it consists of grading people (or events), not by the average mark for all their achievements, but the average (or total) of, say, the best four results out of six. I suppose this marking technique allows for the fact that everyone has their 'off days' and some of those 'off days' can be quite seriously bad. A similar technique is sometimes used when analysing a set of statistics; the outliers are ignored and the remainder analysed.

For my example I shall assume that the six results are: (6, 3, 7, 5, 2, 9).
The 'best four' results from these six are (5, 6, 7, 9).
The two lowest results, (2, 3), are rejected.

Of course, this simple example can be extended to more results or a different selection of the raw data, for example a selection of the lowest values. Another variation of this technique is to select, not the 'best' nor the 'worst' but a 'middle cut'. I am sure that once you see how I use just the highest four of six then you will be able to extend the method which I describe to suit your requirements.

The technique I shall describe can be applied to other spreadsheets but, if you do so, then you may find that you have to use different functions from mine because the Fireworkz functions I use might not be available in your chosen spreadsheet.

You may need to write your own functions to place of the "sort" and "index" I shall use.

The Solution

In principle, the solution is to take all the results (six in my example), then sort them so that, the smallest value of the six (in my example) is first and the largest value last. Having sorted the results it is simple to select the 'best' four results and add them together.

The six results of my example, (6, 3, 7, 5, 2, 9), when sorted become (2, 3, 5, 6, 7, 9).
Adding the last four from this sorted array we get 5 + 6 + 7 + 9 = 27.

Having decided what to do in principle, all I have to do now is turn this technique into a spreadsheet! I am using Fireworkz for my example because the majority of the requests I have received ask for a Fireworkz solution.

That is possibly because Fireworkz is available not only on the Iyonix (in 32 bit format) and other older and more recently developed RISC OS machines, but also it is available as Fireworkz for Windows, a version of Fireworkz which runs under Windows.

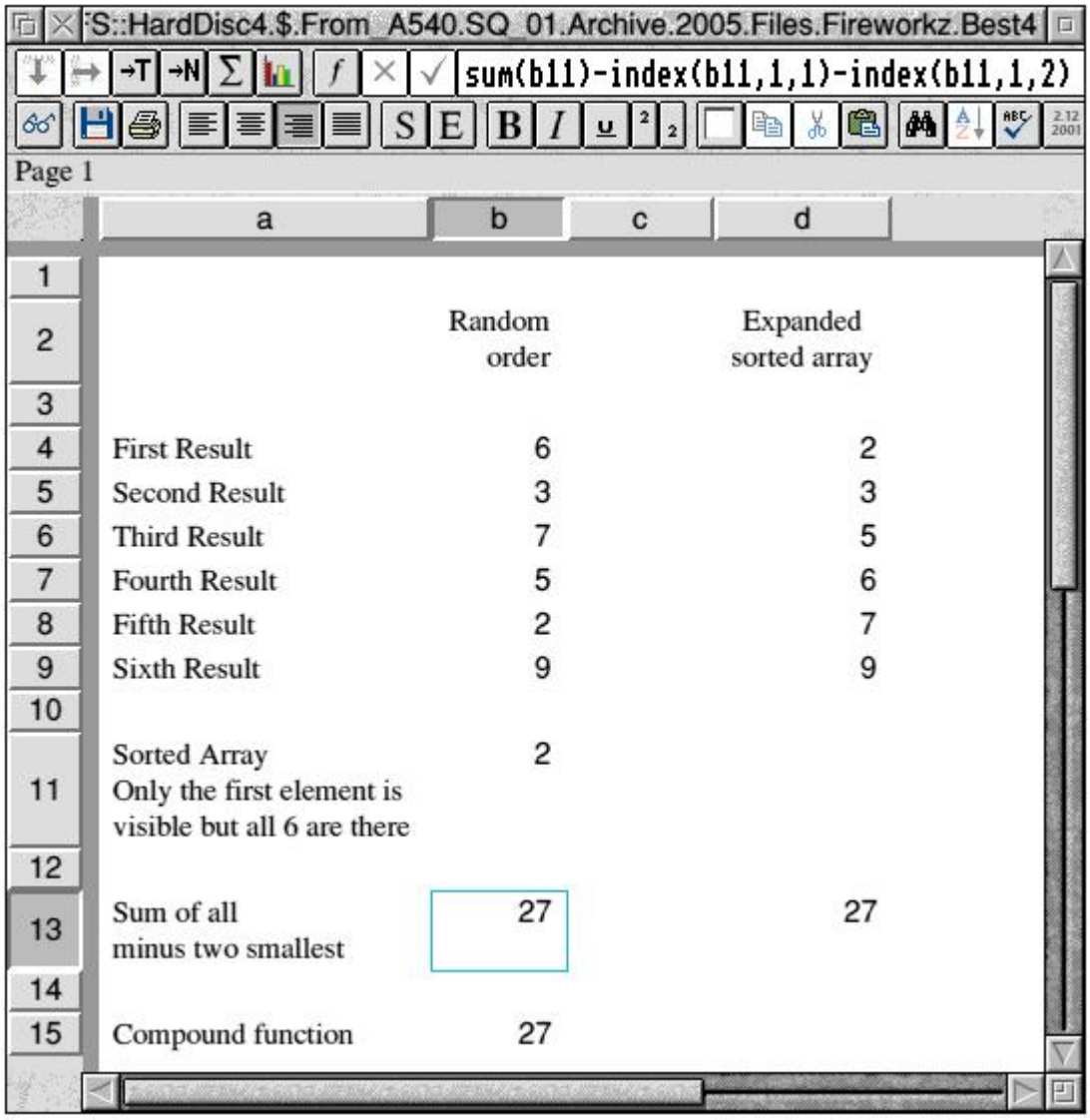
The Bottom Line

The sum of the best 'n' results out of 'm' results in the column array called 'array' is returned by the complicated looking formula sum(index(sort(array),1,(m-n+1),1,m))!

In what follows I shall try to explain how I arrive at this formula.

The File [Best4]

The six results, (6, 3, 7, 5, 2, 9), in random order (or perhaps, chronological order) are in the column array shown in cells (b4b9). In the column array (d4d9) you will see the six results, (2, 3, 5, 6, 7, 9), after they have been sorted in ascending order.



The cell (b11) shows "2" but it contains not only the "2" but also the other five numbers of the sorted array. If you select cell (b11) and then click the <menu> button followed by the command <Edit - Make - Make constant> then the formula shown will change to {2;3;5;6;7;9}.

In the screenshot below you will see this just as it appears in the formula line.

The formula in cell (b13) is shown in the first screenshot. It contains a formula which returns 27, the sum of the four largest values. The formula in (d13) also returns 27, the required answer using the column array of (d4d9).

The formula in (b15) is the formula sum(index(sort(b4b9),1,3,1,4)). This is the compound formula: sum(index(sort(array),1,(m-n+1),1,m)) with array = (b4b9), m = 6 and n = 4. This formula returns the answer we want, namely 27.

The sort(array) function

Cell (b11) contains the formula sort(b4b9); it returns a 'single cell array'.

A 'single cell array' is an array of many numbers all within the same slot.
An array can be a column array, a row array, or a matrix consisting of both rows and columns.

A row array has its elements separated by commas. For example {2,3,5,6,7,9} is a row array.

A column array has its elements separated by semicolons.
For example the array {2;3;5;6;7;9} is a column array.

An array such as {1,2:3,4:5,6;7,8} is a matrix array consisting of 2 columns and 4 rows (eight elements in all). The first row is the row array {1,2}.
The first column is the column array {1;3;5;7;9}.

The function sort(array) sorts the values of 'array' and places the sorted values in a single column array. Hence sort(b4b9), the formula in (b11), returns the single column array {2;3;5;6;7;9}. Only the first element, "2", is visible but, as I explained earlier, all six elements of the array are present within the one cell.

The set_value(destination,source) function

Cell (d4) contains the formula set_value(d4d9,sort(b4b9)). By now (I hope!) we know what sort(array) does. The function set_value(destination,source) can be used to expand an array from a single cell array to an array in which every element is visible. This function, set_value(,), works with any form of array including two dimensional matrix arrays.

In this example, the set_value(,) function expands the sorted array into the column (d4d9).

Having expanded the sorted array into the column (d4d9) it is a simple matter to sum just those values which are wanted into cell (d13).

There are many ways of doing this. I have chosen a method which might be considered a bit circuitous. The formula I have entered into the cell (d13) is not sum(d6d9) (which will return 27) but "sum(d4d9) ™ sum(d4d5)" because it might help you to follow the formula in (b13).

The index(array,col,row) function

The formula in (b13) is "sum(b11) - index(b11,1,1) - index(b11,1,2)".

The formula sum(b11) sums every value of the sorted array in (b11).
On its own it would return the value 36 because 2 + 3 + 5 + 6 + 7 + 9 = 36.

The formula index(b11,1,1) returns 2, the first element of the sorted array.
The formula index(b11,1,2) returns 3, the second element of the sorted array.

The first argument of the three in index(b11,1,2) is the array (this can be a single cell array).
The second argument is the column number.
Since there is only one column this number is "1".
The third argument is the number of the row within the array.
The number "2" picks up the second element in the array; this is a "3".

Hence the formula in (b13) returns 36 - 2 - 3 = 27.

The index(array,col,row,cols,rows) function

This function has five and not three arguments like the index(array,col,row) function.

The extra two arguments allow us to 'collect' from the array not one single element but a whole array of adjacent elements. The argument "cols" is the number of columns and the argument "rows" are the number of columns and rows 'collected' from the array.

Using as our example our array (b4b9) is a '1 column 6 row' array as is the array sort(b4b9).
What we want to do is to sort the array (into a single column array) and then 'collect' the last four elements (rows) of the first and only column. So, when we use the five argument version of index(array,col,row,cols,rows) we put "cols" = 1 and "rows" = 4.

The array which we want inside index(array,col,row,1,4) is not the original array which is in cells (b4b9) but the sorted array returned by the function sort(b4b9).

The values used for "col" and "row" specify the top left element of the sub-array which is collected by the index(array,col,row,1,4) function. We want to start at the third element so we chose "row" = 3.

Look in cell b15 and you will find the formula sum(index(sort(b4b9),1,3,1,4).
This formula sums the values in the array returned by index(sort(b4b9),1,3,1,4).

The final answer is 27!

Summary

The sum of the highest four values in the array (b4b9) is returned by the compound formula sum(index(sort(b4b9),1,3,1,4)).
There is no need to generate an extensive spreadsheet as I have done in the file [Best4].
The one single rather awesome (at first glance) function is all that is required.

Communication

Thanks for all your emails, letters and discs.

These days I am finding an increasing number of my correspondents are sending me CDs they have burned rather than floppies. That is because the files they are sending me are much larger than can be fitted onto a floppy. I guess it is a sign of the times?

I am still on a dial up Internet connection so please don't send me huge files as an email attachment. My filter may reject it!

Please contact me by email (preferred) or by letter if you have any questions or comments. You can email me at <Archive@abacusline.demon.co.uk>.