

## Determinant    $u . v$    2                Dot Prod

<p>The phrases <code>-/ . *</code> and <code>+/ . *</code> are the determinant and permanent of square matrix arguments. More generally, the phrase <code>u . v</code> is defined in terms of a recursive expansion by minors along the first column, as discussed below.</p>	<p>For vectors and matrices, the phrase <code>x +/ . * y</code> is equivalent to the dot, inner, or matrix product of math; other rank-0 verbs such as <code>&lt; .</code> and <code>* .</code> are treated analogously. In general, <code>u . v</code> is defined by <code>u@(v"(1+1v,_) )</code>, restated in English below.</p>
---	--

For example:

```
x=: 1 2 3 [ m=: >1 6 4;4 1 0;6 6 8
det=: -/ . * [ . mp=: +/ . *
x ([ ; ] ; det@] ; mp ; mp~ ; mp~@]) m
+-----+-----+-----+-----+-----+
| 1 2 3 | 1 6 4 | _112 | 27 26 28 | 25 6 42 | 49 36 36 |
|       | 4 1 0 |      |       |       | 8 25 16 |
|       | 6 6 8 |      |       |       | 78 90 88 |
+-----+-----+-----+-----+-----+

```

The monad `u . v` is defined as illustrated below:

```
DET=: 2 : 'v./@,`({."1 u. . v. $:@minors)@.(0:<{:@$) @ ,. "2'
minors=: }. "1 @ (1&([\..))
-/ DET * m
_112
-/ DET * 1 16 64
49
-/ DET * i.3 0
1
+/ DET * m
320

```

The definition `u@(v"(1+1v,_) )` given above for the dyadic case may be re-stated in words as follows: `u` is applied to the result of `v` on lists of “left argument cells” and the right argument in toto. The number of items in a list of left argument cells must agree with the number in the right argument. Thus, if `v` has ranks `2 3` and the shapes of `x` and `y` are `2 3 4 5 6` and `4 7 8 9 10 11`, then there are `2 3` lists of left argument cells (each shaped `4 5 6`); and if the shape of a result cell is `sr`, the overall shape is `2 3, sr`.