



▶ Version 10.1

▶ General Enhancements

▶ Function Header Namelists

Function Header Namelists

In APL, the argument to a function is passed as a single array. So if you want to write a function that takes more than one parameter, you must unpack the individual items from the argument into separate variables using assignment. Typically, you also need to remember to localise these names.

For example, if you want to write a function that takes a date expressed as (Year Month Day), you will probably have written it like this:

```

      ▽ R←FOO Date;Year;Month;Day
[1]   Year Month Day←Date
      ...
      ▽
  
```

and call it with an expression such as:

```
FOO 2004 6 12
```

In this case, the local variable *Date* is really just a dummy name whose sole purpose is to act as a temporary store for the single 3-element array that is passed to the function when it is called.

In Version 10.1, the header syntax for defined functions has been extended to allow you to automatically assign the elements of the right argument to local variables

So the function may now be defined as:

```

      ▽ R←FOO (Year Month Day)
[1]   ...
      ▽
  
```

Similarly, the result of a function must be a single array, so if you want to return an number of items you are obliged to collect the items together before the function returns its result. This is potentially even more onerous if the function has more than one logical exit point.

```

      ▽ R←FOO1 IDN;Year;Month;Day
[1]   ⍺ Calculate Year, Month, day from IDN
      ...
[n]   R←Year Month Day
      ▽
  
```

The new header syntax also caters for this requirement, and the same function may now be expressed as follows:

Dyalog Ltd

South Barn
Minchens Court
Minchens Lane
Bramley
Hampshire
RG26 5BH
United Kingdom

Phone:

+ 44 (0) 1256 830 030

Fax:

+ 44 (0) 1256 830 031

e-mail:

sales@dyalog.com



▶ Version 10.1

▶ General Enhancements

▶ Function Header
Namelists

```

▽ (Year Month Day)←F001 IDN
[1]  A Calculate Year, Month, day from IDN
    ...
[n]

```

▽

Combining both of these features together, it is possible to write a function `AddDays` which takes a 3-element right argument of (Year Month Day), and returns a 3-element result of (Year Month Day) as follows:

```

▽ (Year Month Day)←Days AddDays (Year Month
Day)
[1]  A Add Days to YMD Date
    ...
▽
    18262 AddDays 1949 4 30
1999 4 30

```

Dyalog Ltd

South Barn
Minchens Court
Minchens Lane
Bramley
Hampshire
RG26 5BH
United Kingdom

Phone:

+ 44 (0) 1256 830 030

Fax:

+ 44 (0) 1256 830 031

e-mail:

sales@dyalog.com