Subject: Re: [Q]: ID analog to FORTRAN "sign" function
Posted by Dick Jackson on Thu, 12 Oct 2000 07:00:00 GMT
View Forum Message <> Reply to Message

"Mark Hadfield" <m.hadfield@niwa.cri.nz> wrote in message
news:971127957.784431@clam-ext...
> "Alex Schuster" <alex@pet.mpin-koeln.mpg.de> wrote in message
> news:39E1C067.16F7488E@pet.mpin-koeln.mpg.de...
>> Rostyslav Pavlichenko wrote:
>>
>>> Does the IDl have something close to Fortran SIGN (DSIGN... so on...)
>>> functions
>>> ...
>>> result = SIGN (a, b)
>>> a (Input) Must be of type integer or real.
>>>
>>> b Must have the same type and kind parameters as a.
>>>
>>> Results:
>>> =========
>>> The result type is the same as a.
>>> The value of the result is
>>> | a | if b >= zero
>>> and -| a | if b < zero.
>>> ...
>> No, but you can easily write it:
>>
>> function sign, a, b
>> if (b ge 0) then $
>> return, abs(a) $
>> else $
>> return, -abs(a)
>> end
>
> The following is more compact and works when b is an array
>
> return, abs(a) * (fix(b ge 0) - fix(b lt 0))

Do I dare offer one more? Subscript lookups seem faster than arithmetic operations, making this one faster, more compact and no less cryptic! :-)

```
Return, Abs(a) * ([-1, 1])[b GE 0]
```

A bit faster still, if you know the expected type of a and b, to avoid an extra type conversion:

```
Return, Abs(a) * ([-1.0, 1.0])[b GE 0]
```

or

```
   Return, Abs(a) * ([-1.0D, 1.0D])[b GE 0]
```

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / Voice/Fax: +1-403-242-7398
