Subject: Re: Double precision data into caldat
Posted by Jim Pendleton on Tue, 08 Sep 2015 21:18:56 GMT
View Forum Message <> Reply to Message

On Tuesday, September 8, 2015 at 2:01:09 PM UTC-6, David Fanning wrote:
> Larry H. writes:
>
>> I'm trying to convert some Julian dates back into standard dates using
> caldat. (In fact, I am testing what I got out of julday in the first
> place.) There should be hours and minutes in the results. If I take
> the output of julday as a variable, say TEST, and put it into caldat as
>>
>> caldat, test, m, d, y, h, mm, s
>>
>> I get the right answer. However, if I just use the actual value of test, I get odd results. So, for the Julian day 2456658.56250000, I should get
>>
>> 2014 1 1 1 30 0
>>
>> in year, month, day, hour, min, sec form. If I use
>>
>> caldat, 2456658.56250000D, m, d, y, h, mm, s
>>
>> the results are correct, but if I do the type conversion using double(), it doesn't. So I have
>>
>> caldat, double(2456658.56250000), m, d, y, h, mm, s
>>
>> and
>>
>> p = double(2456658.56250000)
>> caldat, p, m, d, y, h, mm, s
>>
>> both giving the result
>>
>> 2014 1 1 0 0 0
>>
>> Does anyone know why this is? As far as I know, all of those input values are the same.
>
> Some required reading for all those who work with IDL:
>
> http://www.idlcoyote.com/math_tips/sky_is_falling.html
>
>
>> PS Why on earth does IDL use month, day, year rather than year, month, day in both caldat and julday??
>
> I believe this was an attempt to make IDL "unique" among programming
> languages. It has infuriated some and made IDL endearing to others. ;-)

>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

I'm going to recommend a recently published book by my colleague Ron Kneusel, "Numbers and Computers", published by Springer and available on Amazon. It's like the "Sky is Falling" essay, but on steroids.

Ron goes into great depth on the topic of how computers represent data such as integers, both big and small, fixed point and floating point numbers, rational numbers, etc.

He also discusses how mathematical operations from basic comparisons to trigonometric functions are frequently implemented. All this is described at a fundamental bit-twiddling level.

Though there are no IDL examples (what's up with that?) there are full algorithmic implementations in C, on which IDL is largely based, and frequently in Python as well.

Also see numbersandcomputers.com.

Jim P.
