Subject: Re: Set Precision???????????
Posted by Craig Markwardt on Mon, 14 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Anil Kochhar <anilk@mtolympus.ari.net> writes:
>
> Hi All
>
> I'd like create a varible which always holds numbers out to 6 places
> passed the decimal point, and 4 numbers before the decimal point (e.g.
> 1999.123456). I tried double
> precision but this seems to only hold 8 digits total for decimal numbers.
> However I would like to create a variable corresonding to a fraction of a
> Year (e.g 1995.123456). I have not been able to find a procedure allowing
> me to create 10 digit decimal number , without the number being rounded
> off to 8 digits.
> Does anyone know of a way to declare a variable to hold a 10 digit
> decimal number?

You should make a distinction between the *internal precision* of a
floating point or double precision number, and the *textual
representation* of that number. Both are quite different.

Internally, double precision maintains about 16 decimal digits in the
mantissa irrespective of the position of the decimal point, which is
plenty more than you need. [ Floating point keeps about 7 decimal
digits, probably not enough for you. ]

The important thing to keep in mind is that IDL doesn't always print
this many digits when it prints a number. I'm sure this is primarily
for convenience, since people probably don't want huge multi-digit
output on their screen most of the time.

By *default*, IDL prints 8 digits before the decimal, and 7 after the
decimal (at least it did in my simple trial). If you want to have a
different output format, then you need to tell IDL explicitly how to
format it using the FORMAT keyword to PRINT, PRINTF or STRING. That's
too complicated a subject to write about here but I can get you
started.

FORMAT strings are similar in spirit to IDL format statement in
FORTRAN, or the format string in C. For double precision output
formats, you describe (a) the total record length, and (b) the number
of digits after the decimal. In your case, the total record length is
11 ( = 4 year digits + 1 decimal point digit + 6 fractional digits),
and the number of digits after the decimal is 6. The format string is
thus 'D11.6'. Here is how that goes together in an IDL print
statement.

```
IDL> y = 1999.123456D
IDL> print, y
      1999.1235
IDL> print, y, format='(D11.6)'
1999.123456
```

Voila!

Craig

--
 ------------------------------------------------------------ --------------
Craig B. Markwardt, Ph.D.         EMAIL: craigmnet@astrog.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 ------------------------------------------------------------ --------------

## Subject: Re: Set Precision???????????
Posted by Liam Gumley on Mon, 14 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Anil Kochhar wrote:
> I'd like create a varible which always holds numbers out to 6 places
> passed the decimal point, and 4 numbers before the decimal point (e.g.
> 1999.123456). I tried double
> precision but this seems to only hold 8 digits total for decimal numbers.
> However I would like to create a variable corresonding to a fraction of a
> Year (e.g 1995.123456). I have not been able to find a procedure allowing
> me to create  10 digit decimal number , without the number being rounded
> off to 8 digits.
> Does anyone know of a way to declare a variable to hold a 10 digit
> decimal number?

I think you are confusing internal machine precision with print formatting precision. For example, to print the double precision PI system variable:

```
IDL> print, !dpi
      3.1415927
IDL> print, !dpi, format='(e20.10)'
   3.1415926536e+00
```

The default print format only prints 8 digits, but you can change the print format to print more digits. On machines with IEEE arithmetic, 64 bit double precision stores about 15 digits of information. For a nice

discussion of how floating point numbers are represented, see 'Section 4
- Floating-point numbers' at

http://metalab.unc.edu/pub/languages/fortran/unfp.html

Cheers,
Liam.

--
Liam E. Gumley
Space Science and Engineering Center, UW-Madison
http://cimss.ssec.wisc.edu/~gumley

---