JUST ASK!

Question: Last week I got a call from someone who told me I should not use the modulo operator because the definition is wrong and I could end up doing serious damage. Is there a problem with modulo?

Editors: This question has come up occasionally over the years and appears to stem from a combination of ignorance and insecurity.

In 1992, the MDC received a request to "fix the modulo problem." As part of its debate, the committee asked for a comparison of the different modulo calculations in the industry, which Frederick Hiltz, Ph.D., prepared. (Dr. Hiltz is also a review board member for this journal.) The results are as follows:

```
X#Y=X-([X/Y]*Y)              IF Y'=0
     divide by zero error    IF Y=0
     where [a] = the largest
                 integer'>a
```

Ada
BASIC
Common Lisp
Fortran 90 (MODULO)
M

```
X#Y=X-([X/Y]*Y)               IF Y>0
     divide by zero error     IF Y=0
     undefined error          IF Y<0
     where [a] = the largest
                 integer'>a
```

Modulo-2
Pascal

```
X#Y=X-([X/Y]*Y)               IF Y'=0
     0                        IF Y=0
     where [a] = the largest
                 integer'>a
```

PL/1

```
X#Y=X-([X/Y]*Y)               IF Y'=0
     X                        IF Y=0
     where [a] = the largest
                 integer'>a
```

APL

```
X#Y=X-([X/Y]*Y)               IF Y'=0
     divide by zero error     IF Y=0
     where [a] = the integer part
                 of a, rounded
                 towards 0
```

FORTRAN 77
FORTRAN 90 (MOD)
Lisp

```
X#Y = machine-dependent
```

C

```
X#Y = implementation-specific
```

Prolog

As you can see, there is no consensus within the computer community about the correct calculation of modulo. As always, the real danger in using any tool is in not fully understanding what it does. When you design an algorithm using $PIECE, you do so understanding how the function works and what it will return under the limitations within which the algorithm will be applied. Do the same thing with modulo.

Question: We just started to replace our minicomputers with personal computers. Both the mini vendor and the PC vendor claim that their M is standard, but it looks as if we are going to have to rewrite all device-handling because the OPEN and USE commands are so different between the two. They can't both be standard; who's wrong?

Editors: We have not included the details of your question because the fact is that both are standard. There are several places within the standard where implementors are free to include any extensions they feel are necessary, useful, or desirable for the application programmer. Your question focused on the device parameters, all of which are defined by the implementor, not the standard.

The best way to avoid difficult conversions that you have encountered is to isolate the OPEN, USE, and CLOSE commands into a central utility anytime the nonstandard device parameters are needed. While this will be of little comfort to you for your present conversion, now is an excellent time to add the utility and prevent a repeat in the future.

Speaking of the future, the next standard will standardize device parameters to some extent. All of the current code will continue to work as it does now. A new feature, mnemonicspace, will be added, however. Anytime a mnemonicspace is in effect for the current device, the device parameters will be specified by the standard. Vendors will be able to expand

on the standardized device parameters in the same way they now add commands and functions—by using the z . . . namespace.

Question: I have attended our Association's Annual Meeting for several years now. While the tutorials, roundtable discussions, and so on are very informative and useful, I want to do more. How can I get involved?

Editors: You just did. Let me count the ways . . . Getting involved in the Annual Meeting is as simple as volunteering to work on the Program Committee. Each year, there is a planning meeting for those interested in working on next year's program. If you cannot attend the Annual Meeting, a phone call to the M Technology Association office will get you on the list of volunteers. At the planning meeting, volunteers can tell us about their preferences for serving on subcommittees, or about their special skills or expertise the committee could tap. The Program Chair and Co-Chair then assign volunteers to subcommittees based on where their skills can best be used. Each of the subcommittees undertakes tasks throughout the year to plan the program for the Annual Meeting. You could help decide what tutorials will be slated for 1994, or find and recruit new exhibitors, or review technical papers for publication. Your time and contributions to the Program Committee are what make MTA meetings so well attended and well received by the M community.

Plan to attend the 1994 Planning Meeting at the Annual Meeting, June 21-25, 1993, at the Washington Hilton and Towers in Washington, D.C. Or call the MTA office and get on the list of volunteers for next year's committee. M

Advertisers' Index

This index appears as a service to our readers. The publisher does not assume any liability for errors or omissions.