NAME

qtod − convert Fortran program from quadruple-precision to double-precision

SYNOPSIS

qtod [**−−?**] [**−−author**] [**−−copyright**] [**−−help**] [**−−version**] < infile >outfile

qtod copies its standard input to standard output, converting Fortran quadruple-precision constants, built-in functions, and type declarations to double precision.

Floating-point FORMAT specifications are left intact; on some ancient systems, they may require modifications. They do *not* under the rules of Fortran 77.

Leading tabs are correctly interpreted according to common extended Fortran rules.

qtod recognizes all of the standard Fortran 77 double-precision functions, quadruple-precision extensions, the pair **qrand/drand** (UNIX pseudo-random number generators), and the pair **q1mach/d1mach** from the PORT library framework.

Although quadruple precision is not provided for by the Fortran 77 Standard, a number of vendors, including DEC, IBM, and Sun support it with data types **REAL*16** and **COMPLEX*32**, together with a set of functions analogous to the double-precision Fortran functions, with the letter *D* in their names changed to *Q*. Quadruple-precision constants use the letter *Q* instead of *D* or *E*. *Q* format items already mean something else in some of these extended Fortrans, so *D* or *E* must be used instead.

qtod's other purpose is to demonstrate a modest **lex**(1) program.

OPTIONS

Options can be prefixed with either one or two hyphens, and can be abbreviated to any unique prefix. Thus, **−v**, **−ver**, and **−−version** are equivalent.

All options in this program are diagnostic, and suppress processing of the input stream. Execution terminates with a success return code after processing one or more options, but unrecognized options cause immediate termination with a failure return code.

−−? Same as **−−help**.

−−author Display a brief author credit on *stdout*.

−−copyright Display copyright and license information on *stdout*.

−−help Display a brief help message on *stdout*, giving a usage description.

−−version Display the program version number and release date on *stdout*.

BUGS

Undeclared variables are not type-converted. To find such instances, use the Extended PFORT Verifier, **pfort**(1), or the Fortran checker, **ftnchek**(1). Some UNIX Fortran compilers have a compile-time option, usually called *−u*, to flag undeclared variables.

Text beyond column 72 is discarded when lines are collected into Fortran statements.

qtod does not handle embedded ASCII tab characters correctly when long lines are to be broken. A Fortran-sensitive detabbing utility should be applied first if the input file possibly contains embedded tabs. Note that **expand**(1) *cannot* be used to do this job correctly!

Mixed-precision code may not be converted correctly. For example, **QEXT(DFLOAT(N))** will become **DBLE(DFLOAT(N))**, which is syntactically incorrect.

Functions and variables of type **COMPLEX** are not converted, because Fortran 77 does not define a quadruple precision complex type. Complex constants will be converted, however, since their real and imaginary parts look like normal floating-point values.

SEE ALSO

dtoq(1), **dtos**(1), **ftnchek**(1), **lex**(1), **pfort**(1), **stod**(1).

AUTHOR

Nelson H. F. Beebe
Department of Mathematics
University of Utah
Salt Lake City, UT 84112-0090
USA
Tel: +1 801 581 5254
FAX: +1 801 581 4148
Email: beebe@math.utah.edu, beebe@acm.org, beebe@computer.org
Web: http://www.math.utah.edu/~beebe/