

! This is a test program for version 1.4 of module FM_QUAD_INT, which contains the interface routines allowing quadruple-length integer variables in the user's program to be used in assignments, arithmetic, and comparisons involving type (fm), (im), and (zm) variables. The same operations are provided as those in the basic module FMZM for single or double precision variables.

! All of the routines in module FM_QUAD_INT are tested, and if all tests are completed successfully, this line is printed:

! 280 cases tested. No errors were found.

```
MODULE TEST_VARS
```

```
USE FMVALS
```

```
USE FMZM
```

```
USE FM_QUAD_INT
```

```
TYPE (FM), SAVE :: M_A, MFM1, MFM2, MFM3, MFM4, MFM6, &  
                  MFMV1(3), MFMV2(3), &  
                  MFMA(3,3), MFMB(3,3)
```

```
TYPE (IM), SAVE :: M_J, MIM1, MIM2, MIM3, MIM4, MIM5
```

```
TYPE (IM), SAVE, DIMENSION(3) :: MIMV1, MIMV2
```

```
TYPE (IM), SAVE, DIMENSION(3,3) :: MIMA2, MIMB2
```

```
TYPE (ZM), SAVE :: M_Z, MZM1, MZM2, MZM3, MZM4, MZM5, &  
                  MZMV1(3), MZMV2(3), &  
                  MZMA2(3,3), MZMB2(3,3)
```

```
INTEGER (QUAD_INT), SAVE :: QI1, QI2, QI3, QI4, QI5, QIV(3), QIV2(3,3)
```

```
REAL, SAVE :: RV(3), RV2(3,3)
```

```
DOUBLE PRECISION, SAVE :: DV(3), DV2(3,3)
```

```
COMPLEX, SAVE :: CV(3), CV2(3,3)
```

```
COMPLEX (KIND(0.0D0)), SAVE :: CDV(3), CDV2(3,3)
```

```
INTEGER, SAVE :: KLOG, KWSAVE, NCASE, NERROR
```

```
REAL, SAVE :: TIME1, TIME2
```

```
END MODULE TEST_VARS
```

```
MODULE TEST_A
```

```
USE TEST_VARS
```

```
CONTAINS
```

```
SUBROUTINE TEST1
```

```
!           Test the = assignment interface.
```

```
IMPLICIT NONE
```

```
WRITE (KW, "(/' Testing the derived type = interface.')")
```

```
NCASE = 1
```

```
QI4 = MFM1
```

```
IF (QI4 /= 581) CALL PRterr(KW)
```

```

NCASE = 2
QI4 = MIM1
IF (QI4 /= 661) CALL PRterr(KW)

NCASE = 3
QI4 = MZM1
IF (QI4 /= 731) CALL PRterr(KW)

NCASE = 4
MFM3 = QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_SUB(MFM3,MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
CALL FM_ABS(MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
CALL FM_ST2M('0',MFM3)
IF (FM_COMPARE(MFM4,'GT',MFM3)) CALL PRterr(KW)

NCASE = 5
MFM3 = QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_SUB(MFM3,MFM4,MFM6)
CALL FM_EQU(MFM6,MFM4,NDIG,NDIG)
CALL FM_ABS(MFM4,MFM6)
CALL FM_EQU(MFM6,MFM4,NDIG,NDIG)
CALL FM_ST2M('0',MFM3)
IF (FM_COMPARE(MFM4,'GT',MFM3)) CALL PRterr(KW)

NCASE = 6
MFM3 = QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_SUB_R2(MFM3,MFM4)
CALL FM_ABS(MFM4,MFM6)
CALL FM_EQU(MFM6,MFM4,NDIG,NDIG)
CALL FM_ST2M('0',MFM3)
IF (FM_COMPARE(MFM4,'GT',MFM3)) CALL PRterr(KW)

NCASE = 7
MFM3 = QI2
MFM4 = TO_QUAD_INT(MFM3)
CALL FM_SUB_R2(MFM3,MFM4)
CALL FM_ABS(MFM4,MFM6)
CALL FM_EQU(MFM6,MFM4,NDIG,NDIG)
CALL FM_ST2M('0',MFM3)
IF (FM_COMPARE(MFM4,'GT',MFM3)) CALL PRterr(KW)

NCASE = 8
MFM3 = QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_EQU(MFM3,MFM6,NDIG,NDIG)
CALL FM_SUB_R1(MFM6,MFM4)
CALL FM_EQU(MFM6,MFM4,NDIG,NDIG)
CALL FM_ABS(MFM4,MFM6)
CALL FM_EQU_R1(MFM6,NDIG,NDIG)
CALL FM_EQ(MFM6,MFM4)
CALL FM_ST2M('0',MFM3)
IF (FM_COMPARE(MFM4,'GT',MFM3)) CALL PRterr(KW)

```

```
NCASE = 9
MIM3 = QI2
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_SUB(MIM3,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
CALL IM_ST2M('0',MIM3)
IF (IM_COMPARE(MIM4,'GT',MIM3)) CALL PRterr(KW)
```

```
NCASE = 10
MIM3 = TO_IM(QI2)
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_SUB(MIM3,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
CALL IM_ST2M('0',MIM3)
IF (IM_COMPARE(MIM4,'GT',MIM3)) CALL PRterr(KW)
```

```
NCASE = 11
MIM3 = TO_IM(QI2)
MIM4 = TO_QUAD_INT(MIM3)
CALL IM_SUB(MIM3,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
CALL IM_ST2M('0',MIM3)
IF (IM_COMPARE(MIM4,'GT',MIM3)) CALL PRterr(KW)
```

```
NCASE = 12
MZM3 = QI2
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_SUB(MZM3,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
CALL ZM_ABS(MZM4,MFM4)
CALL FM_ST2M('0',MFM3)
IF (FM_COMP(MFM4,'GT',MFM3)) CALL PRterr(KW)
```

```
NCASE = 13
MZM3 = TO_ZM(QI2)
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_SUB(MZM3,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
CALL ZM_ABS(MZM4,MFM4)
CALL FM_ST2M('0',MFM3)
IF (FM_COMP(MFM4,'GT',MFM3)) CALL PRterr(KW)
```

```
NCASE = 14
MZM3 = TO_ZM(QI2)
MZM4 = TO_QUAD_INT(MZM3)
CALL ZM_SUB(MZM3,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
CALL ZM_ABS(MZM4,MFM4)
CALL FM_ST2M('0',MFM3)
IF (FM_COMP(MFM4,'GT',MFM3)) CALL PRterr(KW)
```

```
NCASE = 15
QI1 = 123
MZM3 = TO_ZM(QI1,QI2)
CALL ZM_ST2M('123 + 123456789012345678901234567 i',MZM4)
CALL ZM_SUB(MZM3,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
```

```
CALL ZM_ABS(MZM4,MFM4)
CALL FM_ST2M('0',MFM3)
IF (FM_COMP(MFM4,'GT',MFM3)) CALL PRERR(KW)
```

```
END SUBROUTINE TEST1
```

```
SUBROUTINE TEST2
```

! Test the derived type == interface.

```
IMPLICIT NONE
```

```
WRITE (KW, "(/' Testing the derived type == interface.')" )
```

```
NCASE = 16
```

```
QI1 = 123
```

```
M_A = QI1
```

```
IF (.NOT.(M_A == QI1)) THEN
```

```
    CALL ERRPRT_FM(' == ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
```

```
ENDIF
```

```
NCASE = 17
```

```
QI1 = 123
```

```
M_A = QI1
```

```
IF (.NOT.(QI1 == M_A)) THEN
```

```
    CALL ERRPRT_FM(' == ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
```

```
ENDIF
```

```
NCASE = 18
```

```
QI1 = 123
```

```
M_J = QI1
```

```
IF (.NOT.(M_J == QI1)) THEN
```

```
    CALL ERRPRT_IM(' == ',M_J,'M_J',M_J,'M_J')
```

```
ENDIF
```

```
NCASE = 19
```

```
QI1 = 123
```

```
M_J = QI1
```

```
IF (.NOT.(QI1 == M_J)) THEN
```

```
    CALL ERRPRT_IM(' == ',M_J,'M_J',M_J,'M_J')
```

```
ENDIF
```

```
NCASE = 20
```

```
QI1 = 123
```

```
M_Z = QI1
```

```
IF (.NOT.(M_Z == QI1)) THEN
```

```
    CALL ERRPRT_ZM(' == ',M_Z,'M_Z',M_Z,'M_Z',M_Z,'M_Z')
```

```
ENDIF
```

```
NCASE = 21
```

```
QI1 = 123
```

```
M_Z = ( 123.0 , 34.5 )
```

```
IF (M_Z == QI1) THEN
```

```
    CALL ERRPRT_ZM(' == ',M_Z,'M_Z',M_Z,'M_Z',M_Z,'M_Z')
```

```
ENDIF
```

```
NCASE = 22
```

```
QI1 = 123
```

```

M_Z = QI1
IF (.NOT.(QI1 == M_Z)) THEN
    CALL ERRPRT_ZM(' == ',M_Z,'M_Z',M_Z,'M_Z',M_Z,'M_Z')
ENDIF

NCASE = 23
QI1 = 123
M_Z = ( 123.0 , 34.5 )
IF (QI1 == M_Z) THEN
    CALL ERRPRT_ZM(' == ',M_Z,'M_Z',M_Z,'M_Z',M_Z,'M_Z')
ENDIF

RETURN
END SUBROUTINE TEST2

SUBROUTINE TEST3

```

! Test the derived type /= interface.

```

IMPLICIT NONE

WRITE (KW, "(/' Testing the derived type /= interface.')" )

NCASE = 24
QI1 = 123
M_A = 1 + QI1
IF (.NOT.(M_A /= QI1)) THEN
    CALL ERRPRT_FM(' /= ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
ENDIF

NCASE = 25
QI1 = 123
M_A = 1 + QI1
IF (.NOT.(QI1 /= M_A)) THEN
    CALL ERRPRT_FM(' /= ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
ENDIF

NCASE = 26
QI1 = 123
M_J = 1 + QI1
IF (.NOT.(M_J /= QI1)) THEN
    CALL ERRPRT_IM(' /= ',M_J,'M_J',M_J,'M_J')
ENDIF

NCASE = 27
QI1 = 123
M_J = 1 + QI1
IF (.NOT.(QI1 /= M_J)) THEN
    CALL ERRPRT_IM(' /= ',M_J,'M_J',M_J,'M_J')
ENDIF

NCASE = 28
QI1 = 123
M_Z = 1 + QI1
IF (.NOT.(M_Z /= QI1)) THEN
    CALL ERRPRT_ZM(' /= ',M_Z,'M_Z',M_Z,'M_Z',M_Z,'M_Z')
ENDIF

```

```

NCASE = 29
QI1 = 123
M_Z = ( 123.0 , 34.5 )
IF (.NOT.(M_Z /= QI1)) THEN
    CALL ERRPRT_ZM(' /= ',M_Z, 'M_Z',M_Z, 'M_Z',M_Z, 'M_Z')
ENDIF

```

```

NCASE = 30
QI1 = 123
M_Z = 1 + QI1
IF (.NOT.(QI1 /= M_Z)) THEN
    CALL ERRPRT_ZM(' /= ',M_Z, 'M_Z',M_Z, 'M_Z',M_Z, 'M_Z')
ENDIF

```

```

NCASE = 31
QI1 = 123
M_Z = ( 123.0 , 34.5 )
IF (.NOT.(QI1 /= M_Z)) THEN
    CALL ERRPRT_ZM(' /= ',M_Z, 'M_Z',M_Z, 'M_Z',M_Z, 'M_Z')
ENDIF

```

```

RETURN
END SUBROUTINE TEST3

```

```

SUBROUTINE TEST4

```

! Test the derived type > interface.

```

IMPLICIT NONE

```

```

WRITE (KW, "(/' Testing the derived type > interface.')" )

```

```

NCASE = 32
QI1 = 123
M_A = QI1 + 1
IF (.NOT.(M_A > QI1)) THEN
    CALL ERRPRT_FM(' > ',M_A, 'M_A',M_A, 'M_A',M_A, 'M_A')
ENDIF

```

```

NCASE = 33
QI1 = 123
M_A = QI1 - 1
IF (.NOT.(QI1 > M_A)) THEN
    CALL ERRPRT_FM(' > ',M_A, 'M_A',M_A, 'M_A',M_A, 'M_A')
ENDIF

```

```

NCASE = 34
QI1 = 123
M_J = QI1 + 1
IF (.NOT.(M_J > QI1)) THEN
    CALL ERRPRT_IM(' > ',M_J, 'M_J',M_J, 'M_J')
ENDIF

```

```

NCASE = 35
QI1 = 123
M_J = QI1 - 1
IF (.NOT.(QI1 > M_J)) THEN
    CALL ERRPRT_IM(' > ',M_J, 'M_J',M_J, 'M_J')

```

```
ENDIF
```

```
RETURN
```

```
END SUBROUTINE TEST4
```

```
SUBROUTINE TEST5
```

! Test the derived type \geq interface.

```
IMPLICIT NONE
```

```
WRITE (KW, "(/' Testing the derived type  $\geq$  interface.')")
```

```
NCASE = 36
```

```
QI1 = 123
```

```
M_A = QI1 + 1
```

```
IF (.NOT.(M_A  $\geq$  QI1)) THEN
```

```
    CALL ERRPRT_FM('  $\geq$  ', M_A, 'M_A', M_A, 'M_A', M_A, 'M_A')
```

```
ENDIF
```

```
NCASE = 37
```

```
QI1 = 123
```

```
M_A = QI1 - 1
```

```
IF (.NOT.(QI1  $\geq$  M_A)) THEN
```

```
    CALL ERRPRT_FM('  $\geq$  ', M_A, 'M_A', M_A, 'M_A', M_A, 'M_A')
```

```
ENDIF
```

```
NCASE = 38
```

```
QI1 = 123
```

```
M_J = QI1 + 1
```

```
IF (.NOT.(M_J  $\geq$  QI1)) THEN
```

```
    CALL ERRPRT_IM('  $\geq$  ', M_J, 'M_J', M_J, 'M_J')
```

```
ENDIF
```

```
NCASE = 39
```

```
QI1 = 123
```

```
M_J = QI1 - 1
```

```
IF (.NOT.(QI1  $\geq$  M_J)) THEN
```

```
    CALL ERRPRT_IM('  $\geq$  ', M_J, 'M_J', M_J, 'M_J')
```

```
ENDIF
```

```
RETURN
```

```
END SUBROUTINE TEST5
```

```
SUBROUTINE TEST6
```

! Test the derived type $<$ interface.

```
IMPLICIT NONE
```

```
WRITE (KW, "(/' Testing the derived type  $<$  interface.')")
```

```
NCASE = 40
```

```
QI1 = 123
```

```
M_A = QI1 - 2
```

```
IF (.NOT.(M_A  $<$  QI1)) THEN
```

```
    CALL ERRPRT_FM('  $<$  ', M_A, 'M_A', M_A, 'M_A', M_A, 'M_A')
```

```
ENDIF
```

```

NCASE = 41
QI1 = 123
M_A = QI1 + 2
IF (.NOT.(QI1 < M_A)) THEN
    CALL ERRPRT_FM(' < ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
ENDIF

```

```

NCASE = 42
QI1 = 123
M_J = QI1 - 2
IF (.NOT.(M_J < QI1)) THEN
    CALL ERRPRT_IM(' < ',M_J,'M_J',M_J,'M_J')
ENDIF

```

```

NCASE = 43
QI1 = 123
M_J = QI1 + 2
IF (.NOT.(QI1 < M_J)) THEN
    CALL ERRPRT_IM(' < ',M_J,'M_J',M_J,'M_J')
ENDIF

```

```

RETURN
END SUBROUTINE TEST6

```

```

SUBROUTINE TEST7

```

! Test the derived type <= interface.

```

IMPLICIT NONE

```

```

WRITE (KW, "(/' Testing the derived type <= interface.')" )

```

```

NCASE = 44
QI1 = 123
M_A = QI1 - 2
IF (.NOT.(M_A <= QI1)) THEN
    CALL ERRPRT_FM(' <= ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
ENDIF

```

```

NCASE = 45
QI1 = 123
M_A = QI1 + 2
IF (.NOT.(QI1 <= M_A)) THEN
    CALL ERRPRT_FM(' <= ',M_A,'M_A',M_A,'M_A',M_A,'M_A')
ENDIF

```

```

NCASE = 46
QI1 = 123
M_J = QI1 - 2
IF (.NOT.(M_J <= QI1)) THEN
    CALL ERRPRT_IM(' <= ',M_J,'M_J',M_J,'M_J')
ENDIF

```

```

NCASE = 47
QI1 = 123
M_J = QI1 + 2
IF (.NOT.(QI1 <= M_J)) THEN

```



```
CALL ERRPRT_IM(' <= ',M_J,'M_J',M_J,'M_J')
ENDIF
```

```
RETURN
END SUBROUTINE TEST7
```

```
SUBROUTINE TEST8
```

! Test the '+' arithmetic operator.

```
IMPLICIT NONE
```

```
WRITE (KW,("/' Testing the derived type + interface.'"))
```

```
NCASE = 48
MFM3 = QI2 + MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_ADD(MFM4,MFM1,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 49
MFM3 = QI2 + MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_ADD_R1(MFM4,MFM1)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 50
MFM3 = QI2 + MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_ADD_R2(MFM1,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 51
MIM3 = QI2 + MIM1
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_ADD(MIM4,MIM1,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 52
MZM3 = QI2 + MZM1
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_ADD(MZM4,MZM1,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
NCASE = 53
MFM3 = MFM1 + QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_ADD(MFM1,MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 54
MIM3 = MIM1 + QI2
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_ADD(MIM1,MIM4,MIM5)
```

```
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 55
MZM3 = MZM1 + QI2
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_ADD(MZM1,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
END SUBROUTINE TEST8
```

```
SUBROUTINE TEST9
```

```
!           Test the '-' arithmetic operator.
```

```
IMPLICIT NONE
```

```
WRITE (KW,("/' Testing the derived type - interface.'"))
```

```
NCASE = 56
MFM3 = QI2 - MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_SUB(MFM4,MFM1,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 57
MIM3 = QI2 - MIM1
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_SUB(MIM4,MIM1,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 58
MZM3 = QI2 - MZM1
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_SUB(MZM4,MZM1,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
NCASE = 59
MFM3 = MFM1 - QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_SUB(MFM1,MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 60
MIM3 = MIM1 - QI2
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_SUB(MIM1,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 61
MZM3 = MZM1 - QI2
CALL ZM_ST2M('123456789012345678901234567',MZM4)
```

```
CALL ZM_SUB(MZM1,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
END SUBROUTINE TEST9
```

```
END MODULE TEST_A
```

```
MODULE TEST_B
USE TEST_VARS
```

```
CONTAINS
```

```
SUBROUTINE TEST10
```

```
!           Test the '*' arithmetic operator.
```

```
IMPLICIT NONE
```

```
WRITE (KW,("/' Testing the derived type * interface.'"))
```

```
NCASE = 62
MFM3 = QI2 * MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_MPY(MFM4,MFM1,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 63
MFM3 = QI2 * MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_MPY_R1(MFM4,MFM1)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 64
MFM3 = QI2 * MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_MPY_R2(MFM1,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 65
MFM3 = QI2 * MFM1
MFM4 = MFM1 * TO_FM('123456789012345678901234567')
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 66
MIM3 = QI2 * MIM1
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_MPY(MIM4,MIM1,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 67
MZM3 = QI2 * MZM1
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_MPY(MZM4,MZM1,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
NCASE = 68
MFM3 = MFM1 * QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_MPY(MFM1,MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 69
MIM3 = MIM1 * QI2
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_MPY(MIM1,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 70
MZM3 = MZM1 * QI2
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_MPY(MZM1,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
END SUBROUTINE TEST10
```

```
SUBROUTINE TEST11
```

```
!           Test the '/' arithmetic operator.
```

```
IMPLICIT NONE
```

```
WRITE (KW,("/' Testing the derived type / interface.'"))
```

```
NCASE = 71
MFM3 = QI2 / MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_DIV(MFM4,MFM1,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 72
MFM3 = QI2 / MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_EQ(MFM1,MFM6)
CALL FM_DIV_R2(MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 73
MIM3 = QI2 / MIM1
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_DIV(MIM4,MIM1,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 74
MZM3 = QI2 / MZM1
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_DIV(MZM4,MZM1,MZM5)
```

```
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
NCASE = 75
MFM3 = MFM1 / QI2
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_DIV(MFM1,MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 76
MIM3 = MIM1 / QI2
CALL IM_ST2M('123456789012345678901234567',MIM4)
CALL IM_DIV(MIM1,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 77
MZM3 = MZM1 / QI2
CALL ZM_ST2M('123456789012345678901234567',MZM4)
CALL ZM_DIV(MZM1,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
END SUBROUTINE TEST11
```

```
SUBROUTINE TEST12
```

```
!           Test the '**' arithmetic operator.
```

```
IMPLICIT NONE
```

```
WRITE (KW, "(/' Testing the derived type ** interface.')
```

```
NCASE = 78
MFM3 = QI2 ** MFM1
CALL FM_ST2M('123456789012345678901234567',MFM4)
CALL FM_POWER(MFM4,MFM1,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 79
QI4 = 2
MIM3 = QI4 ** MIM1
CALL IM_ST2M('2',MIM4)
CALL IM_POWER(MIM4,MIM1,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 80
QI4 = 23
MZM3 = QI4 ** MZM1
MZM4 = QI4
CALL ZM_POWER(MZM4,MZM1,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
NCASE = 81
```

```
QI4 = 2345
MFM3 = MFM1 ** QI4
MFM4 = QI4
CALL FM_POWER(MFM1,MFM4,MFM6)
CALL FM_EQ(MFM6,MFM4)
IF (.NOT.(MFM3 == MFM4)) CALL PRterr(KW)
```

```
NCASE = 82
QI4 = 17
MIM3 = MIM1 ** QI4
CALL IM_ST2M('17',MIM4)
CALL IM_POWER(MIM1,MIM4,MIM5)
CALL IM_EQ(MIM5,MIM4)
IF (.NOT.(MIM3 == MIM4)) CALL PRterr(KW)
```

```
NCASE = 83
QI4 = 179
MZM3 = MZM1 ** QI4
MZM4 = QI4
CALL ZM_POWER(MZM1,MZM4,MZM5)
CALL ZM_EQ(MZM5,MZM4)
IF (.NOT.(MZM3 == MZM4)) CALL PRterr(KW)
```

```
END SUBROUTINE TEST12
```

```
SUBROUTINE TEST13
```

```
!           Test functions TO_FM, TO_IM, TO_ZM, ..., TO_QUAD_INT.
```

```
IMPLICIT NONE
INTEGER :: J, K
```

```
WRITE (KW, "(/' Testing the derived type TO_FM, ..., TO_QUAD_INT interfaces.')")
```

```
NCASE = 84
QIV = (/ 123, -432, 567 /)
MFMV1 = TO_FM(QIV)
MFMV2 = QIV
DO J = 1, 3
  IF (.NOT.(MFMV1(J) == MFMV2(J))) THEN
    CALL PRterr(KW)
  EXIT
ENDIF
ENDDO
```

```
NCASE = 85
QI3 = 123
MFMV1 = QI3
MFMV2 = 123
DO J = 1, 3
  IF (.NOT.(MFMV1(J) == MFMV2(J))) THEN
    CALL PRterr(KW)
  EXIT
ENDIF
ENDDO
```

```
NCASE = 86
QIV = (/ 123, -432, 567 /)
```

```

MFMV1 = TO_FM(QIV)
QIV = TO_QUAD_INT(MFMV1)
MFMV2 = QIV
DO J = 1, 3
  IF (.NOT.(MFMV1(J) == MFMV2(J))) THEN
    CALL PRTERR(KW)
    EXIT
  ENDIF
ENDDO

NCASE = 87
QIV2 = RESHAPE( (/ (11+3*J, J=1,9) /) , SHAPE = (/ 3,3 /) )
MFMA = TO_FM(QIV2)
MFMB = QIV2
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MFMA(J,K) == MFMB(J,K))) THEN
      CALL PRTERR(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

NCASE = 88
QI3 = 1234
MFMA = QI3
MFMB = 1234
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MFMA(J,K) == MFMB(J,K))) THEN
      CALL PRTERR(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

NCASE = 89
QIV2 = RESHAPE( (/ (11+3*J, J=1,9) /) , SHAPE = (/ 3,3 /) )
MFMA = TO_FM(QIV2)
QIV2 = TO_QUAD_INT(MFMA)
MFMB = QIV2
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MFMA(J,K) == MFMB(J,K))) THEN
      CALL PRTERR(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

NCASE = 90
QIV = (/ 123, -432, 567 /)
MIMV1 = TO_IM(QIV)
MIMV2 = QIV
DO J = 1, 3
  IF (.NOT.(MIMV1(J) == MIMV2(J))) THEN
    CALL PRTERR(KW)
    EXIT
  ENDIF
ENDDO

```

```

ENDIF
ENDDO

NCASE = 91
QI3 = 1234
MIMV1 = QI3
MIMV2 = 1234
DO J = 1, 3
  IF (.NOT.(MIMV1(J) == MIMV2(J))) THEN
    CALL PRterr(KW)
    EXIT
  ENDIF
ENDDO

```

```

NCASE = 92
QIV = (/ 123, -432, 567 /)
MIMV1 = TO_IM(QIV)
QIV = TO_QUAD_INT(MIMV1)
MIMV2 = QIV
DO J = 1, 3
  IF (.NOT.(MIMV1(J) == MIMV2(J))) THEN
    CALL PRterr(KW)
    EXIT
  ENDIF
ENDDO

```

```

NCASE = 93
QIV2 = RESHAPE( (/ (11+3*J, J=1,9) /) , SHAPE = (/ 3,3 /) )
MIMA2 = TO_IM(QIV2)
MIMB2 = QIV2
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MIMA2(J,K) == MIMB2(J,K))) THEN
      CALL PRterr(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

```

```

NCASE = 94
QI3 = 1234
MIMA2 = QI3
MIMB2 = 1234
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MIMA2(J,K) == MIMB2(J,K))) THEN
      CALL PRterr(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

```

```

NCASE = 95
QIV2 = RESHAPE( (/ (11+3*J, J=1,9) /) , SHAPE = (/ 3,3 /) )
MIMA2 = TO_IM(QIV2)
QIV2 = TO_QUAD_INT(MIMA2)
MIMB2 = QIV2
DO J = 1, 3

```



```

DO K = 1, 3
  IF (.NOT.(MIMA2(J,K) == MIMB2(J,K))) THEN
    CALL PRterr(KW)
    EXIT
  ENDIF
ENDDO
ENDDO

```

```

NCASE = 96
QIV = (/ 123, -432, 567 /)
MZMV1 = TO_ZM(QIV)
MZMV2 = QIV
DO J = 1, 3
  IF (ABS(MZMV1(J)-MZMV2(J)) /= 0) THEN
    CALL PRterr(KW)
    EXIT
  ENDIF
ENDDO

```

```

NCASE = 97
QI3 = 1234
MZMV1 = QI3
MZMV2 = 1234
DO J = 1, 3
  IF (ABS(MZMV1(J)-MZMV2(J)) /= 0) THEN
    CALL PRterr(KW)
    EXIT
  ENDIF
ENDDO

```

```

NCASE = 98
QIV = (/ 123, -432, 567 /)
MZMV1 = TO_ZM(QIV)
QIV = TO_QUAD_INT(MZMV1)
MZMV2 = QIV
DO J = 1, 3
  IF (ABS(MZMV1(J)-MZMV2(J)) /= 0) THEN
    CALL PRterr(KW)
    EXIT
  ENDIF
ENDDO

```

```

NCASE = 99
QIV2 = RESHAPE( (/ (11+3*J, J=1,9) /), SHAPE = (/ 3,3 /) )
MZMA2 = TO_ZM(QIV2)
MZMB2 = QIV2
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MZMA2(J,K) == MZMB2(J,K))) THEN
      CALL PRterr(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

```

```

NCASE = 100
QI3 = 1234
MZMA2 = QI3

```

```

MZMB2 = 1234
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MZMA2(J,K) == MZMB2(J,K))) THEN
      CALL PRTERR(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

```

```

NCASE = 101
QIV2 = RESHAPE( (/ (11+3*J, J=1,9) / ), SHAPE = (/ 3,3 /) )
MZMA2 = TO_ZM(QIV2)
QIV2 = TO_QUAD_INT(MZMA2)
MZMB2 = QIV2
DO J = 1, 3
  DO K = 1, 3
    IF (.NOT.(MZMA2(J,K) == MZMB2(J,K))) THEN
      CALL PRTERR(KW)
      EXIT
    ENDIF
  ENDDO
ENDDO

```

```

NCASE = 102
CALL FMM2QI(MFM1%MFM, QI3)
IF (TO_INT(MFM1) /= QI3) CALL PRTERR(KW)

```

```

NCASE = 103
CALL IMM2QI(MIM1%MIM, QI3)
IF (TO_INT(MIM1) /= QI3) CALL PRTERR(KW)

```

```

NCASE = 104
CALL FMM2QI(MZM1%MZM(1), QI3)
IF (TO_INT(MZM1) /= QI3) CALL PRTERR(KW)

```

```

END SUBROUTINE TEST13

```

```

SUBROUTINE TEST14

```

! Test the derived-type interface routines that are not used elsewhere in this program.

```

IMPLICIT NONE

```

```

NCASE = 105
QI4 = MFM1
CALL FMM2QI(MFM1%MFM, QI5)
IF (QI4 /= QI5) CALL PRTERR(KW)

```

```

NCASE = 106
MIM3 = MIM1 / 13
QI5 = MIM1
QI5 = QI5 / 13
IF (.NOT.(MIM3 == QI5)) CALL PRTERR(KW)

```

```

NCASE = 107
QI4 = MOD(MIM1, TO_IM(13))
QI5 = MOD(TO_QUAD_INT(MIM1), 13_QUAD_INT)

```

```
IF (QI4 /= QI5) CALL PRterr(KW)
```

```
NCASE = 108
```

```
QI4 = MIM1
```

```
CALL IMM2QI(MIM1%MIM,QI5)
```

```
IF (QI4 /= QI5) CALL PRterr(KW)
```

```
NCASE = 109
```

```
QI4 = MZM1
```

```
CALL FMM2QI(MZM1%MZM(1),QI5)
```

```
IF (QI4 /= QI5) CALL PRterr(KW)
```

```
END SUBROUTINE TEST14
```

```
END MODULE TEST_B
```

```
MODULE TEST_C
```

```
USE TEST_VARS
```

```
CONTAINS
```

```
SUBROUTINE TEST15
```

! Test type (FM) array equal assignments.

```
IMPLICIT NONE
```

```
INTEGER :: J
```

```
WRITE (KW, "(/' Testing derived-type array operations.')")
```

```
KWSAVE = KW
```

```
CALL FMSETVAR(' KW = 22 ')
```

```
CALL FMSETVAR(' NTRACE = 0 ')
```

```
NCASE = 110
```

```
MFM3 = TO_FM('234.56')
```

```
QIV = MFM3
```

```
QI5 = 0
```

```
DO J = 1, 3
```

```
    QI5 = QI5 + ABS(QIV(J) - 234)
```

```
ENDDO
```

```
CALL FM_ST2M(' 1.0E-45 ',MFM4)
```

```
IF (.NOT.(TO_FM(QI5) <= MFM4)) THEN
```

```
    CALL PRterr(KWSAVE)
```

```
ENDIF
```

```
NCASE = 111
```

```
QIV = (/ 12, -34, 56 /)
```

```
MFMV1 = QIV
```

```
MFM3 = 0
```

```
DO J = 1, 3
```

```
    MFM3 = MFM3 + ABS(MFMV1(J) - QIV(J))
```

```
ENDDO
```

```
CALL FM_ST2M(' 1.0E-45 ',MFM4)
```

```
IF (.NOT.(MFM3 <= MFM4)) THEN
```

```
    CALL PRterr(KWSAVE)
```

```
ENDIF
```

```

NCASE = 112
MFMV1 = (/ TO_FM('12.1123456789') , TO_FM('-34.2123456789') , TO_FM('56.3123456789') /)
QIV = MFMV1
MFM3 = 0
DO J = 1, 3
    MFM3 = MFM3 + ABS(QIV(J) - INT(MFMV1(J)))
ENDDO
CALL FM_ST2M(' 1.0E-45 ',MFM4)
IF (.NOT.(MFM3 <= MFM4)) THEN
    CALL PRterr(KWsave)
ENDIF

KW = KWsave
RETURN
END SUBROUTINE TEST15

SUBROUTINE TEST16

```

! Test type (IM) array equal assignments.

```

IMPLICIT NONE
INTEGER :: J

KWsave = KW
CALL FMSETVAR(' KW = 22 ')

NCASE = 113
MIM1 = TO_FM('234.56')
QIV = MIM1
QI5 = 0
DO J = 1, 3
    QI5 = QI5 + ABS(QIV(J) - 234)
ENDDO
CALL FM_ST2M(' 1.0E-45 ',MFM4)
IF (.NOT.(TO_FM(QI5) <= MFM4)) THEN
    CALL PRterr(KWsave)
ENDIF

NCASE = 114
QIV = (/ 12, -34, 56 /)
MIMV1 = QIV
MFM3 = 0
DO J = 1, 3
    MFM3 = MFM3 + ABS(MIMV1(J) - QIV(J))
ENDDO
CALL FM_ST2M(' 1.0E-45 ',MFM4)
IF (.NOT.(MFM3 <= MFM4)) THEN
    CALL PRterr(KWsave)
ENDIF

NCASE = 115
RV = (/ 12.1, -34.2, 56.3 /)
MIMV1 = RV
MFM3 = 0
QIV = RV
DO J = 1, 3
    MFM3 = MFM3 + ABS(MIMV1(J) - QIV(J))
ENDDO

```