

```

PROGRAM main

  IMPLICIT NONE

  INTEGER, PARAMETER :: n=4,n2=16
  REAL, DIMENSION(n2) :: aa=(/1,2,3,4,5,6,7,8,9,10,&
                             11,12,13,14,15,16/)
  REAL, DIMENSION(n,n) :: mat1,mat2

  mat1=RESHAPE(aa,(/n,n/))
  CALL outmat(mat1)
  PRINT*,"-----"
  mat2=RESHAPE(aa,(/n,n/),ORDER=(/2,1/))
  CALL outmat(mat2)
  PRINT*,"-----"
  mat1=transmat(mat2)
  CALL outmat(mat1)

CONTAINS

! *****

SUBROUTINE outmat(mat)
!**** Subroutine to output a matrix to the screen.

  REAL, DIMENSION(:,:), INTENT(IN) :: mat !** Dummy declaration

  INTEGER :: i

  DO i=1,SIZE(mat,1)
    PRINT*,mat(i,:)
  ENDDO

END SUBROUTINE outmat

! *****

FUNCTION transmat(mat)

  !**** Dummy arguments
  REAL, DIMENSION(:,:), INTENT(IN) :: mat
  INTEGER :: m,n,i,j
  REAL, DIMENSION(SIZE(mat,2),SIZE(mat,1)) :: transmat !** Return

  !**** Find out the no. of rows and cols in the matrix

  m=SIZE(mat,1) ; n=SIZE(mat,2)

  !**** Perform the transpose
  !**** using two DO loops

  DO i=1,n
    !** Loop over rows in mat
    DO j=1,m
      !** Loop over cols in mat
      transmat(i,j)=mat(j,i)
    ENDDO
  ENDDO

! *****

END FUNCTION transmat

END PROGRAM main

*****
*****

PROGRAM classproj1
!*** Program to multiply together two matrices mat1 & mat2 and store
!*** the result in mat3. USES dynamic allocation of memory!!!

```

```

  IMPLICIT NONE

  INTEGER :: m,n,k
  REAL, ALLOCATABLE, DIMENSION(:,:) :: mat1
  REAL, ALLOCATABLE, DIMENSION(:,:) :: mat2
  REAL, ALLOCATABLE, DIMENSION(:,:) :: mat3,mat4

  !**** Input the two matrices from the keyboard
  !**** Press return after each element
  !**** matrix one then matrix two
  !**** NOTE you could read each matrix is separately if you prefer!

  PRINT*,"Please input the no. of rows in matrix one"
  READ*,m
  PRINT*,"Please input the no. of cols in matrix one"
  READ*,n
  PRINT*,"Please input the no. of cols in matrix two"
  READ*,k

  ALLOCATE(mat1(m,n))
  ALLOCATE(mat2(n,k))
  ALLOCATE(mat3(m,k))
  ALLOCATE(mat4(m,k))

  !**** Input the two matrices from the keyboard
  !**** Press return after each row
  !**** matrix one then matrix two

  mat1=getmat(m,n)
  mat2=getmat(n,k)

  !**** Use our matrix mult. function to calc. mat3
  mat3=mulmat(mat1,mat2)

  !**** Perform the matrix multiplication
  !**** using Fortran's MATMUL(A,B) function

  mat4=MATMUL(mat1,mat2)

  !**** Print out each matrix to the screen
  !**** NOTE the use of / to create a blank line

  PRINT '(/,"Matrix One",/)'
  CALL outmat(mat1)

  PRINT '(/,"Matrix Two",/)'
  CALL outmat(mat2)

  PRINT '(/,"Matrix1 * Matrx2 (My Answer)",/)'
  CALL outmat(mat3)

  PRINT '(/,"Matrix1 * Matrx2 (Fortran Answer)",/)'
  CALL outmat(mat4)

  DEALLOCATE(mat4)
  DEALLOCATE(mat3)
  DEALLOCATE(mat2)
  DEALLOCATE(mat1)

CONTAINS

! *****

FUNCTION getmat(m,n)
!** Read in matrix with "m" rows and "n" columns

  INTEGER, INTENT(IN) :: m,n !**** Dummy declaration

```

```

REAL, DIMENSION(m,n) :: getmat  !**** Local Declaration

INTEGER :: i  ! ** Loop variable

DO i=1,m
  PRINT '("Enter matrix row :",i2)',i !** Prompt for row number
  READ*,getmat(i,:) !** Read in row
ENDDO

END FUNCTION getmat

! *****

SUBROUTINE outmat(mat)
!** Print any matrix out to screen

REAL, DIMENSION(:,:), INTENT(IN) :: mat !** Dummy declaration

INTEGER :: i  ! ** Loop variable

DO i=1,SIZE(mat,1) !** Loop for each row.
  PRINT*,mat(i,:) !** Print each row.
ENDDO

END SUBROUTINE outmat

! *****

FUNCTION mulmat(mat1,mat2)
!**** Function to premultiply mat2 with mat1

!**** Use assumed shape arrays for dummy arrays****
REAL, DIMENSION(:,:), INTENT(IN) :: mat1 !** Dummy declaration
REAL, DIMENSION(:,:), INTENT(IN) :: mat2 !** Dummy declaration

!***** Local Declarations *****
INTEGER :: m,n,k,i,j,p
REAL, DIMENSION(SIZE(mat1,1),SIZE(mat2,2)) :: mulmat

!***** Find out the matrix sizes for the DO loop limits *****

m=SIZE(mat1,1) ; n=SIZE(mat1,2) ; k=SIZE(mat2,2)

!**** Perform the matrix multiplication
!**** using three DO loops

IF (SIZE(mat2,1) == n) THEN
  DO i=1,m !** For each row of getmat (mat3)
    DO j=1,k !** For each column of getmat (mat3)
      mulmat(i,j)=0 !** initialise to zero
      DO p=1,n
        mulmat(i,j)=mulmat(i,j)+mat1(i,p)*mat2(p,j)
      ENDDO
    ENDDO
  ENDDO
ELSE
  PRINT*,"Size mismatch in mulmat"
ENDIF

END FUNCTION mulmat

! *****

END PROGRAM classproj1

*****
*****

```