City University
School of Engineering and Mathematical Sciences
Common part 1

COMPUTING

Fortran Lecture 2- FORMAT and Arithmetic Statements

1. COMMENTS

When we write a computer program it is normal to include some *comments*, written in English, to explain how the program works. Professional programmers make their program source files *self documenting.* In Fortran 95 an exclamation mark "!" is used as the first character in a comment line. This means that the compiler will ignore it. Older versions of Fortran (Fortran IV and Fortran 77) used a "C" for comments.

2.1 Format Statements

The last lecture explained the difference between REAL and INTEGER variables and constants. We also need CHARACTER and COMPLEX variables and characters. Each type of variable is uses a different binary code. When we need to input data to the computer, or output answers, the binary code must be converted to printable characters. The exact conversion method is specified by the FORMAT statement.

```
!
!============
!Program PROG6
! This demonstrates formats for REAL numbers
!============
!
            Program PROG6
            WRITE (*,1000)              ! tell the user what to do
            READ (*,1001) X,Y,Z         ! get three numbers into memory
            F=X*Y*Z                     ! arithmetic
            WRITE (*,1002)  X,Y,Z,F     ! write out answers
             STOP                       ! all done now
       1000 FORMAT (1X,'Please enter three real numbers   ')
       1001 FORMAT (3F12.6)             ! three real numbers
       1002 FORMAT (1X, F12.6, '   multiplied by  ',F12.6, &
              ' and ',F12.6,   '  equals  ' ,F12.6)
             END PROG6
```

Program PROG6 is old Fortran, but still works with Fortran 95. Fortran 95 can also use PRINT and the FORMAT is easier. The WRITE (*,1000) statement

means write to device * - i.e. the VDU terminal we are logged into, with FORMAT statement number 1000. The 1X means “start a new line” and the quotation defines a CHARACTER STRING to write out. READ (*,1001) X,Y,Z means input three real numbers from the keyboard using a conversion specified in statement number 1001 , and store them in variables called X,Y and Z. In statement 1001 the F is a code for REAL number conversion F12.6 means a real number with 6 digits after the decimal place and a maximum of 12 characters, including the decimal point. 3F12.6 means repeat the F12.6 three times.

WRITE (*,1002) X,Y,Z,F Means write out , to the VDU screen, the four real variables X,Y,Z,F and use a conversion defined by FORMAT statement number 1002. FORMAT 1002 must contain conversions for four real numbers.

2.2 Input errors

Program PROG6 expects three REAL numbers. What happens if you enter integers or any CHARACTER constants? Much more of this latter.

2.3 REAL format conversions

Beside the F code we can also use G, ES and EN conversions for REAL numbers. Change PROG6 to read

```
1002 FORMAT (1X,ES12.6,’ times ‘,EN12.6, ‘ times ‘,G12.6, &
        ‘ equals ‘, EN12.6 )
```

2.4 Ampersand

Ask your tutor what the “&” character in the FORMAT statement does.

2.5 Integer and spaces format

The I code is use for integers. Format specifications of I8 will print out an integer number with a maximum of 8 digits, including any plus or minus sign. 5I8 means print out 5 integers with 8 digits each. Blank spaces can be specified with an X format. 3(I5,3X) means print out an integer number with 5 digits, leave 3 spaces then repeat twice more. Try to modify PROG6 with:-

```
1002 FORMAT (1X,ES12.6,’ times ‘,10X,EN12.6, ‘ times ‘,10X,G12.6, &
        ‘ equals ‘, 3X,EN12.6 )
```

2.6 fortran 95 free format

```
!-==================
! Fortran 95 free format is much easier.
! the compiler will allocate format conversions automaticaly depending on
!the variable type name
!==================
! this progam  computes the square &
! cube of a number
     Program PROG7
     real num
!------
! declare variable num as real, WRITE (*,*) means write to
! VDU screen with free format.
!------
      write(*,*)'type in a number and hit return'
      read(*,*)num   ! have defined num as real,
                     !so compiler will
                     !automatically use real format
      square = num*num
      cube = num*num*num
      write(*,*)square  ! square must be real, so F format
      write(*,*)cube    ! cube real, so F format used
      stop
      end PROG7
```

3.Arithmetic operations

All computers can only perform arithmetic, not algebra. This means that all variables in the program must have a numerical value before we can use them. Fortran arithmetic operators are exponent, i.e. raising to a power, multiply and divide, addition and subtraction. Brackets are also used to define which order the operations are performed. Be very careful with mixed integer and real variable arithmetic!

Arithmetic operations

Operation	**Fortran**
Addition	a + b
Subtraction	a – b
Multiplication	a*b
Division	a/b
Exponentiation (a^b)	a**b

Priority of operations

Priority	Operation
1	Parentheses
2	Exponentiation
3	Multiplication & division
4	Addition & subtraction

Nested parentheses are evaluated from the inner most pair. When two operations are of the same priority, the expression is performed from left to right. The exception to this rule is exponentiation, which is evaluated from right to left. E.g., the expression a**b**c is equivalent to a**(b**c).

4. Functions

A function is a small program usually stored in a library which we can use. It is compiled separately from the main program and comes into our .exe file when we build the program. A Function will calculate one specific answer using data which we supply. The data we give the function is called the *arguments of the function.*

Some intrinsic functions

Function	Function Value	Remarks
`sqrt(x)`	Square rot of x	
`abs(x)`	Absolute value of x	
`sin(x)`	Sine of angle x	X must be in radians
`cos(x)`	Cosine of angle x	X must be in radians
`tan(x)`	Tangent of angle x	x must be in radians
`exp(x)`	e raised to the power of x	
`log(x)`	Natural log of x	
`log10(x)`	Common log of x	
`int(x)`	Converts a real value to an integer value	Truncates x
`real(i)`	Converts an integer value to a real value	

Intrinsic means the functions are already available in the *.dll library. To use a function just write

Variable = function (arguments)

e.g. to find the square root of a number, write

```
x=2.0                          !real number
rootx= sqrt(x)
```

Note the trigonometric functions expect the angles in *radians* not *degrees.*
be careful with real and integer numbers. We can use two functions i=int(x) and y=real(m) to convert data types.