FORTRAN Subroutines and Functions Engineering Applications

Dr. Ugur Guven

Aerospace & Nuclear Engineer

Functions in FORTRAN

- In engineering applications, one of the most important things is to calculate a function in order to solve an engineering problems.
- Some functions may be used more then once in engineering calculations.
- Hence constantly using IF-THEN or GOTO or Loops would be counterproductive and as a result the FUNCTION Command will be useful.

Function Statement

- A function begins with a FUNCTION statement and ends with the next END statement. A function can contain any statements except BLOCK DATA, ENTRY, FUNCTION, PROGRAM, or SUBROUTINE.
- The Function statement can contain a set of operations or a single function depending upon your need.

Function Example

```
function func(i) result(j)
      integer i                          ! input
      integer j                          ! output
   j = i**2 + i**3
 end function func
```

Function Example

- Define the function in the beginning of the program
- Define the input variable in the function in the beginning of the program as well as in the beginning of the function itself
- Define the output variable in the beginning of the function routine as integer, real etc.
- Use the word result in your function statement to overcome confusion

Fortran Program with Function Statement

```
program xfunc
   integer i
   integer func
              print*, "Input the number"
              read*, i

   print*,"sum of the square and cube of",i," is",func(i)

               end program xfunc

function func(i) result(j)

   integer i  ! input
   integer j   !output

    j = i**2 + i**3
end function func
```

Function Statement Remarks

- Function Statement is fine when you only have a single function that will be used repeatedly in a setting
- There should be a single result of a function
- You can call a function as many times as you wish, but you should be very careful with handling of the input and the output variables
- Redundantly define your variables in function each time, but make sure that they are the same in the main program as well.

Subroutines in FORTRAN

- You will want to use a function if you need to do a complicated calculation that has only one result which you may or may not want to subsequently use in an expression. However, that is the biggest advantage of a function as you can use it directly in a FORTRAN expression
- Subroutines are used to perform several tasks at once as many times as you want in the program.
- However, calls to subroutines cannot be placed in an expression.

Subroutine Syntax in FORTRAN

```
SUBROUTINE subroutine-name (arg1, arg2, ..., argn
   IMPLICIT NONE
   [specification part]
   [execution part]
   [subprogram part]
END SUBROUTINE subroutine-name
```

Call Statement in FORTRAN

- In the main program, a subroutine is activated by using a CALL statement which include the subroutine name followed by the list of inputs to and outputs from the subroutine surrounded by parenthesis.
- The inputs and outputs are collectively called the arguments.
- Subroutine names should be different than those used for variables or functions

Subroutine Format

- They begin with a line that includes the word SUBROUTINE, the name of the subroutine, and the arguments for the subroutine.
- The subroutine name is not declared anywhere in the program.
- All variables used by the subroutine, including the arguments, must be declared in the subroutine
- A subroutine is finished off with a RETURN and an END statement.

Program Example with Subroutines

```
      PROGRAM SUBDEM
      REAL A,B,C,SUM,SUMSQ
      CALL INPUT(  A,B,C)
      CALL CALC(A,B,C,SUM,SUMSQ)
      CALL OUTPUT(SUM,SUMSQ)
      END

      SUBROUTINE INPUT(X, Y, Z)
      REAL X,Y,Z
      PRINT *,'ENTER THREE NUMBERS => '
      READ *,X,Y,Z
      RETURN
      END

      SUBROUTINE CALC(A,B,C, SUM,SUMSQ)
      REAL A,B,C,SUM,SUMSQ
      SUM = A + B + C
      SUMSQ = SUM **2
      RETURN
      END

      SUBROUTINE OUTPUT(SUM,SUMSQ)
      REAL SUM, SUMSQ
      PRINT *,'The sum of the numbers you entered are: ',SUM
      PRINT *,'And the square of the sum is:',SUMSQ
      RETURN
      END
```

Subroutine Semantics

- The meaning of a subroutine is very simple: A subroutine is a self-contained unit that receives some "input" from the outside world via its formal arguments, does some computations, and then returns the results, if any, with its formal arguments.
- Unlike functions, the name of a subroutine is **not** a special name to which you can save a result. Subroutine's name is simply a name for identification purpose and you cannot use it in any statement except the **CALL** statement.
- A subroutine receives its input values from its formal arguments, does computations, and saves the results in some of its formal arguments. When the control of execution reaches **END SUBROUTINE**, the values stored in some formal arguments are passed back to their corresponding actual arguments.
- Any statements that can be used in a **PROGRAM** can also be used in a **SUBROUTINE**.

THANK YOU

www.itlectures.co.cc

drguven@live.com