FORTRAN 90: Selective Execution

Meteorology 227
Logical Expressions

• Can be simple or compound.

• (expression) (relational-operator) (expression)

• Relational Operators
  – < or .LT. Is less than
  – > or .GT. Is greater than
  – == or .EQ. Is equal to
  – <= or .LE. Is less than or equal to
  – >= or .GE. Is greater than or equal to
  – /= or .NE. Is not equal to

• Note:
  – == is a relational operator
  – = is an assignment statement

• Logical Expressions evaluate to .TRUE. Or .FALSE.
Examples

- \( B^{**2} \geq 4.0 \times A \times C \)
- \( 1.0 \leq 24.0 \)
- “A” < “F”
- “cat” < “dog”
- “cat” < “cow”
- “June” < “July”
- “cat” < “cattle”
Compound Logical Expressions

• Compound operators
  – .NOT. (negation)
  – .AND. (conjunction)
  – .OR. (disjunction)
  – .EQV. (equivalence)
  – .NEQV. (Non-equivalence)

• Order of operations
  1. Arithmetic operations
  2. Relational operators
  3. Logical operation in the order: .NOT., .AND., .OR., .EQV. (.NEQV.)

• Use Parentheses to remove any ambiguity.
Examples

• Let \( N=4 \)

• \( N^2 + 1 > 10 \) .AND. .NOT. \( N < 3 \)

• \( (N^2 + 1 > 10) \) .AND. .NOT. \( (N < 3) \)

• Data_Name == “RAIN_NON” .OR. “RAIN_CON”
IF constructs

• Block IF construct
  – IF (logical-expression) THEN
    statement-sequence
    ENDIF
  – IF (Precip_Rate >= 0.50) THEN
    Accumulation = Precip_Rate * Time_Interval
    Print *, ‘It is raining cats and dogs!’
    ENDIF

• Logical IF statement
  – IF (logical-expression) statement
  – IF (Precip_Rate >= 0.50) Print *, ‘It is raining cats and dogs!’
General form

• Flowchart
• IF (logical-expression) THEN
  statement-sequence 1
ELSE
  statement-sequence 2
ENDIF
• IF ((Precip_Rate > 0.0).AND.(Temp < 32.0)) THEN
  Print *, 'It is Snowing'
ELSE
  Print *, 'It is Raining'
ENDIF
IF-ELSE IF constructs

- How would you do it now?
  - IF (logical-expression) THEN
    statement-sequence
  ELSE
    IF (logical-expression) THEN
      statement-sequence
    ELSE
      statement-sequence
    ENDIF
  ENDIF

- This method is somewhat tedious. FORTRAN helps you out a bit.
IF-ELSE IF cont.

- IF (logical-expression) THEN
  statement-sequence
ELSE IF (logical-expression) THEN
  statement-sequence
ELSE IF (logical-expression) THEN
  statement-sequence
ELSE
  statement-sequence
ENDIF
CASE construct

• Less general form of IF-ELSE IF construct, but still very useful.

• General Form
  – SELECT CASE (Selector)
    CASE (label-list-1)
      Statement-sequence 1
    CASE (label-list-2)
      Statement-sequence 2
      …..
    CASE (label-list-n)
      Statement-sequence n
  END SELECT
CASE cont.

• Selector: Integer, character, or logical expression

• Label-list: List of one or more possible values of the selector, enclosed in parentheses, or the word default.
  – Value
  – Value-1 : Value-2
  – Value-1 :
  – : Value-2

• If the value is not any of the lists of values, the sequence of statements associated with DEFAULT is executed.
Example 1 – Class Code

- SELECT CASE (ClassCode)
  CASE (1)
    Print *, ‘Freshman’
  CASE (2)
    Print *, ‘Sophomore’
  CASE (3)
    Print *, ‘Junior’
  CASE (4)
    Print *, ‘Senior’
  CASE (5)
    Print *, ‘Graduate’
  CASE DEFAULT
    Print *, ‘Illegal class code:’, ClassCode
END SELECT
Example 2 – Enhanced Fujita Scale

- SELECT CASE (Wind_Speed)
  CASE (:85)
    Print *, ‘EF-0’
  CASE (86:110)
    Print *, ‘EF-1’
  CASE (111:135)
    Print *, ‘EF-2’
  CASE (136:165)
    Print *, ‘EF-3’
  CASE (166:200)
    Print *, ‘EF-4’
  CASE (200:318)
    Print *, ‘EF-5’
  CASE DEFAULT
    Print *, ‘Finger of God’
END SELECT
LOGICAL Data Type

- Two logical constants in FORTRAN
  - .TRUE. And .FALSE.

- Logical variable
  - LOGICAL :: list
  - LOGICAL :: End_of_Data, RootExists

- Assignment Statement
  - RootExists = Discriminant >=0

- List output of a logical variable
  - Let A=True, B and C = False
  - Print *, A,B,C,.True,.False.
  - T_F_F_T_F

- Program example