Fortran 77: 6. Conditional Branches

Logical statements that are used control the flow of code; if a condition is true then do something, if it isn't then do something else.

IF statement

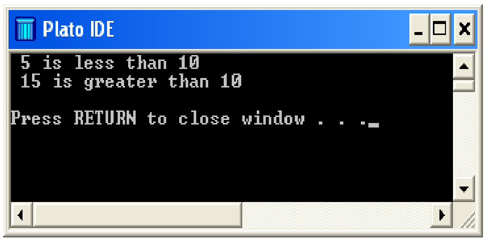
The IF statement in Fortran 77 has the following syntax:

IF (<logical-expression>) <statement>

If the <logical-expression> evaluates to .TRUE. then the <statement> is executed and control passes to the next statement in the program. If the <logical-expression> evaluates to .FALSE. then the <statement> is ignored and control passes to the next statement in the program.

The following program demonstrates how the IF statement works.

```
C **********************
C Conditional Branch - IF
C **********************
    PROGRAM IF1
    INTEGER X
    X=5
    IF (X.GT.10) WRITE(*,*) '5 is greater than 10'
    IF (X.LT.10) WRITE(*,*) '5 is less than 10'
    X=15
    IF (X.GT.10) WRITE(*,*) '15 is greater than 10'
    IF (X.LT.10) WRITE(*,*) '15 is less than 10'
    STOP
    END
```



```
 5 is less than 10
 15 is greater than 10

Press RETURN to close window . . .
```

Block IF

The block IF statement allows us to bundle a number of statements for execution conditional on the value of a logical expression. The block IF statement has the following syntax.

```
IF (<logical-expression>) THEN
        <statement1>
        <statement2>
       …
        <statementN>
END IF
```

The list of statements <statement1> <statement2> ... <statementN> may be written as follows.

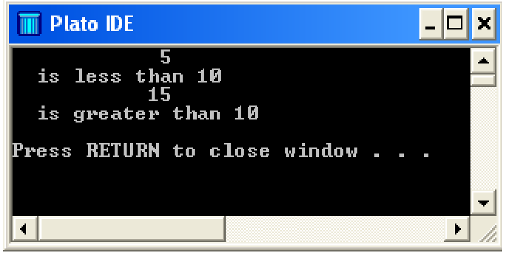
```
IF (<logical-expression>) THEN

        <statement-block>

END IF
```

An example of using the block-IF statement is given in the following program

```
C **********************
C Block IF
C **********************
   PROGRAM BLOCKIF
   INTEGER X
   X=5
   IF (X.GT.10) THEN
     WRITE(*,*) X
     WRITE(*,*)' is greater than 10'
   ENDIF
   IF (X.LT.10) THEN
     WRITE(*,*) X
     WRITE(*,*)' is less than 10'
   ENDIF
   X=15
   IF (X.GT.10) THEN
     WRITE(*,*) X
     WRITE(*,*)' is greater than 10'
   ENDIF
   IF (X.LT.10) THEN
     WRITE(*,*) X
     WRITE(*,*)' is less than 10'
   ENDIF
   STOP
   END
```



```
          5
 is less than 10
         15
 is greater than 10

Press RETURN to close window . . .
```

IF-THEN-ELSE-ENDIF

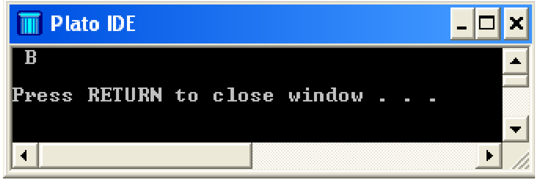
By using the IT-THEN-ELSEIF-ENDIF construct, two or more branches in the code can be designed. The statement has the following syntax:

```
IF (<logical-expression1>) THEN
        <statement-block1>
ELSE IF (<logical-expression2>) THEN
        <statement-block2>
…
ELSE
        <statement-blockN>
END IF
```

Once a logical expression is satisfied, the associated statement block is executed and the IF-THEN-ELSEIF-ENDIF construct is exited. If none of the logical expressions are satisfied then the statement following ELSE is executed.

In the following code the IF-THEN-ELSEIF-ENDIF construct is demonstrated; a grade of A, B, C or F is given depending on the mark.

```
C *********************
C IFELSE
C *********************
   PROGRAM IFELSE
   INTEGER MARK
   MARK=60
   IF (MARK.GE.70) THEN
    WRITE(*,*) 'A'
   ELSE IF (MARK.GE.55) THEN
    WRITE(*,*) 'B'
   ELSE IF (MARK.GE.40) THEN
    WRITE(*,*) 'C'
   ELSE
    WRITE(*,*) 'F'
   ENDIF
   STOP
   END
```



```
 B
Press RETURN to close window . . .
```

Good Practice Tip

In the example programs, you will note that the statements attached to an IF statement have been indented. Indentation aids clarity and is considered as good practice.