IBM

FORTRAN for the IBM 1130

Chapter 2

Programmed Instruction Course

IBM

FORTRAN for the IBM 1130

Chapter 2

Programmed Instruction Course

FOREWORD

In the preceding chapter you were shown how to refer to quantities - constants, variables, or lists - with FORTRAN notation and how to involve these quantities in useful mathematical computation. Most programs require much more than this, however. Very few problems suitable for computer treatment can be solved by a simple sequence of formulae. In most cases the program must also involve decision making, such as deciding whether a number is positive, zero, or negative. Instead of writing the program in a "straight line" the programmer builds in "loops" and "branches". For these reasons FORTRAN provides a set of "Control Statements" to give the programmer the tools with which to provide these options. This chapter will cover in detail the essential Control Statements provided by FORTRAN.

Copies of this publication can be obtained through IBM Branch Offices.
Address comments concerning the contents of this publication to:
IBM DPD Education Development, Education Center, Endicott, New York.

1 The normal order of execution of statements by the computer is the order in which they are written. The statements

```
1     Y = A+3.*B
2     X = X+Y
```

would tell the computer to execute statement No. 1 completely, then go on to execute statement No. 2.

Q. In the example shown above, the variable named Y has its value defined in statement No. _____.

●●●

A. 1

2 This chapter will be devoted to the use of Control statements with which the programmer can alter the normal order of execution of statements.

Q. (True or False) Except where Control statements are used, the order in which statements are executed by the computer is the order in which they are written.

●●●

A. True

3 A Control statement is, by definition, a statement which may cause the computer to execute a statement other than the next statement in sequence.

Q. A Control statement changes the _____ in which statements are executed.

●●●

A. order (or sequence)

4 You have observed that writing Arithmetic statements permits considerable flexibility in assigning names and constructing expressions, etc. as long as you remain within the basic format. Most Control statements, however, have rigid rules of wording, punctuation, and format which must be observed.

5 In order to tell the computer to execute an instruction which is not the next in the written sequence there must be a way in which to refer to a specific statement in the program. This is accomplished by statement numbers.

Q. FORTRAN statements may have _____ by which they may be referred to by other statements.

●●●

A. statement numbers

6 In FORTRAN programs any statement may be assigned a number. This number is arbitrarily chosen by the program writer and is placed to the left of the actual statement, as, for example:

```
100  Y = A+3.*B
1    X = X+Y
```

Q. The second statement shown above has a statement number of _____.

●●●

A. 1

7 The rules for numbering statements are simple: any statement may have an assigned number, no particular order of numbers is required, and, naturally, no two statements may have the same number.

Q. Statement numbers are arbitrarily assigned numbers appearing on the _____ of the statement to which they refer.

●●●

A. left

8 Given the statements

```
    1    X = 2.1059
    3    A = 3.
 3000    B = 4.
    2    Y = A*X**2+B*X
```

the computer will proceed to execute them in the order written: 1, 3, 3000, and then 2.

Q. (True or False) The numerical value of a statement number has no bearing on the order of execution.

●●●

A. True

9 Although any statement may have an arbitrarily assigned statement number, it is used in most cases only where a "label" is needed; that is, a statement number is used where it is necessary to refer to that statement from some other part of the program.

Q. (True or False) Arithmetic statements are the only statements which can have statement numbers.

●●●

A. False

10 Statement numbers are chosen in an arbitrary fashion, but they must not be larger in size than a certain upper limit. On the 1130, for example, statement numbers may not exceed 5 digits.

Q. The maximum statement number in 1130 FORTRAN would be _____.

●●●

A. 99999

11 The principle use of statement numbers is to provide a reference for Control statements. The Control statement tells the computer which statement to execute next by referring to the statement number that has been assigned to that statement.

Q. Statement numbers are referred to chiefly by _____ statements.

●●●

A. Control

12 The Control statement enables the computer to make controlled "decisions". These decisions are always a choice of the next statement to be executed based on some elementary condition, such as whether a number is positive, negative, or zero. This chapter will show how to use these decisions to control a program.

Q. (True or False) The "decision" the computer has to make is "which statement do I execute next"?

●●●

A. True

13 Since the computer can only decide where to go next in the program, it is up to the programmer to arrange the parts of the program in a logical order to take advantage of this control; that is, the statements must be grouped and ordered according to the effect you wish the decision to have.

14 One of the most useful decision-making statements is the "IF" statement. This statement always consists of the word "IF" followed by a pair of parentheses containing any desired expression, after which come three statement numbers. An example of an IF statement might be IF (X-Y) 10, 20, 30.

Q. The IF statement always contains exactly _____ statement numbers.

●●●

A. three

15 The IF statement tells the computer to do the following: "compute the entire expression contained in the parentheses; if the computed value is negative go next to the first indicated statement, if the value is zero go next to the second indicated statement, if the computed value is positive go next to the third indicated statement."

Q. In the example IF(X-Y)10,20,30 assume the computed value of (X-Y) is positive. When this IF statement is executed under those conditions, the next statement to be executed would be number _____.

●●●

A. 30

16 The computer's ability to select one of three possible successive statements as in the IF statement is often called "branching" or "conditional transfer." When the IF statement is executed the computer will "branch" on the condition of a negative, zero, or positive value for the indicated expression.

Q. In the statement IF(X**2-1.0)30,20,10 if the value of X were -1.0 the computer would go next to statement number _____ when executing this statement.

●●●

A. 20 (the expression (X**2-1.0) is zero for that value)

17 The IF statement must conform exactly to the prescribed format, so let's look at that again: the statement consists of the word "IF" followed by a pair of parentheses containing an expression (which may have additional parentheses), followed in turn by exactly three statement numbers.

Q. (True or False) The statement IF(X**3)17,2,39,103 is a legal IF statement.

●●●

A. False (too many options)

18 Notice that the statement numbers are separated by commas. Naturally there must be some way of clearly defining where one statement number ends and the next begins. However, there is no comma permitted either before the first statement number or after the last.

Q. The statement IF(X*Y-25.)567 can be made legitimate by inserting two _____.

●●●

A. commas

19 It is possible to make a two-way branch with an IF statement by making two of the statement number options the same (three statement numbers in all are still required). For instance, the statement IF(A)7,7,8 will send the computer to statement number 8 only if the value of A is positive. (Not negative and not zero)

Q. Regardless of the values of A, B, and C, the computer when executing the statement IF(A+B-C)20,20,20 will always go next to statement number _____.

●●●

A. 20 (clearly this is nonsense; no useful decision would be made)

20 The nonsense question of the preceding frame is intended to make a point: the IF statement will branch to one of three alternative paths (if the indicated statement numbers are all different) or one of two alternative paths (if any two of the three numbers are alike), but it serves no purpose if all the numbers are alike.

Q. If you wished to have the computer execute statement No. 100 if VALUE were a positive quantity or execute statement No. 200 otherwise, you would write the following statement: _____.

●●●

A. IF(VALUE)200,200,100

21 An IF statement, like any FORTRAN statement, may have a statement number of its own for reference by other statements. An IF statement is not allowed to branch to itself, however, because such a condition could obviously be disastrous. Think what would happen in the statement 3 IF(A) 1,2,3 if A were positive!

Q. If the variable A in the statement sample shown above were positive, the computer would go on executing statement number _____ until someone intervened.

●●●

A. 3

22 In an IF statement, the expression whose value is to be tested may be as simple or complex as desired. It may contain parentheses of its own but it must be completely contained in a pair of parentheses belonging to the IF statement itself.

Q. (True or False) The statement IF ((A+B)*C)2,2,3 is a legal IF statement.

●●●

A. True

23 Q. Write a statement to test the expression A(J)**3-TEST. If the computed value is negative, the computer should execute statement No. 39 next; otherwise the computer should go to statement No. 1. _____.

●●●

A. IF(A(J)**3-TEST)39,1,1

24 If your answer agrees with the one shown above, go to frame 29. If you did not get the correct answer, go to the next frame.

25 This problem was obviously intended for an IF statement: write down the word "IF" and an open-parenthesis; follow this with the tested expression and the balancing close-parenthesis. Next comes the group of statement numbers which indicate the branch directions.

26 If the expression's value was negative in this problem the computer was to go next to statement number 39; otherwise, statement number 1 was to be executed next. The order of statement number options always corresponds to negative, zero, or positive expression values, respectively. Thus 39,1,1 is the proper sequence of statement numbers for this problem.

27 The complete statement to fulfill the requirements of this sample problem is: IF(A(J)**3-TEST)39,1,1

Q. Write a statement such that the computer will execute No. 27 next if A*X**2-C**(J-1) has a value of zero or No. 28 otherwise. _____.

●●●

A. IF(A*X**2-C**(J-1))28,27,28

28 If your answer agrees with the one shown above (including at least four parentheses) go on to frame 29.

If you answer is incorrect, better review all the material up to this point on IF statements, starting with frame 14. If you have already reviewed the material, see your advisor.

29 One very common application of the IF statement is to determine if a computed quantity is less than, equal to, or greater than another quantity. This is done by writing the expression as a difference of the two quantities so that the resulting value will be negative, zero, or positive, respectively.

Q. In the statement IF(X-3.0)20,30,40 the computer is told to go next to statement No. _____ if the value of X is greater than 3.0.

●●●

A. 40 (since (X-3.0)> 0.)

30 The example of the preceding frame showed a variable compared with a constant. The same method can be applied to two variables as in the example IF(X-Y)1,2,5

Q. The computer, when executing the statement above, will go next to statement number 1 if the value of X is _____ than the value of Y.

●●●

A. less (since result is negative)

31 In review: the IF statement causes the computer to compute an expression, just as it does for an Arithmetic statement, and selects the part of the program to which it will go next depending on whether the value of the computation is negative, zero, or positive.

32 Perform Exercise 2.1 in your problem book.

One important point: the value of the computed expression is lost as soon as the computer has completed the IF statement. The expression in the IF statement is simply a series of operations resulting in a single computed value and does not imply that a variable will contain this value as in the Arithmetic statement.

Q. (True or False) After executing the IF statement, the value of the expression computed therein is then available for future statements.

●●●

A. False

33 To illustrate this point consider the statement IF(A*B-C)1,1,2 which will cause the computer to compute the expression A*B-C and pick out its next statement according to the usual rule. The value of A*B-C is no longer available to future statements unless it is recomputed by an Arithmetic statement, resulting in duplication of effort and wasted time.

Q. (True or False) After executing the statement IF(X-Y)10,20,30 the computed value of X-Y is not available and must be recomputed if needed in future statements.

●●●

A. True

34 The programmer should give a lot of thought before writing a complicated expression in an IF statement. He should first determine if he needs that computed value in other statements; if so, the expression should be computed in an Arithmetic statement prior to executing the IF to avoid redundant calculation.

Q. Given the statement IF(Z-G)10,25,1001 and the value of -1.0 for Z and -2.0 for G, what statement would the computer execute next?

●●●

A. 1001 (Z-G>0)

35 The statements SAMPL = X**2-B*Z and IF(SAMPL) 10,20,30 executed in that order will have the same effect as IF(X**2-B*Z)10,20,30, but in the first case the value of the expression is saved in the variable SAMPL and will be available for future use.

Q. If the variables X, B, and Z in the example above each had a value of 2.0, the computer would select its next executed statement (in either case) as statement number _____.

●●●

A. 20 (X**2-B*Z equals 0.)

36 Remember that the parentheses of an IF statment contain an expression subject to all the rules of hierarchy and mode that apply to expressions in Arithmetic statements. In particular, remember that the expression normally contains quantities of the same mode only, except for exponents and subscripts.

Q. (Yes or No) Does the statement IF(HENRY+HOMER-JIM)1,2,3 contain a mixed mode expression?

●●●

A. Yes

37 The IF statement, then, contains three elements:
The word "IF"
An expression contained in parentheses
Three statement numbers indicating three possible paths for the computer.
All IF statements are constructed in this basic form, with just the few simple rules stated above to follow.

38 Perform Exercise 2.2 in your problem book.

Now that you know everything about IF statements, let's look at a few examples to show how they are used in FORTRAN programs. For example, suppose you wish to compute the square root of the quantity called X and you must first make sure that X is a positive quantity. The use of the IF statement to accomplish this is illustrated on Panel 2.1 in the Illustration Book. Turn to it now.

Q. If the value of X were a negative number, statement number _____ would be executed to change its sign.

●●●

A. 40

39 When statement number 40 on the panel example is executed by the computer, the value of X will have its sign changed and the computer will continue on to statement 30 ROOT = X**0.5 to compute the root. If the value of X had originally been positive, statement number 30 would have been executed immediately.

Q. The decision made by the IF statement of the preceding example was to ensure that the sign of the quantity whose root was being computed was _____.

●●●

A. positive (plus)

40 The example on the panel also illustrates two statements worthy of mention. The statement X = -X is a perfectly legal Arithmetic statement: the original value of X has its sign changed and the new quantity replaces that value in X. The statement ROOT = X**0.5 shows the use of the exponent 0.5 to compute the square root. This is true because in FORTRAN the equal sign denotes "calculate and replace" rather than "is identical to."

41 One common characteristic of good computer programming is the ability to re-execute a statement or sequence of statements in a process called a "loop". A basic series of operations may be defined by a sequence of statements which the computer will execute a number of times with different values of the variables involved.

Q. A basic sequence of operations which is executed repeatedly is called a _____.

●●●

A. loop

42 The IF statement can be a valuable aid in controlling the progress of a "loop". The example in Panel 2.2 will illustrate the use of the IF statement to compute "X-factorial" which is defined as the product of the quantities X, X-1, X-2, X-3, and so on down to 1. For example, 4-factorial is (4)(3)(2)(1) = 24. We assume that the real quantity X has a whole-number value.

43 Turn to Panel 2.2. Have you studied the program thoroughly? Here is the explanation: the first two statements are initialization steps: FACT = 1.0 prepares the variable FACT for the first multiplication and Y = X gives us a working variable Y such that the value of X is preserved.

Q. (True or False) The two statements described above are executed just once in the program segment shown on the panel.

●●●

A. True

44 The statements

```
10   FACT = FACT*Y
     Y = Y-1.0
     IF(Y)20,20,10
```

represents the actual loop: the first time through FACT becomes (1.)*(Y) and Y is reduced by 1.

Q. If Y had an initial value of 10.0 the IF statement would transfer to statement number _____ the first time through this loop.

●●●

A. 10 (Y>0.)

45 The loop described in the preceding frame will continue repeating, storing the successive products in the variable FACT as the factorial definition prescribes, reducing the working variable Y by steps of 1.0, and testing so that when Y has a value of zero the loop is finished and the computer will go to No. 20.

Q. Looking at the panel which shows the sample program segment discussed above, if the variable X had a value of 10.0, how many times would statement 10 be executed? _____

●●●

A. 10 (on the tenth execution of the IF statement, Y = 0.)

46 If you answered the last question correctly, you are grasping this loop concept pretty well. The idea of a loop in a program is to avoid writing similar statements over and over, wasting your time and energy and the computer's space and energy as well. One basic set of statements with proper control will often do the job.

Q. Statement 20 on the panel will be executed a total of _____ time(s).

●●●

A. 1

47 Another use of the IF statement might be testing a counter where a loop is to be cycled a prescribed number of times. The next few frames will describe such a program which will compute the sum of an arithmetic progression. If you are familiar with the term "arithmetic progression" you may skip to frame 49.

48 An arithmetic progression is simply an ordered sequence of numbers which differ by the same amount. The series 1,2,3,4,5,6,7,8,9, 10,.... is an arithmetic progression, as is 1,3,5,7,9,....etc. An arithmetic progression beginning with 2 having a difference of 4 would be 2,6,10,14,18,....and so on. A progression usually has a definite number of terms, also. The arithmetic progression beginning with 1 having ten terms and a difference of 2 would be: 1,3,5,7,9,11,13,15,17,19.

49 A program, or sequence of statements, to compute the sum of all the terms of any arithmetic progression must be given the following information: the first number, the difference, and the number of terms involved. Assume that the variables FIRST, DIF, and NUM respectively represent these quantities in the following example, and that they have been given values from some other part of the program.

50 Turn to Panel 2.3. Have you studied the sample program segment? Here is the explanation: the first two statements are initialization, setting SUM equal to the first term in the progression and setting the counter to one. The loop itself begins with statement number 10.

Q. The first two statements of the sample program segment are executed _____ time(s) each.

●●●

A. one

51 The next four statements: 10 FIRST = FIRST + DIF; SUM = SUM + FIRST; I = I + 1; and IF (I-NUM)10,20,20 constitute the "loop" such that the variable SUM has the quantity FIRST added to it each time through the loop and the counter I is increased by 1 each time through. As soon as the variable I has attained a value equal to NUM the loop is terminated by passing on to statement number 20.

Q. If NUM has a value of 15 in the loop shown above, statement number 10 will be executed a total of _____ times (careful, now!)

●●●

A. 14

52 Statement number 10 in this example is executed one less time than the value of NUM since I has a value of 2 by the time the IF statement is executed the first time and consequently will have a value equal to NUM after "NUM-1" cycles through the loop and will go on to statement 20 at that time.

Q. When statement 20 of this example is executed, the variable I will have a value of _____ (assuming NUM has a value of 15).

●●●

A. 15

53 This was admittedly a trivial example (the problem is better solved by direct substitution in a formula) but it was intended chiefly to illustrate the loop concept. IF statements can control the progress of a loop by testing either a counter or some other computed quantity in the loop to determine when to exit the loop.

54 Perform Exercise 2.3 in your problem book.

The IF statement is one of the simplest and yet most powerful of the Control statements. Another very simple statement which is discussed in the next few frames is the GO TO statement.

55 The GO TO statement consists of the two words "GO TO" followed by a single statement number. When a GO TO statement is executed the computer is told to execute next the indicated statement. For example, the statement GO TO 12 will tell the computer to go next to statement number 12.

Q. After executing the statement GO TO 1000 the computer will execute statement number _____ next.

●●●

A. 1000

56 The use of the GO TO statement is often called "unconditional branching" or "unconditional transfer" since no choice is made by the computer. Actually the GO TO statement usually appears at the end of a sequence of statements representing an option of some other control statement, and serves to direct the computer back to some other part of the program, or to skip some portion of the program.

57 Turn to Panel 2.4. The sample program segment on the panel shows an IF statement directing the computer to one of three options. No matter which path the computer takes, the GO TO statements will eventually direct the computer to statement number 40. Notice that, in this case, the third path does not need a GO TO statement since the program naturally arrives at statement number 40 without requiring control.

Q. In the GO TO statement, the words "GO TO" are always followed by a _____.

●●●

A. statement number

58 You have been shown some of the uses of two of the members of the control statement family. Both of these statements have the ability to tell the computer to go somewhere other than to the next statement as it normally would. The IF statement provides conditional branching while the GO TO statement involves unconditional branching.

59 A while back in this chapter you saw a discussion of a program with a loop in which an IF statement was used to test the value of a counter. The counter was a variable which was increased in value by some amount each time a pass through the loop was made.

Q. The loop approach described above requires increasing and testing a variable called a _____.

●●●

A. counter

60 To construct a loop using a counter and an IF statement to test its status requires separate statements to step the counter and test its current value. FORTRAN provides a powerful statement to keep track of both operations in controlling a loop: The DO statement.

61 The DO statement is constructed in the following form: the word "DO" followed by a statement number, and followed in turn by an index definition. An example of a DO statement might be DO 10 I = 1,100 where 10 is the statement number and I = 1,100 is the index definition.

Q. The DO statement always begins with the word _____.

●●●

A. DO

62 The index definition (e.g. I = 1,100) is always made up of a non-subscripted integer variable followed by an equal sign and at least two integer quantities separated by a comma.

Q. (True or False) The index definition ANUM = 1,1000 is valid according to the above definition.

●●●

A. False (ANUM is not an integer variable)

63 The DO statement tells the computer to "repeatedly execute the successive sequence of statements up to and including the statement whose number appears in the DO, cycling the index each time through according to the index definition." That is, all those statements following the DO represent a "loop."

Q. The statement DO 10 I = 1,100 will cause the computer to repeatedly execute the statements following the DO up to and including statement number _____.

●●●

A. 10

64 The index definition (e.g. I = 1,100) indicates that the variable on the left of the equal sign is set equal to the first quantity on the right of the equal sign for the first pass through the "loop", and the variable's value is increased by 1 on each successive pass until it reaches a value equal to the second quantity on the right of the equal sign.

Q. The index definition shown above would cause the variable I to take on values from _____ to _____.

●●●

A. 1,100

65 The DO statement, then, controls a loop by causing the computer to repeatedly execute the statements following the DO up to and including the indicated statement, beginning with the index variable set equal to the first indicated value and increasing its value until it equals the second indicated value, at which time the loop ends and the computer continues with the program.

Q. The statement DO 20 K = 1,50 will execute the statement through No. 20 a total of _____ times.

●●●

A. 50

66 Consider an example: the statement DO 10 I = 1,100 tells the computer to start with the very next statement and execute all the statements down to and including statement number 10, having given the variable I a value of 1. When statement number 10 has been reached the computer will go back to the first statement after the DO and repeat, having increased the value of I by 1. This goes on until I reaches 100.

Q. On the pass in which I of the above example reaches 100 the computer will go on to the statement immediately after statement number _____ when it finishes the loop.

●●●

A. 10

67 The format of the DO statement is very important, so let's review that again. The word DO, a statement number, (warning: no comma allowed at this point, normal as that may appear) an integer variable, an equal sign, and lastly, at least two integer quantities separated by a comma.

Q. Identify the valid DO statements below by a plus sign; indicate the incorrect ones with a minus sign:

```
DO 111 III = 1,1111      _____
DO 100 A = 10,20         _____
DO 10,J = 1,10           _____
DO A121 = NO,NONO        _____
```

●●●

A. +
- (index must be integer)
- (no comma permitted before J)
- (statement number must be numeric only)

68 The quantities to the right of the equal sign in the index definition may be integer constants or integer variables whose values are computed elsewhere. Thus, DO 5 I = 1,K; DO 215 JOB = N,100; and DO 100 L = L1, L2 are all examples of legal DO statements.

Q. Assume K equals 10 in the first example above. The computer will loop through statement No. 5 a total of _____ times.

●●●

A. 10

69 The use of variables on the right of the equal sign in an index definition permits the programmer to construct a loop whose number of cycles can be changed by a computation in the program (outside the loop). If you attempt to alter these values during the loop execution, it will have no effect, as the limits are already set.

Q. DO 5 I = 1,2 will cause the computer to loop through statement number 5 a total of _____ times.

●●●

A. 2

70 Once again, be warned that the DO statement format must be strictly observed. No comma is permitted between the statement number and the index definition. Also, integer constants or variables are permitted as index defining limits, but no expression involving a mathematical operation is permitted.

Q. (True or False) The statement DO 10 I = 1,N-1 is a legal DO statement.

●●●

A. False (N-1 is an expression involving a mathematical operation)

71 The first quantity on the right of the equal sign in the index definition (index lower limit) must be a smaller value than the second number (upper index limit) since the index will always be counted upward. If this rule is not observed, the loop will still cycle once (with the lower limit value) and then drop out.

Q. The loop described by DO 15 N = 10,1 will be executed one time with a value of _____ for N.

●●●

A. 10

72 The index definition permits a third quantity on the right of the equal sign: if it is desired that the index count up in steps other than one unit at a time, the desired step value may be written in the DO statement as the third quantity in the index definition, separated from the upper limit value by a comma.

Q. The first time through the loop prescribed by DO 5 I = 5,10 the variable I will have a value of _____.

●●●

A. 5

73 If, for example, you wished the index variable in a DO loop to take on the values of 1,3,5,7,....,99, you might write a DO statement as DO 8 I = 1,99,2 which would cause the index variable I to start with a value of 1 and loop repeatedly, adding 2 to its value each time through the loop.

Q. When the index variable in the example above exceeds a value of _____ the loop will be completed.

●●●

A. 99

74 With the third quantity in the index definition specified as other than 1, it would be possible to have a situation in which the index variable would never reach the exact value of the upper limit (for example, I = 1,100,2). In this case the loop is terminated after the index reaches the largest possible value without exceeding the upper limit (in this example: 99).

Q. In the statement DO 10 I = 1,11,3 the largest value attained by the index I would be _____.

●●●

A. 10

75 This third quantity in the index definition is called the index increment and the complete format of the index definition can be stated as: an index variable, an equal sign, a lower limit, an upper limit, and an optional increment. These last three items may be either constants or variables, but not expressions involving operations. If the increment is not otherwise specified, it is taken to be 1 as in the original definition.

76 The actual procedure used by the computer in interpreting a DO statement is as follows: set the index variable equal to the indicated first value (lower limit) and execute all the statements up to and including the indicated statement; increase the value of the index variable by the indicated increment (or 1 if none specified) and check the current value against the upper limit: if the index is less than or equal to the upper limit, go back and re-execute the loop; if the index has reached a value greater than the upper limit, drop out of the loop.

77 Actually, the DO statement is thought of as being executed once while its prescribed loop sequence is being cycled according to the DO control. Incidentally, a loop under control of a DO statement is usually referred to as a "DO-loop".

Q. The loop prescribed by the statement DO 1 I = 10,10 will be executed once with a value of _____ for the index.

●●●

A. 10

78 The next few frames will describe a simple example of the use of the DO-loop. The problem is to find the sum of 50 numbers in a list called A.

79 Turn to Panel 2.5. Have you studied the sample program segment? Here is the explanation: the variable SUM is initialized to a value of 0.0 (this variable will be used to accumulate the sum of the list called A): the DO statement then controls a loop of one statement, executing that statement 50 times. Each time statement 10 is executed the value of I, the index variable, is increased by 1 such that the reference to A(I) will refer each time to a successive number in the A list. When the loop is completed, SUM will contain the desired sum of the A's.

80 This example illustrates a couple of notable points. First of all, the use of the index variable is definitely permitted within the loop itself. In fact this is a very desirable feature of the DO-loop and many instances will arise where the index value is useful as a subscript, exponent, etc.

Q. In the preceding example, the statements DO 10 I = 1,50 and 10 SUM = SUM+A(I) show the use of the index variable I as both a counter and a _____.

●●●

A. subscript

81 The only restriction on the use of the index variable within the loop is that you do not change its value in any way such as using it on the left of an Arithmetic statement. It may be used as a term or factor in an expression if desired, but remember that it is an integer quantity and you may want to change its mode.

Q. The integer value of I may be converted to a real value in the variable XI by the Arithmetic statement _____.

●●●

A. XI = I

82 The second item of note in the sample program is the use of a variable as an accumulator for a summing process. The statement SUM = SUM+A(I) has been used in this way such that each time this statement is executed under the DO control a new number of the A array is added into the current sum. When the loop is finished, SUM contains the desired result. This is a commonly used technique.

Q. (True or False) In order to use the above-mentioned summing technique, the variable SUM must first be initialized to zero.

●●●

A. True

83 Any FORTRAN statement may be used in a DO-loop but the loop must not end with a control statement (such as IF or GO TO). If it is necessary to conclude a DO-loop with a controlled decision, a dummy statement must be used after the control statement, as in the following example.

```
      DO 10 I = 1,15
      IF(A(I)-1.)10,20,10
10    CONTINUE
```

84 The dummy statement used at the end of a DO-loop is the CONTINUE statement, consisting simply of the word "CONTINUE". Its use will be demonstrated in the following program which will determine the largest number in an array (or list) of 1000 numbers called A.

85 Turn to Panel 2.6. Have you studied the program? Here is the explanation: BIGA, the variable whose value will eventually be the largest number in A, is first set equal to A(1) to be used as a basis for comparison. The DO-loop then compares the current BIGA value with successive numbers in the A array. The IF statement that does this either goes to statement 30 if the new A value is larger (and then replaces the old BIGA value) or goes to statement 50 if the A value is not larger than the current BIGA. When the loop is finished, BIGA is the desired value.

86 The reason for the existence of the CONTINUE statement is to provide the common finishing point for the DO-loop. Remember, the definition of the DO statement required that the computer execute the statements through the one indicated in the DO statement. This type of problem which involves branching in the loop often requires the use of a CONTINUE statement to provide this finishing point.

Q. A DO-loop cannot end with a _____ (type) statement.

●●●

A. Control

87 Notice that the IF statement needs someplace to go if the new A value is not larger than the current BIGA value. The IF statement cannot go back to the DO statement without starting the loop over again; therefore, it must have someplace to go forward in the program, namely the last statement in the loop as prescribed by the DO definition.

88 The CONTINUE statement actually does nothing in the execution of the program; that is, it doesn't tell the computer to do anything. It merely serves the purpose of providing the DO statement with a reference number in a loop which might otherwise end with a Control statement.

Q. (True or False) Regardless of how many branch directions there are within a DO-loop, all the possible paths within the loop must lead to a single particular statement at the end of the loop.

●●●

A. True

89 All in all, the DO statement is pretty simple once you become familiar with the little details. All of the statements which comprise the loop must be between the DO statement and the statement whose number appears in the DO. You may tell the computer to jump around inside the loop but all paths must eventually lead to the statement marking the end of the loop.

90 Perform Exercise 2.4 in your problem book.

A branching statement may cause the computer to leave a DO-loop before the index has completely cycled in the normal fashion. When this happens, the index value is available for use in subsequent statements, and usually when such a situation occurs this value is of interest, as in the following example to locate the first occurrence of the number zero in an array of 2000 numbers called BLOCK.

91 Turn to Panel 2.7. The DO-loop in this example simply tests successive numbers in the BLOCK array, the IF statement going to statement 10 to repeat the loop until a zero is found in BLOCK. At this time the IF statement goes to statement 20 where I (the location in BLOCK where zero was found) has its value saved in LOC. If the loop goes all the way through BLOCK and does not find a zero, the computer goes to the statement after No. 10 which sets LOC to zero (to indicate that BLOCK does not have the desired number) and continues with the program.

92 The example serves to demonstrate a jump out of a loop before the index cycle is completed. This will be called a "special exit." The ordinary condition of leaving a DO-loop by completing the index cycle will be called a "normal exit" (as when statement 11 is reached).

Q. If a value of zero is found in BLOCK in this example, the computer will do a " _____ exit" from the loop.

●●●

A. special

93 As demonstrated in that example when a special exit is made, the current value of the DO index is intact and may be used in any desired way. If a normal exit is made, however, there is no guarantee that the index value is meaningful. This is no special problem, since in a normal exit the index upper limit is known.

Q. (True or False) In Panel 2.7, if no zero value is found in BLOCK, statement number 20 will not be executed.

●●●

A. True

94 The statements following the DO statement up to the statement indicated in the DO itself are called the "range" of the DO-loop. There are certain strict rules about the ranges of DO-loops which are mentioned in the next few frames.

Q. The range of the statement DO 1010 INDEX = I,J,K extends through statement number _____.

●●●

A. 1010

95 The first rule concerning DO-loop ranges states that you are not allowed to transfer (e.g. IF or GO TO) to any statement inside the range of a DO from outside its range. You can transfer to a DO statement itself, but not to any statement in its range; those statements are under control of the DO only.

Q. (True or False) The sequence DO 10 I = 1,100
10 A(I) = B(I); GO TO 10 is legal.

●●●

A. False (the third statement transfers into the range of the DO-loop)

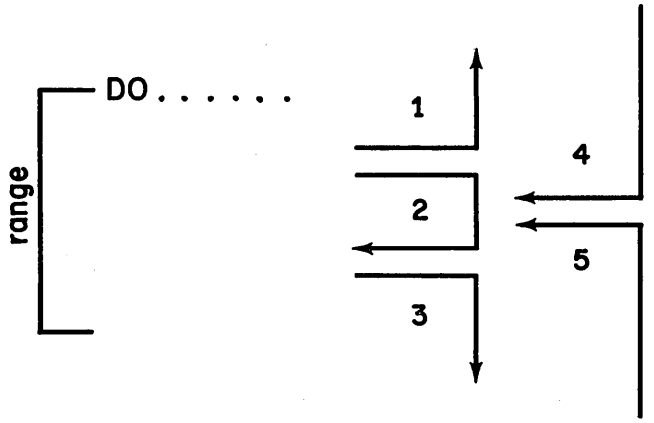
96 A second rule about ranges states that a DO-loop may exist completely inside the range of another DO-loop. When this structure is used, the innermost loop must be entirely contained by the outer loop; in other words, the loops must not overlap in any way. (This condition of loops within loops is further explained a little later.)

Q. (True or False) The sequence DO 10 I = 1,1000 .... DO 20 J = 1,10 .... 10 CONTINUE .... 20 CONTINUE is illegal because the second DO-loop extends beyond the range of the first.

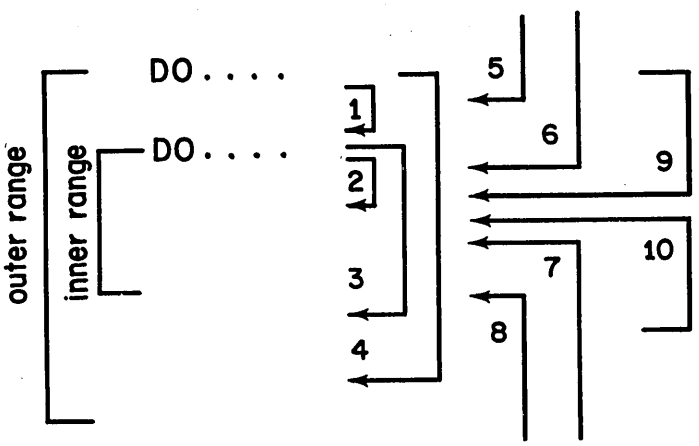
●●●

A. True

97 The arrow diagram below shows some of the legal branch conditions in DO-loops (numbers 1, 2, and 3) and also shows some of the illegal cases (numbers 4 and 5). The arrows simply show direction of transfer.



98 The arrow diagram below shows some legal (1,2,3, and 4) and illegal (5 through 10) branch conditions in nested DO-loops:



99 Q. Write a DO-loop to set to zero all the number values in a 1000- number array called X.

10

Use I for the index variable name and statement number 10 for the range of the loop.

●●●

A.
```
      DO 10 I = 1,1000
10    X(I) = 0.0
```

100 If your answer agrees with the one shown, skip to frame 105. If your answer does not agree with this one, continue to the next frame.

101 Perhaps going over the problem statements in detail will help. You were asked to write a loop to set the value of each number in an array to zero. With your present knowledge of DO-loops, this should imply a DO statement with an index definition which will cycle the index variable from a value of 1 up to the length of the array (1000).

102 The statement DO 10 I = 1,1000 will cause statement number 10 to be executed a total of 1000 times, and it remains for the programmer to specify statement number 10 such that it will perform the desired task: setting the value of successive values in the X array to zero.

```
10      X(I)= 0.0
```

(Note: the zero could be written a number of ways, as long as the decimal point is included to make it a real constant.)

103 Each time the statement 10 X(I) = 0.0 is executed under control of the DO, the index I is 1 larger than the previous time. Thus X(I) refers to successive numbers in the X array each time the statement is executed, as follows: X(1), X(2), X(3), X(4),....,X(1000).

104 The basic use of DO statements should be clear by now, and if you feel that you understand the make-up of this statement you may go to the next frame. If any points need clarifying, you had better review this material starting with frame 61. If, after this review, you still need help, see your advisor.

105 Perform Exercise 2.5 in your problem book.

A few frames back mention was made of DO-loops containing other DO-loops, a situation commonly called "nesting" of loops. Under these circumstances the outer loop is begun first and, during its first pass, it gives control to the inner loop which may completely cycle itself before restoring control to the outer loop.

Q. (True or False) When control passes to an inside loop, it may remain there until the inner loop is completely cycled.

●●●

A. True

106 When an outside loop regains control from an inner loop, it continues through its pass, reaching the end and repeating as usual. In the course of its next pass, it again will completely cycle the inner loop. This same thing happens on all subsequent passes also. A simple analogy will be given in the next frame to give a better picture.

Q. (True or False) With nested loops the inner loop is completely cycled each time a pass is made through the outer loop.

●●●

A. True

107 Picture the following situation: suppose you were walking around the grounds of an amusement park, say in a circular fashion. Each time you pass the merry-go-round you get on and take a ride around the circle a certain number of times. That ride would be analgous to an inner loop and the walk to an outer loop.

Q. (True or False) In this analogy the "inner" loop is completely cycled each time through the "outer" loop.

●●●

A. True

108 To carry this parallel one step further, if you decided to jump off the merry-go-round before it stopped, you might continue walking around the park or you might decide to go home. These cases correspond to permitted branch situations in nested DO-loops.

Q. If you got off the ride before it stopped this would be equivalent to a " _____ exit" but if you were to wait for the merry-go-round to finish its run this would be the same as a " _____ exit" in DO-loops.

●●●

A. special, normal

109 Problems seldom arise that require more than two or three loops within loops. Quite often you will want to use DO-loops end-to-end inside another DO-loop (sort of like a merry-go-round and a ferris wheel in our carnival analogy).

Q. The sequence

```
      DO 10 I = 1,10
      DO 10 J = 1,10
10    A(I) = A(I)+B(J)
```

will cause statement number 10 to be executed a total of _____ times.

●●●

A. 100

110 When a special exit is made from an inner loop of a nested set of loops, the index values of all loops are preserved and available for use, including, of course, the index of the exited loop. Turn to Panel 2.8. This is an example of nested DO-loops. This program looks through two arrays looking for the first duplication of values between the arrays and records this location (index) in the arrays.

111 Have you studied the program? Here is the explanation: the outer DO sets I to 1 and goes to the second DO statement, setting J to 1. Thus the IF statement compares A(1) and B(1) the first time through. The inner DO then sets J to 2 (I is still 1) and compares again, and so on until J equals 10. At this time the value of I is increased to 2 and the inner loop begins again, J going from 1 to 10. When the A and B values become equal, the IF statement will jump out to No. 20 and record the locations (index values).

112 You will notice that, even when two nested DO-loops end at the same statement, as in this example, control is passed to the outer loop only when the inner loop has completed its index cycle. Thus it is perfectly permissible for the two loops to end their respective ranges on the same CONTINUE, as done here.

Q. If the IF statement in Panel 2.8 causes a special exit to statement No. 20 during the 14th time it is executed, the values placed in LOC(1) and LOC(2) are, respectively, _____ and _____.

●●●

A. 2,4

113 If you were able to answer the previous question correctly, you are showing excellent progress and you may continue to the next frame immediately. Here is the explanation: if the IF statement is executed 14 times in that program we know that the inner loop was completely cycled once (10 executions of the IF) and has gone 4 times through on the second execution of the outer loop; hence, I has a value of 2 and J a value of 4.

114 Perform Exercise 2.6 in your problem book.

One important note: the index variables in nested DO-loops must be different in name; that is, you are not allowed to use the same index for two DO-loops if one contains the other. However, there is no reason why you cannot use the same variable name in sequential DO-loops, since the index variable is free for any desired use in the program once a DO-loop is through with it.

Q. (True or False) The sequence DO 5 I = 1,200; DO 5 X = 1,40,2; (some statement) is legal.

●●●

A. False (the index variables, though different, must both be integers)

115 Turn to Panel 2.9. One last example of nested DO-loops demonstrates the use of the outer loop's index variable to determine the inner loop's index limits. In this program the outer loop is actually a counter to re-execute the inner loop a sufficient number of times to complete its function which is to sort an array of numbers into algebraic order.

116 This program uses the technique of comparing adjacent numbers in an array, reversing the numbers if they are not in the correct algebraic sequence. The first time through the loop ARRAY(1) is compared to ARRAY(2), the second time ARRAY(2) is compared to ARRAY(3), etc. Thus if the largest value of the array were in the first position, by the time the program had finished the inner loop once it would be in the last position in the list (with each number moved down one slot). This will happen, in fact, no matter where the largest number is.

117 After the inner loop is completely cycled once, the outer loop's index I changes its value to 2 and the inner loop begins again with a new upper index limit. This means that the inner loop is cycled one less time, resulting in one less comparison of number pairs. This is possible due to the fact that the largest number in the array is already in correct position at the end of the list and need not be checked again.

118 Each complete execution of the inner loop will result in the largest of the remaining numbers ending up at the highest available position in the list, no matter where it is located in the array before the sorting begins. Thus the completion of this sample program finds the list called ARRAY sorted into algebraic order.

Q. In the sample program on Panel 2.9, if N (the length of the array to be sorted) has a value of 100, the inner loop will have an upper index limit of _____ the first time it is executed.

●●●

A. 99

119 The IF statement in the inner loop of our sample program checks the successive pairs of numbers in ARRAY and, if they are in correct order (or equal), they are not changed and the loop begins again; however, if they are in reverse order, statement 10 is called on to reverse the position of these numbers in ARRAY.

Q. If ARRAY(1) is larger than ARRAY(2) in the original position, the IF statement will go to statement _____ the first time it is executed.

●●●

A. 10

120 Notice the sequence at statement number 10 in this example: a special variable (TEMP) is used to save one of the values when the swap is made. This is necessary because if one value were placed in the other's position without first preserving its original value, that original value would be erased and lost forever.

121 So far, in this chapter, you have been introduced to four of the Control statements in the FORTRAN language: the IF statement which provides three possible branch directions based on the value of an expression; the GO TO statement which directs the computer unconditionally to a particular part of the program; the DO statement which controls looping; and the CONTINUE statement which provides the DO statement with a reference point when the loop might otherwise end up with a Control statement. These four statements are the most important of the Control statements, and are required in most programs much more often than the other Control statements.

122 Perform Exercise 2.7 in your problem book.

The IF statement, you will recall, has three possible branch directions. The next few frames will explain a Control statement which allows branching to more than three paths.

123 The multiple-branching statement is a special form of the GO TO statement called the "Computed GO TO" statement. It begins with the words "GO TO" followed by a pair of parentheses containing any desired amount of statement numbers separated by commas. The close-parenthesis is followed by a comma and a non-subscripted integer variable. An example of a Computed GO TO is shown below:

```
GO TO (10,9,8,7,6,5,4,3,2,1),N
```

124 As in most Control statements, the format is most important, so let's go over it once more: the words GO TO, an open-parenthesis, a series of statement numbers separated by commas, a close-parenthesis, a comma, and lastly an integer variable. Each of these items must be present.

Q. (True or False) The statement GO TO (1,2,10) I is a legal computed GO TO statement.

A. False (a comma must be used to separate the close-parenthesis and the variable)

125 The Computed GO TO is executed as follows: the computer will go next to the statement whose statement number appears in the parenthesized list in the order corresponding to the value of the integer variable. That is, if the variable has a value of 1 the first number in the list is the next statement; if the value is 2 the second statement number is the next statement, and so on.

Q. In the statement GO TO (10,9,8,7,6,5,4,3,2,1),N if N has a value of 3 the computer will go next to statement number _____.

●●●

A. 8

126 The variable's value does not have a direct bearing on the statement number chosen for the next statement to be executed. For example, in the statement GO TO (10,9,8,7,6,5,4,3,2,1),N a value of 10 for the variable N will not direct the computer to statement number 10. Rather, N acts like a subscript to the list of numbers.

Q. In the example shown above, a value of 10 for the variable N will direct the computer to statement No. _____.

●●●

A. 1

127 The list of statement numbers may be as long as desired and the same number may appear more than once in the list if desired (as was permitted in IF statements). Care must be taken to see that the control variable on the right does not have a value larger than the number of statement numbers in the list.

Q. In the statement GO TO (312,311,312,1),K the value of K must never exceed (number) _____.

●●●

A. 4

128 The Computed GO TO is used whenever it is convenient for the computer to select one of several branch options through the value of a variable. Turn to Panel 2.10. Suppose, for example, that you wish to go to different parts of the program depending on the value of a variable X which has a value from 0. to <10. If the value is from 0. to <1.0 you wish to go to statement number 1; if the value is from 1.0 to <2.0, you wish to go to No. 2, etc.

129 This program uses the value of the tested variable X to form a control variable K. In a real expression the variable X has 1.0 added to it, then this value is converted to integer mode and placed in K. If X had a value from 0. to less than 1.0, K would have a value of 1; for X from 1. to less than 2. K would be 2, etc.

Q. (True or False) K will always have an integer value due to the truncation of fractions when converting from real to integer numbers.

●●●

A. True

130 This Computed GO TO lists the options from which the computer chooses, depending on the value of the control variable K. Remember, K's value depended on the value of X. When the GO TO is executed, the statement selected will indirectly depend on the value of X as outlined in the problem statement.

Q. Looking at the sample program segment on the panel, if the value of X were exactly 10.0, what statement would the computer choose (K would be 11)? _____.

●●●

A. 10

131 Perform Exercise 2.8 in your problem book.

The following is a list of the Control statements described so far in this chapter:

IF
GO TO
DO
CONTINUE
Computed GO TO

132 The next two Control statements are used to bring the computer to a halt. As such they do not quite fit the general definition of Control statements (telling the computer where to go next) but they do exercise some "control" in their execution.

133 The first of the halting Control statements is the STOP statement. This consists very simply of the word "STOP" and its execution is exactly what the name implies. The computer is brought to a halt and may not be restarted without beginning all over again. For example:

```
(Your program statements)
STOP
```

Simple enough?

134 A similar statement which will also cause the computer to halt is the PAUSE statement, which consists of the word "PAUSE". When the PAUSE statement is executed the computer will halt, but if the START button on the computer console is pushed, the computer will continue on in the program at the statement following the PAUSE.

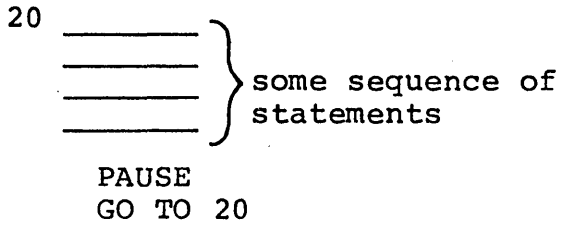
Q. The STOP and PAUSE statements both cause the computer to halt but the _____ statement will permit the computer to start up again when the START button is pushed.

●●●

A. PAUSE

135 In addition to causing the computer to halt, the PAUSE statement will also cause the message "PAUSE 0000" to be printed on the system's printer.

Q. In the following example, what will be printed on the printer?



●●●

A. PAUSE 0000

136 Because a program can contain many PAUSE statements, it may be desirable to know just which PAUSE statement has caused the computer to halt. To uniquely identify a PAUSE statement the programmer can write the word "PAUSE" followed by an integer number of up to four digits. For instance, if PAUSE 1234 were written by the programmer, the computer will print "PAUSE 1234" when that PAUSE is executed.

Q. Write the statement which will halt the computer and cause "PAUSE 0023" to be printed.

●●●

A. PAUSE 0023 or PAUSE 23. In the latter case, two leading zeros will be supplied automatically by the computer to fill out the printed message to four digits.

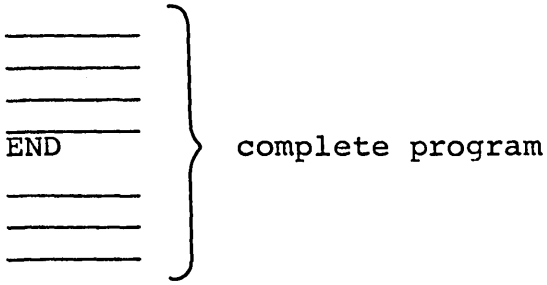
NOTE: The STOP statement may also have an identifying number.

137 Perform Exercise 2.9 in your problem book.

The last statement type covered in this chapter is the END statement. This statement is different from all the other statements covered so far in this course. It is not "executed" but rather acts as a source of information to the program, and as such is actually a specification statement. Its function is very simple: it marks the last statement in the written program.

138 The END statement consists simply of the word "END", and must be the last written statement in a program.

Q. Would the following program be correct?



●●●

A. No. The END statement must be the last written statement.

139 As pointed out in the previous frame the only function of the END statement is to mark the last written statement in a program. That is, it has nothing whatever to do with halting the computer or any other "executable" operation. Its purpose is simply to tell the computer during the FORTRAN translation stage that the end of the source program has been reached. All programs written in FORTRAN are incomplete until an END statement is placed after the last statement in each source deck.

Q. Is the END statement mandatory at the end of a FORTRAN program?

●●●

A. Yes.

140 The following is a list of Control statements covered in this chapter:

IF
GO TO
DO
CONTINUE
Computed GO TO
STOP
PAUSE
END

141 You may proceed to the examination on page 33 of your problem book after which you may report to your advisor.

REFERENCE INDEX

R29-0102-0

Printed in U.S.A.

R29-0102-0

IBM

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, New York