IBM

FORTRAN for the IBM 1130

Chapter 3

Programmed Instruction Course

IBM

FORTRAN for the IBM 1130

Chapter 3

Programmed Instruction Course

FOREWORD

You have been shown how to construct the middle part of a computer program, the part in which you tell the computer how to manipulate the numbers and also how to make simple tests and decisions from which it executes control. Most programs require that the computer also be able to accept information at the start of the program and to divulge the results at the conclusion. In other words, the computer must be capable of reading and writing as well as performing calculations. Various computer systems use different media for these operations of "input/output"; most commonly the input information comes from punched cards, punched tape, magnetic disk, or magnetic tape, while output data is usually written on a printer or magnetic tape or disk or, less commonly, punched in cards or paper tape. The 1130, for example, has a card reader/punch and a typewriter or printer in a typical system. A paper tape reader and paper tape punch may also be included.

This chapter will discuss the input/output statements used by the FORTRAN language.

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.

15 Incidentally, the variables in the list of a READ statement may be of any mode. Moreover, both integer and real variable values may be specified in the same list with no taboo about mixing the modes.

Q. (True or False) According to the above definition, the statement READ(2,1)A,B,C,I,J,K is not valid.

●●●

A. False (it is perfectly valid)

16 The variables in the READ statement list may be subscripted or not as desired. The subscripts, if used, may be constants or previously defined variables, but absolutely no form of mathematical expression is permitted. For example, the statement READ (2,1)A,B,X(3),N(K) will read four numbers from an input unit assigning the values respectively to A,B, the third number in the X array, and the Kth number in the N array.

Q. (True or False) The statement READ(5,1)A(2*K-5) is a valid READ statement.

●●●

A. False (the subscript is a mathematical expression)

17 The significant points covered so far have been: the computer can read data at the time the program is being run, reading from a variety of media. The READ statement causes the computer to read information from an attached input device. The READ statement contains a list of variables which specifies where the numbers being read are to be assigned. The variables in the list may be of either mode and may contain subscripts.

Q. Write a statement to read a value for each of the variables ALPHA, BETA, and GAMMA, in that order. Use 4 as the input reference number, and 1 as the statement number.

●●●

A. READ(4,1)ALPHA,BETA,GAMMA

18 If your answer agrees exactly with the one shown above, skip to frame 22. If you have omitted any of the three commas, take note of where they belong and skip to frame 22. If your answer was otherwise wrong, go to the next frame.

19 The question called for a READ statement. It would start with the word "READ", followed by a set of parentheses containing the input reference number (which we said was 4) and the statement number (which we said would be 1).

Q. Write the beginning of the read statement up to and including the close parentheses.

●●●

A. READ(4,1)

20 The rest of the statement consists of the list of variables to be defined: ALPHA, BETA, and GAMMA. Each name must be separated from the others by a comma.

Q. If we substituted JOE, JIM, and JACK for ALPHA, BETA, and GAMMA, the complete statement to read values for the variables would be: _____.

●●●

A. READ(4,1)JOE,JIM,JACK

21 If you answered this satisfactorily, go to the next frame; if not, go back to frame 3 and review this material.

22 Perform Exercise 3.1 in your problem book.

The associated statement whose number appears in a READ statement plays an important part in the reading operation. This statement is called a FORMAT statement and its job is to describe to the computer the form of the number being read. For instance, if cards were being read, the FORMAT statement would tell the computer which card columns are to be read, and how many numbers there are per card. (For instructional purposes, we will assume from here on that the information entering the computer is coming from cards, unless otherwise noted.)

23 Nearly all Input statements use a FORMAT statement to specify the way the numbers are laid out in the cards, or other external medium. The general rule is that the Input statement tells the computer what is to be read and the FORMAT specifies how and where the numbers will be found within the record.

Q. (Yes or No) Does the statement READ(2,1)A,B,C tell you how many cards are to be read? _____.

●●●

A. No (FORMAT statements will be covered in more detail later.)

24 Most all Input statements contain a list of variables which specifies the quantities being read. The next few frames will explain a few details of how the list is used.

25 When an Input statement such as a READ statement is executed, the values being read are immediately defined as the values of their corresponding variables in the list. Thus, an integer value read early in the list can be used as a subscript later in the list, e.g., READ(2,1)K,(A(K)).

26 When the statement READ(2,1)K,(A(K)) is executed, two numbers will be read from a card. The first number, an integer quantity, will become the subscript of the second number immediately. In this fashion a deck of cards can be indexed and need not be sorted into any special order.

Q. If the value of K is 3 on a particular card, the second number on that card will be assigned to the _____ position of the A array, using the statement shown above.

●●●

A. third

27 When this feature of using a read-in value as a subscript in the same list is employed, a special restriction must be observed: at least one open-parenthesis (other than the normal subscript parentheses) must separate the two references to that integer variable, as in READ(2,1)K,(A(K)). Naturally it must also be balanced with a close-parenthesis.

Q. The statement READ(2,1)I,J,K,(A(I),B(J),C(K)) will cause a total of _____ values to be read.

●●●

A. six

28 When variables appear in an Input list with variable subscripts, these subscripts need not necessarily be defined within the same list as in the preceding example. Their subscript values may have been defined by other statements (by a DO-loop index, for example).

29 Turn to Panel 3.2. The sample program segment on the panel shows a READ statement which will be executed a total of ten times. Each time it is executed a value will be read and placed in the A array, its position determined by the DO index value.

Q. In the sample program on the panel, the sixth time statement 5 is executed, a value will be assigned to A (_____). (Fill in a number).

●●●

A. 6

30 Turn to Panel 3.3. You now have seen two examples of subscripted variables in an Input statement list (see panel). The first example showed the subscript value being defined within the same list; the second example showed another means of defining the subscript's value (notice, no extra parentheses required in this second case). One more example similar to the second case is shown in Panel 3.4.

31 Turn to Panel 3.4. The first READ statement in the example on the panel causes the computer to read a single quantity, N, which then becomes the upper limit of a DO-loop. Thus, the read-in value of N determines how many numbers will be read within the DO-loop itself.

Q. If the value read for N were 50, statement number 5 on the panel would be executed (how many?) _____ times.

●●●

A. 50

32 An Input statement's list of variables may have subscripts which are also variables. As shown in the preceding examples, these variable subscripts may have their values defined either by reading them in the same list or by other statements preceding the READ statement (by a DO-loop, for example).

33 Perform Exercise 3.2 in your problem book.

A DO-loop, then is one method for programming the reading of arrays. Another method, more widely used, is the self-indexing list technique. In this type of Input statement list, the variables with their variable subscripts are followed by an index definition exactly as defined in a DO-loop; for instance:

```
READ(2,1)(A(I),I=1,50)
```

34 The statement READ(2,1)(A(I),I=1,50) is exactly identical in operation to the statement: READ(2,1)A(I),A(2),A(3),A(4)..., A(50). This statement, though only executed once, will cause the reading of 50 numbers into successive positions in the A array.

Q. The statement READ(2,1)(ARRAY(K),K=1,1000) will cause a total of _____ numbers to be read.

●●●

A. 1000

35 The index definition as used in input lists may utilize the same features available to DO-loops. It may have variables for lower or upper limits and it may use the optional third quantity defining an increment other than one (which may also be a variable if desired).

36 Take the statement READ(2,1)N,(A(I),I=2,N) for example: in this statement the index definition contains a variable as the upper limit; the value of this variable is read in the same list. The execution of this statement is to read an integer quantity, N, and then read values into A(2) through A(N), inclusive.

Q. If the first number punched in the first card read by the above statement were 15 a total of _____ values would then be read into the A array.

●●●

A. 14

37 This self-indexing input list is a very useful shorthand method of reading long lists of quantities. You can well imagine the cumbersome list that would result if you read in a lot of numbers by naming each one in the input list.

Q. If you wished to read 100 numbers into the A array with the statement READ(2,1)N,(A(I),I=1,N) the first number on the first card must be punched as _____ (number).

●●●

A. 100

38 Like most other FORTRAN statements, the format you must observe is rather strict: the subscripted variable (or variables) must be separated from the index definition by a comma (and from each other, if more than one); the entire sequence - variables, dummy subscripts, and index definition - must be enclosed in parentheses.

Q. The statement READ(2,1) A(I),I=1,10 is not valid because of the absence of _____.

●●●

A. parentheses
READ(2,1)(A(I),I=1,10) is correct.

39 An input list may contain other elements either before or after a self-indexing sequence. For example, the statement

READ(2,1)A,B,C,(X(I),I=1,3),D,E,F

is perfectly valid. The nine values read by this statement will be assigned respectively to A,B,C,X(1),X(2),X(3),D,E, and F.

Q. If the upper limit of the index definition in the example shown were 10 instead of 3, a total of _____ numbers would be read by this statement.

●●●

A. 16

40 More than one variable may be included in a self-indexing sequence, as, for example, in the statement READ(2,1)(A(I),B(I),I=1,100). This will cause 200 numbers to be read and assigned in the order A(1), B(1), A(2), B(2), A(3),.....,A(100), B(100).

Q. The sixth number read by the statement shown above would be assigned to _____ (variable and subscript).

●●●

A. B(3)

41 An input list may also contain more than one self-indexing sequence as in the example statement READ(2,1)(A(I),I=1,10),(B(I),I=1,10). This will cause the computer to read twenty numbers and assign them in the order A(1), A(2), A(3),..., A(10), B(1), B(2), B(3),...,B(10).

Q. The number assigned to B(3) as read by the statement shown above would be the _____ number (in the order) read by the computer.

●●●

A. thirteenth

42 Notice that each separate self-indexing sequence has its own pair of parentheses. Any self-indexing list sequence must be so enclosed in parentheses, and these parentheses must contain only the variable (or variables) whose subscript is indexed, plus the index definition.

Q. Write a statement to read values for the first five numbers in the BLOCK array, using self-indexing form.
_____.

●●●

A. READ(2,1)(BLOCK(I),I=1,5)

43 If your answer agrees with the one shown above, or is different only in the arbitrary choice of the dummy name, skip to frame 49. If your answer did not agree with the one given, continue to the next frame.

44 Perhaps a re-definition of the self-indexing form is in order. Begin the list with an open parenthesis, followed by the array name; attach to this name, in parentheses, a "dummy" subscript; then follow it with a comma and an index definition which defines the values to be taken by the dummy subscript. The list is ended with a close parenthesis.

45 The index definition is exactly the same as that used in a DO statement. If you want the index to cycle through values of 1 to 5, for example, you would use the sequence I=1,5. Although the choice of the dummy index name is arbitrary, it must be an integer variable.

46 The entire list includes the array name, subscripted with the dummy index, followed by the index definition. For this problem (to read values for the first 5 numbers in BLOCK) the list should be:

(BLOCK(I),I=1,5)

Remember, those two commas are absolutely necessary; don't forget them!

47 The entire statement required for that answer, again, was READ(2,1)(BLOCK(I),I=1,5). Try another case similar to that one.

Q. Write a statement to read values for ALPHA, BETA, GAMMA, and the first 12 numbers in the OMEGA array, in that order. _____

●●●

A. READ(2,1) ALPHA,BETA,GAMMA,(OMEGA(I),I=1,12)

48 If your answer agrees with the one shown above, continue to the next frame. If you missed this question, you had better go back to frame 33 and read again, carefully, the material on self-indexed lists.

49 Perform Exercise 3.3 in your problem book.

Recent examples showed two methods of recording numbers into two different blocks with one statement. In one case, both variables were controlled by the same index definition; the other case showed two separate self-indexed sequences. In the first case the numbers being read alternated from one block to another while the second case showed that one block was filled before any numbers were read for the second block.

READ(2,1)(X(I),Y(I),I=1,1000)

50 The statement above illustrates the first case with two variables under control of the same index. This approach would be used ideally when the data is apt to be found in pairs (such as rectangular coordinates). At least, if you programmed this statement, you would have to arrange the data in alternating order in the cards.

Q. (True or False) The statement shown above will read the first 1000 numbers into the X array.

●●●

A. False (every other number will be read into the X array)

51 READ(2,1)(X(I),I=1,1000),(Y(I),I=1,1000)

This statement shows the other form; two self-indexed sequences used end-to-end. This method would be used where the data would be more easily punched in separate blocks on the cards, perhaps where the arrays are unrelated sets of numbers.

Q. (True or False) The first 1000 numbers read by the statement above would be assigned to the X array.

●●●

A. True

52 In any event when a READ statement is written into a program and the program is being run, the data punched on the cards must be in the form and order expected. Usually the READ statement is written first and the cards are punched later, although sometimes the statement is tailored to existing card decks.

Q. (True or False) If a card deck has consecutive pairs of of x and y coordinates (100 pairs in all), a statement such as READ(2,1)(X(I),Y(I),I=1,100) should be used to read this deck.

●●●

A. True

53 You might question the usefulness of the self-indexing technique, since a DO statement could conceivably perform the same function. For example, the sequence DO 10 I=1,1000; 10 READ(2,1) X(I) would admittedly perform the same operation as READ(2,1)(X(I), I=1,1000). There is a reason, however.

Q. The first example above would execute the READ statement _____ time(s) and the second example would execute the READ statement _____ time(s).

●●●

A. 1000,1

54 There is one slight difference, as pointed out by the last question and the answer shown above. The DO-loop must necessarily execute the READ statement 1000 times, reading one number each time. The self-indexed version executes the READ only once while cycling its own index and reading the 1000 numbers.

55 As the preceding frame points out there is one slight but important difference in the DO-loop and self-indexed methods: the number of times the READ statement is executed. Each time a READ statement is executed a new card must be read by the computer.

Q. When the DO-loop method executes the READ statement 1000 times, at least _____ cards will have to be read.

●●●

A. 1000

56 The DO-loop method executes and re-executes the READ statement, and each time it does so, a new card must be read. The self-indexing method, on the other hand, depends on the FORMAT statement to indicate how many numbers are punched on each card and, in this way, determines when a new card must be read.

Q. The statement READ(2,1)(X(I),I=1,1000) is executed a total number of _____ time(s) to read 1000 numbers.

●●●

A. one (the actual number of cards read, however, depends upon the FORMAT statement)

57 Now the advantage of the self-indexing method is becoming clear. The data cards may be utilized much more efficiently if more than one number is allowed per card. For example, if the associated FORMAT statement permits ten numbers per card, only 100 cards would be required to read the 1000 numbers of the preceding example using self-indexed lists, instead of the 1000 cards required with the DO-loop. (Note: It is possible to program a DO-loop and a READ statement to handle reading efficiently, but the programming is more complex than that in a self-indexed READ statement. The best plan, therefore, is to use the latter.)

58 To review: the READ statement can cause the computer to start reading a new card in the attached card reader and to read enough cards to supply values for all the variables in the READ statement's list of variables. The list may contain combinations of the following items: non-subscripted variables, variables with either constant or variable subscripts, or self-indexed sequences with the dummy subscript and index definition.

59 Perform Exercise 3.4 in your problem book.

So far, we have assumed the data being read into the computer has been coming from cards. The 1130, however, also has the ability to read data punched on paper tape. Paper tape is often used where it is necessary to transmit data from one location to another over telephone or telegraph lines.

60 The READ statement used for paper tape is exactly the same as the READ statement used in reading cards! This happy circumstance comes about because it is the input reference number that determines what input device is to be selected.

Q. (True or False) The statement READ(4,1)A,B,I can read only cards.

●●●

A. False (this statement reads paper tape)

61 To repeat, the FORTRAN statement to read numbers from paper tape is the READ statement, constructed in exactly the same way as if we were reading cards.

Q. (True or False) The input reference number assignment determines whether cards or tape will be read

●●●

A. True

62 The READ statement when used for paper tape causes the computer to act in a manner similar to that when reading cards: the computer starts the tape in motion and reads numbers for the variables in the list according to the associated FORMAT, just as before.

Q. Is the statement READ(4,1)X,Y a valid statement for reading paper tape? (Yes or No) _____

●●●

A. Yes.

63 Let's look at another sample statement: READ(4,1)A,B,C which tells the computer to read three values from the paper tape reader according to FORMAT statement number 1, and assign these values, respectively, to A, B, and C.

Q. Like the card READ statement, the variables whose values are to be defined are specified in a _____.

●●●

A. list

64 The list of variables is subject to exactly the same rules as before. Variables, either with or without subscripts, appear in the list in the order in which their values are to be found on the tape. The self-indexing form may be used.

Q. The statement READ(4,1)(A(I),I=1,1000) will read _____ values from the paper tape reader.

●●●

A. 1000

65 Actually, if you understand card READ statements and their lists, you automatically understand the paper tape READ statements. They are identical. Their operation is identical also except for the medium (cards or tape) selected.

Q. (Assume our familiar FORMAT number 1 is still with us.) Write a statement to read in the value of X,Y,Z,XX,YY, and ZZ from the paper tape reader.

●●●

A. READ(4,1)X,Y,Z,XX,YY,ZZ

66 If your answer agrees with the one shown above, skip to frame 75. If your answer is wrong, continue to the next frame.

67 Don't make this problem too difficult; it isn't. Given the variables (X,Y,Z,XX,YY, and ZZ), whose values are to be defined, and noting that none of them are arrays, makes the list construction a simple matter: just list the variable names in that order.

Q. Write the list only for the problem just completed. _____

●●●

A. X,Y,Z,XX,YY,ZZ (no parentheses or anything else; just the actual list of names)

68 Given also that FORMAT number 1 and input reference number 4 are to be used, the construction of the entire statement to solve that problem is now a simple matter of "fill in the blanks" with those numbers.

Q. Write the statement. _____

●●●

A. READ(4,1)X,Y,Z,XX,YY,ZZ

69 Now let's try another similar problem and see how you are making out. Write a statement to read values for ANN, BETTY, CATHY, and the third number in the GIRL array, reading from the paper tape reader (use input reference 4) with FORMAT No. 1. _____

●●●

A. READ(4,1)ANN,BETTY,CATHY,GIRL(3)

70 If your answer agrees with the one shown, continue to the next frame. If you are still having difficulty, turn back to frame 63, and review the material on tape-reading. If, after review, you still have difficulty, see your advisor.

71 Q. Write a statement to read values for M and N which in turn will be the lower and upper index limits, respectively, of the BLOCK array which is to be read in the same list. Read from the tape reader (use input reference number 4) with FORMAT No. 1. _____

●●●

A. READ(4,1)M,N,(BLOCK(I),I=M,N) (all parentheses required)

72 If your answer agrees with the one shown (except for possible choice of index variable), skip to frame 80. If your answer is wrong, continue to the next frame.

73 This was a considerably tougher problem than the previous one. You were asked to write a statement to read two numbers (integer) and, using these numbers as the lower and upper index limits respectively, read numerical values for the BLOCK array with self-indexed notation. First of all, the statement begins as usual: READ(4,1).

74 It seems the construction of the list for the tape reading statement poses the major problem. First of all, the names M and N must be listed by name (separated by commas), and then the self-indexed list for the BLOCK array follows. This latter element must contain the name, a dummy subscript, and the index definition with M and N used as the lower and upper limits, respectively.

Q. Using I as the dummy subscript and index, write the complete list as requested above. _____.

●●●

A. M,N,(BLOCK(I),I=M,N)

75 The complete statement, of course, is

```
READ(4,1)M,N,(BLOCK(I),I=M,N)
```

If you are convinced that you understand the foregoing, continue to the next frame; otherwise, review some more, starting with frame 61.

76 Perform Exercise 3.5 in your problem book.

Considerable mention has been made of the FORMAT statement without defining what it really is. The FORMAT statement is a special type of statement. It is not executed in the sense that the other statements are. It falls in a class of statements called "specification" statements, and provides information to the computer, or more specifically, to the input/output statements.

77 The FORMAT statement, then, is just a reference source for an input (or output) statement. When a READ statement, for example, is being executed, it "consults" the indicated FORMAT statement to find out certain items of information about how the numbers are found on the cards.

Q. When a READ statement needs to know which columns of the card are to be read, it consults the _____.

●●●

A. FORMAT Statement

78 Every FORMAT statement is referred to by some input or output statement; consequently, every FORMAT statement has a statement number.

Q. (Yes or No) If a FORMAT has the number 100, is the statement GO TO 100 legal? _____

●●●

A. No (A FORMAT statement can not be executed; therefore you cannot send the computer to that statement)

79 The use of the FORMAT statement is basically simple. Suppose you wish to read six real numbers per card, reading several cards to fill an array. The FORMAT statement is only concerned with the desired number of values per card and what card columns will contain those numbers. A typical FORMAT which might be used to satisfy the above conditions is FORMAT(6F12.4). The codes are explained later.

Q. (True or False) The FORMAT statement directly defines the number of cards to be read.

●●●

A. False (the FORMAT only specifies how many numbers per card and where they are located)

80 Naturally, if you wished to have ten numbers punched per card you would construct the FORMAT statement accordingly. The same is true for any desired card layout you might wish to specify.

Q. If you had a FORMAT specifying 6 numbers per card and an input statement with a list of 12 variables using that FORMAT, how many cards do you think would be read by the computer? _____

●●●

A. 2

81 The FORMAT statement is constructed in the following manner: a statement number, the word FORMAT, and a pair of parentheses containing the information about the card layout. The contents of these parentheses are explained in the next few frames.

Q. (True or False) The statement 1 FORMAT (card information) is legal as far as shown.

●●●

A. True

82 The card layout information contained in the FORMAT statement consists of "format codes" which tell the computer how many card columns per number, how many decimal places in the number, whether it is integer or real mode, etc. You should distinguish between the terms "number" and "card column". "Number" refers to a complete value, but a "card column" can contain only a single digit.

83 These format codes used in FORMAT statements are combinations of alphabetic and numeric characters. The alphabetic characters indicate the desired mode of the number, and the numeric characters tell which card columns are used, etc.

Q. (True or False) The list of variables in an input (or output) statement defines the card (or card image) layout.

●●●

A. False (the FORMAT statement always defines the layout).

84 The format code for real numbers uses the letter "F" together with two integers separated by a period (novelty?) which define, respectively, the number of columns of the card used to contain the number and the position of the decimal point. Example:
1 FORMAT(F12.6)

Q. How many card columns does the FORMAT above specify? (1,6,12, or 18) _____.

●●●

A. 12

85 The example of the previous frame, 1 FORMAT(F12.6), indicates real conversion of a number contained in the first 12 columns of a card (or card image) with 6 decimal places.

Q. To indicate real conversion, the letter _____ would be used as a format code.

●●●

A. F

86c The first integer in the FORMAT conversion code specifies the number of card columns used to contain the number being read. This item is commonly called the field width. When a FORMAT specifies a field width for, say, a card, the number to be read must not exceed the bounds of that field.

Q. The field width specified by the statement 10 FORMAT(F15.2) is _____ columns.

●●●

A. 15

87 It should be noted that the second integer in the F-conversion code - the indicator of the decimal point position - is required in the FORMAT but will be ignored by the computer if an actual decimal point is punched in any position in the card being read. If no point is punched, the FORMAT information is used.

Q. The statement 5 FORMAT(F10.1) would specify (how many?) _____ decimal place(s) if no decimal point were punched on the card.

●●●

A. 1

88 Let's see how a card punched with data, would be described with a FORMAT statement. Turn to Panel 3.5. Here you see a card punched with an 8 digit number. Let's assume that the number has 4 decimal places. The FORMAT statement to represent this data would be 1 FORMAT(F8.4).

Q. Write a FORMAT statement for the number if it had only 1 decimal place.

●●●

A. 1 FORMAT(F8.1)

89 In the previous example, notice that the number occupies the left-most positions in the card. The FORMAT statement must always describe the card by starting at column 1 (the left-most position) and moving to the right.

Q. The FORMAT statement describes the contents of the card from _____ to _____.

●●●

A. left, right

90 Turn to Panel 3.6. Here you see a card punched with a number as in the previous example, except that now a decimal point is punched. Notice that the decimal point occupies one of the 8 card columns used for the number.

Q. If the statement 1 FORMAT(F8.4) were used to describe this card, how would the number be represented in the computer?

●●●

A. 0963.412 (the punched decimal point would override the decimal notation (.4) in the FORMAT statement)

91 Don't confuse the numbers read from cards with FORTRAN constants. The constant, you will remember, must have a decimal point if it is real; a number read by the computer with "F" conversion may have the decimal point omitted and the FORMAT will specify where it belongs.

Q. The statement combination READ(2,1)X; 1 FORMAT(F15.5) requires that the number whose value will be read for X be punched in the first _____ columns of the card.

●●●

A. 15 (NOTE: The semicolon separating the two statements in th question implies that these are two separate statements; this convention is used for the remainder of the text)

92 The first integer in the number-conversion code, which specifies the field width, indicates the total column spread occupied by the number. This includes any decimal places specified by the second integer and the column occupied by the decimal point, if punched.

Q. Regardless of the value of the second control integer in a FORMAT conversion code, the total field width depends on the first integer, as in the statement FORMAT(F16.8) which defines a field of _____ columns.

●●●

A. 16

93 The FORMAT statement contains in parentheses, then, a number conversion code. This consists of three items of information in its real form: the mode (indicated by the letter F), the field width (defined by an integer after the F), and the decimal point position (denoted by a second integer, separated from the first with a period).

Q. Write a FORMAT statement to specify converting a real number from the first 8 card columns with three decimal places: _____.

●●●

A. 1 FORMAT(F8.3)

94 If your answer agrees exactly with the one shown, you are correct and may skip to frame 103. If you did not arrive at the correct answer, go to the next frame.

95 Do you remember the definition of the number-conversion code? It consists of an alphabetic character to denote the mode, an integer to specify the field width, a period, and a second integer to define decimal point position in case no decimal point is punched in the field.

Q. (True or False) The statement 1 FORMAT(F12.6) fits the above description.

●●●

A. True

96 For real number conversions, the alphabetic code in the FORMAT is "F". The two integers used with the F are selected to fit a particular problem. In the preceding diagnostic problem you were asked to specify 8 columns of field width and 3 decimal places. The complete conversion code for this problem has to be (F8.3).

Q. (Yes or No) If you punched a number, including its decimal point, on the card to be read by the conversion code shown above, would that integer 3 have any effect? _____.

●●●

A. No (this information is only applied when no decimal point is punched in the number being read)

97 Q. Let's try another case. Construct a FORMAT statement to specify the conversion of a real number with 7 decimal places contained in the first 15 card columns._____

●●●

A. 1 FORMAT(F15.7)

98 If your answer agrees with the one shown above, go on to the next frame. If you still cannot construct a FORMAT statement with the information given, go back to frame 69 and review some more.

99 Perform Exercise 3.6 in your problem book.

If the statement 1 FORMAT(F12.6) were used with the statement READ(2,1)X the computer would convert the contents of the first 12 columns of a card into a real number and place that value in the variable X.

Q. The same FORMAT statement and the statement READ(2,1)Y would define the value of the variable _____.

●●●

A. Y

100 The FORMAT's parentheses contain the complete specification for a card. If there is one number-conversion code in a FORMAT this means there will be only one number read per card.

Q. How many numeric values per card are indicated by the statement 1 FORMAT(F10.4,F12.6)?

●●●

A. 2

101 Naturally, then, a FORMAT statement is permitted to have more than one number-conversion code in a single statement. In fact, there is no set limit on the number of conversions permitted in a single FORMAT as long as you stay within the bounds of the card size (80 columns).

Q. The statement 1 FORMAT(F12.4,F12.6,F6.2) has a total of _____ conversion codes.

●●●

A. three

102 The contents of a FORMAT statement define the layout of a single card. The total column usage of the card is the sum of the field widths of all the number-conversions in the FORMAT. As an example, the statement 1 FORMAT(F12.4,F20.6,F10.5) defines the layout of a card: the first conversion code specifies the contents of columns 1 to 12; the second code specifies columns 13-32; and so forth.

Q. The total number of card columns specified in the FORMAT statement above is _____.

●●●

A. 42

103 So, if you use more than one number-conversion code in a single FORMAT, keep in mind that you are specifying the layout of a card, and be careful that the sum of all the field widths which you reserve does not exceed the allowable size of a card. Note: Naturally, adjacent conversion codes are separated by commas in all FORMAT statements.

Q. (True or False) The statement 1 FORMAT(F40.5,F40.5) would be suitable for reading 80 columns of data.

●●●

A. True (total field width specified is 80 columns)

104 In review, the FORMAT statement serves the various input and output statements as a reference or map to specify the condition of the numbers in the external form (how punched on cards, how arranged on paper tape, etc.). The FORMAT consists of a series of one or more number-conversion codes, each of which specifies the mode, length, and decimal point position of a single number. The entire series of codes constitutes the layout of an entire card.

105 Perform Exercise 3.7 in your problem book.

Recent frames have illustrated FORMAT statements which contain more than one number-conversion code, indicating more than one number is to be read from a card.

Q. The statement FORMAT(F12.4,F10.5,F6.2) denotes (how many?) _____ numbers per card.

●●●

A. three

106 You can also make use of an integer before the letter F in the FORMAT which will indicate that the number-conversion code is to be repeated that number of times in the card layout. For example, the statement FORMAT(3F12.4) is exactly equivalent to FORMAT (F12.4,F12.4,F12.4).

Q. The statement FORMAT (3F12.4,3F12.6) specifies (how many?) _____ numbers per card.

●●●

A. six

107 This count integer placed before the number-conversion code makes the writing of some FORMAT statements considerably easier. If you wished to read twelve numbers per card, six columns per number with two decimal places, the statement FORMAT (12F6.2) is obviously easier to write than twelve separate codes.

Q. The statement FORMAT (12F6.2) specifies (how many?) _____ card columns.

●●●

A. 72 (12 x 6)

108 A complete input or output operation, then, is defined by an Input or Output statement with its list of variables indicating the items involved, working together with a FORMAT which specifies the way in which these items are laid out in the cards.

Q. (True or False) The statements READ(2,1)A,X,H ; 1 FORMAT(3F12.6) will read from one card three values from successive twelve-column fields, and place these values in A, X, and H.

●●●

A. True

109 The important points to remember are that the input statement list determines how much data is read and that the FORMAT statement specifies how many values are contained in a particular card. The combination of these items of information determines how many cards are to be read.

Q. The statement READ(4,1) (ARRAY(I),I=1,100) indicates that (how many?) _____ numbers are to be read from the paper tape reader.

●●●

A. 100

110 Notice that in the last question 100 values are to be read from paper tape. This many numbers obviously could not be placed on a single card. The FORMAT statement used with this input statement will specify the description of one number, and will use the repetition factor to determine how many numbers will be read. For example, FORMAT(100F6.3) will cause the reading of one hundred 6-position numbers.

Q. Write the FORMAT statement needed to read from paper tape thirty-six 7-position numbers with 4 decimal places.

●●●

A. FORMAT(36F7.4)

111 If the list of variables in an input statement is so long that it requires more than one card to contain those values, the FORMAT specifications keep repeating for each new card that is read. The computer will keep on reading additional cards until enough values have been read to satisfy the list.

Q. The statements READ(2,1) (A(I),I=1,1000) ; 1 FORMAT(10F7.3) will read (how many?) _____ cards.

●●●

A. 100 (We need 1000 values, and we get 10 per card.)

112 The relationship of Input and FORMAT statements should be getting clearer now. It is very important that their respective functions be understood; otherwise, you may accidentally tell the computer to do something entirely different from that which you intended.

113 Perform Exercise 3.8 in your problem book.

If a FORMAT statement specifies less than a full card's contents, the unspecified columns are not read by the computer. For example, the statement 1 FORMAT(5F14.5) defines the contents of 70 columns of a card; if anything were punched on the rest of the card, it would be ignored.

Q. FORMAT(12F5.2) ignores all card columns beyond column number _____.

●●●

A. 60

114 If an Input statement list contains fewer quantities than the number-conversion codes in the FORMAT, the reading process still stops when the list is satisfied. For example, the combination READ (2,1)X,Y,Z ; 1 FORMAT(6F12.4) will read only three quantities from the card, even though six are specified. (Assume 2 is the input reference number for the card reader. From here on, 2 will mean the card reader, unless otherwise specified.)

Q. Then the FORMAT shown above and the statement READ (2,1)A,B will read through column number _____ of the card.

●●●

A. 24

115 Sometimes the number of quantities in an Input statement list is not an even multiple of the number of values per card in the associated FORMAT. For example, READ(2,1)(A(I),I=1,10) ; 1 FORMAT(6F12.4). In this case, six numbers are read from the first card and four from the second, and reading stops when the list is satisfied, as usual.

Q. (True or False) Any numbers punched in columns 49 to 80 of the second card in the above example are ignored.

●●●

A. True (columns 73 to 80 of the first card will also be ignored)

116 Q. Write a READ and FORMAT combination that will read a set of monetary values into the DOLLR array. The program has already defined N as the number of values to be read. The largest dollar value is less than 1,000.00 and you wish to punch (including a decimal point) as many numbers per card, evenly spaced, as you can.

```
      READ
1     FORMAT
```

●●●

A.
```
      READ(2,1)(DOLLR(I),I=1,N)
1     FORMAT(13F6.2)
```

117 If your answer agrees with the one shown (except, perhaps, for choice of index variable which is arbitrary), you may skip to frame 122. Otherwise, continue to the next frame.

118 The READ statement should have been no problem unless you're getting a little rusty: N values to be read into the DOLLR array suggests a self-indexed list, such as (DOLLR(I),I=1,N). Every comma and parenthesis shown here is required for all self-indexed lists.

Q. The same indexed list, using J as the index variable would be _____.

•••

A. (DOLLR(J),J=1,N) (the J must appear as the index and the dummy subscript)

119 Six card columns are needed for each number read: three whole-number digits (dollars), a decimal point, and two decimal places (cents). To pack these values most efficiently on cards, the statement FORMAT(13F6.2) specifies the use of 78 columns of the card. (this still wastes 2 columns, (79,80) but this cannot be avoided.)

Q. If you were permitted to use only 72 columns of a standard card, the most efficent FORMAT statement would be
1 FORMAT(_____)for this problem.

•••

A. FORMAT(12F6.2)

120 Q. Try another question about this same problem: given the statement combination

```
      READ(2,1)(DOLLR(I),I=1,N)
1     FORMAT(12F6.2)
```

and a value of 40 for N, the computer would read (how many?) _____ cards, including one incompletely filled card.

•••

A. four (three cards with 12 numbers each, plus one card with 4 numbers)

121 If you could answer this question correctly, you are getting the knack of these Input statement problems. If you are still having trouble, return to frame 99 for some review; otherwise, go on to the next frame.

122 Perform Exercise 3.9 in your problem book.

This chapter has, thus far, concerned itself only with real number conversion in FORMAT statements using the "F" format code. You may recall from Chapter 1 that real numbers can also be represented by using the E exponent form. The following frames will discuss the FORMAT statements needed to convert numbers written in this form.

123 E notation, as you will recall from Chapter 1, is used as a shorthand version for writing numbers. For instance, 9.3E04 represents the number 93000 to the computer.

Q. What number does -4.6219E-2 represent?

●●●

A. -.046219

124 Numbers read into a computer may be coded in E notation. That is, they may be represented as a number times a power of ten. This form is commonly used to reduce the amount of keypunching necessary when reading in very large or very small numbers from cards. For instance, the number 2100000 may be punched as 2.1E6 in a card.

Q. How many card columns are saved by using E notation in the above example?

●●●

A. 2 (2100000 would take 7 columns, while 2.1E6 takes only 5)

125 Note that in the preceding example a card column is required for the decimal point as well as the "E". Turn to Panel 3.7. Here you see a card as it would be punched in columns 1 to 5 with the number 2.1E6. In columns 6 to 11, we have punched the same number expressed as a negative quantity.

Q. If a number in E notation represents a negative quantity, one additional column must be reserved for a _____.

●●●

A. minus sign

126 You may recall that there is actually a good deal of variation allowed in writing E notation. For example, the exponent portion of a number may take any of the following forms to represent the exponent "plus 6": E6, E+6, +6, +06, E 06, E06, and E+06. Notice that the sign of the exponent is optional if the exponent is a positive quantity.

Q. Could the sign of the exponent be omitted if the exponent were negative?

●●●

A. No. If the minus sign were omitted the computer would assume that the exponent is positive.

127 In the preceding frame, a form of E notation was shown which does not use the letter "E". (+6, +06) This form of notation is permissible only when used with input data, and must not be used within program statements because the computer would think you were trying to add or subtract a legitimate constant.

Q. Could the "E" be removed from the statement Y=A*B+4.321E+7 ?

●●●

A. No. To do so would change the value of the statement.

128 Turn to Panel 3.8 and study it carefully. Here you see the number 1234000 punched in a card using different, but equivalent, forms of E notation. If this card were to be read by the computer, we would need to write an appropriate READ and FORMAT statement.

Q. Given an appropriate FORMAT statement, would the statement READ (2,1) A,B,C,D,E,F,G,H,P be an acceptable READ statement?

●●●

A. Yes

129 When input numbers are written in E notation, a FORMAT statement containing the number conversion code "E" is used. The letter "E" specifies that a number written in E notation is to be converted to a real number for use by the computer.

Q. When writing FORMAT statements which refer to numbers with E notation, the letter used in the FORMAT statement is _____.

●●●

A. E

130 The construction of the FORMAT statement code used for E conversion is identical to that used with F conversion. That is, we have the letter "E", then an integer number for the total field width, a decimal point, and finally an integer number to indicate the number of decimal positions.

Q. Is 1 FORMAT(E6.3) a legal statement?

●●●

A. Yes

131 Let's examine the elements within the parentheses of the statement 1 FORMAT(E6.3). We'll assume that the statement is to be used with an appropriate READ statement for reading cards. The "E" means that a number written in E notation is to be converted to a real number. The "6" means that the total number of card columns (field width) punched with the number to be converted is 6. The ".3" means that the number from the card contains three decimal places, prior to whatever effect the E exponent will have on the decimal point.

Q. In the statement 1 FORMAT(E9.6), the field width is _____ card columns, and the number contains _____ decimal places.

●●●

A. 9, 6

Q. Suppose the number 9629E+03 were punched in a card. What would the field width be?

●●●

A. 8

132 Let's take a look at an example which uses an E FORMAT statement. Suppose that the number 1352E03 were punched in a card. This E notation actually represents the number 1352000., but let's assume that we really want to use the number 135.2 in the computer. By writing the statement 1 FORMAT (E7.4), we would achieve this effect. Because we have specified 4 decimal places in the FORMAT statement, the incoming number is assumed to be .1352E03. The exponent "E03" then will convert the number to 135.2

Q. What number would result if the input number is 81296E+2 and the FORMAT statement is 1 FORMAT(E8.5) ?

●●●

A. 81.296 or 81.29600

133 In the two preceding frames, the E numbers punched in cards have not contained decimal points. If the punched number contains a decimal point, the decimal portion of the E FORMAT statement is overridden, just as it was in F conversion.

Q. (True or False) The punched decimal point in an E number has no effect on the FORMAT statement decimal code.

●●●

A. False

134 If the number 12345.6E-3 were punched in a card, and the FORMAT statement conversion code was E10.3, the actual value of the converted number would be 12.3456 Make sure you understand how this happened. Just keep in mind that the FORMAT decimal specification is disregarded if there is a punched decimal point in the number to be converted.

Q. With the specification E10.5, determine the numbers to which the following punched numbers will be converted.

a. +56789.0E2 = _____
b. -5678934E5 = _____
c. 5678.900E0 = _____
d. 567891E-01 = _____
e. +567.893+3 = _____
f. +56.789E-3 = _____

●●●

A. a. 5678900.
b. -5678934.
c. 5678.900
d. .567891
e. 567893.
f. .056789

135 If your answers to the preceding question were correct, you have a good understanding of E conversion. If you had trouble, however, you should go back to frame 122 and review E conversion, particularly noting frames 131 and 132.

136 You have seen in the preceding frames that the E conversion code in the FORMAT statement is constructed just like the F conversion code. The similarity does not end there, however, because all the FORMAT construction options using F also apply to E. For instance, it is perfectly legitimate to have the statement 1 FORMAT (3E10.3) control the conversion of 3 ten-position E numbers in a card.

Q. How many numbers in a card would be converted with the following statement? 1 FORMAT(3E6.2,E7.1,2E6.3)

●●●

A. 6

137 Incidentally, E and F conversion codes may be mixed freely in a FORMAT statement. Thus, the statement 1 FORMAT (F6.2, 2E9.6,4F5.1,E9.4) would convert one six-position number punched without E notation, two 9-position numbers with E notation, four 5-position numbers without, and one 9-position number with E notation.

Q. Write an input FORMAT statement to convert the following sequence of numbers to real numbers:

1. four-position number, two decimals, no E notation
2. 5 six-position numbers, one decimal, no E notation
3. 3 eight-position numbers, no decimals, with E notation
4. a ten-position number, two decimals, with E notation

●●●

A. 1 FORMAT (F4.2, 5F6.1, 3E8.0, E10.2)

138 So far, we have explained the E conversion code only as it applies to input data. Conversion codes apply equally as well to output data, but sometimes they have slightly different effects. The explanation of the E code as it relates to output data will be given later in this chapter when output statements are discussed.

Q. (True or False) Conversion codes such as E and F apply only to input data, because only input data needs to be converted.

●●●

A. False (output data may also be converted)

139 Let's recap the discussion of the E code. You have seen that it is used to convert data written in E notation to real numbers within the computer. The E FORMAT statement has the same construction as the F FORMAT statement, and may, in fact, have F conversion codes intermixed with the E codes. An E code may also have an integer which indicates a repetition of the E code, just as is done with F codes.

Q. (True or False) The F code causes conversion to real numbers, and the E code causes conversion to E numbers.

●●●

A. False (this description of the E code is nonsense - we start with an E number, not end with it)

140 The E and F conversion codes are used to convert data to real numbers. As you know, we can also use integer numbers in FORTRAN, and in the next few frames you will learn the conversion code necessary to convert integer numbers.

141 To read numbers in the integer mode, the letter "I" is used in the FORMAT statement. This letter corresponds to the "F" in real conversion codes, serving only to denote the mode of the numbers being read.

Q. The statement FORMAT(I6) denotes conversion in the _____ mode.

●●●

A. integer

142 The letter "I" in an integer number-conversion code is followed by an integer, just as is the case in real codes, which indicates the field width on the card for the number being read. This is all the information required for integer numbers, since there is no decimal fraction part.

Q. The statement FORMAT(I20) indicates the conversion of an integer number contained in columns 1 through _____ on the card.

●●●

A. 20

143 Integer numbers are relatively uncomplicated; that is, they are simply whole number quantities and they are punched in cards without decimal points. The only information needed by the computer to read an integer number is the card columns in which the number is contained.

Q. To read a five-digit integer from a card, the statement FORMAT (_____)would be used.

●●●

A. I5

144 Since the variable list in an Input statement may have quantities of either mode, it follows that a FORMAT statement will permit number-conversion codes of either mode in the same statement. For example, the statement FORMAT(I5,F15.4) will read two numbers, the first in integer mode, the second, in real mode.

Q. The statement FORMAT(I5,F15.4) specifies the two conversions from the first _____ columns of the card.

●●●

A. 20

145 To illustrate the mixture of modes in Input statements, consider the sequence READ(2,1)N,X and 1 FORMAT(I20,F20.6) which will convert the number punched in columns 1 to 20 into an integer, placing the value in N, and the number punched in 21 to 40 into a real number, placing the value in X.

Q. (True or False) The use of the FORMAT statement number 1 above by the statement READ(2,1)N,M would be proper.

●●●

A. False (the quantity M in the READ list is an integer; the conversion code is real)

146 This last question raises an important point: the actual mode of conversion of the number from the card form to the computer form is specified by the FORMAT statement, with no regard to the mode of the variable which will have that value when reading is complete. This means that an error condition can exist if you are careless.

Q. The execution of READ(2,1)N; 1 FORMAT(F8.4) will cause a _____ number to be assigned to the variable N.

●●●

A. real

147 Unfortunately, this doesn't work like the Arithmetic statement N=X which would convert the mode appropriately before assigning a value to the variable. If a READ statement list variable is of the wrong mode for the conversion code in the associated FORMAT, a number actually in the wrong mode is assigned.

Q. The sequence READ(2,1)A,B ; 1 FORMAT(I5,F12.4) would result in the incorrect conversion of the variable _____.

●●●

A. A

148 Since the real and integer numbers are different inside the computer, the use of the statement sequence of the previous example would result in utter havoc when the variable A was used later in the program. There is literally no telling what would happen, and at best, the results would be meaningless.

Q. (True or False) To prevent variables in the wrong mode, the variable list and the FORMAT statement should correspond in the mode used for each conversion.

●●●

A. True (this is a common error; you should check this part of your programs for hard-to-find bugs)

149 You now know how to combine statements to read groups of numbers of either mode from either cards or paper tape. The following items should be understood completely before continuing: list construction (in general, the items to be read); self-indexed lists (the form that permits reading several values through index definition); "F", "E", and "I" conversion codes (the coded information describing the external form of the numbers); and the list-FORMAT relationship (that determines how many cards, etc. are read).

150 Perform Exercise 3.10 in your problem book.

Before going on to more details of the FORMAT statements and the output statements, it would be appropriate to show you how the input data are punched on cards. The following frame shows a picture of a typical card which might be read with the FORMAT(3I5,2F20.6,E7.5).

151 Turn to Panel 3.9. When used with a statement combination such as READ(2,1)I,J,K,A,B,C and 1 FORMAT(3I5,2F20.6,E7.5) this card, when read, would result in the following values for the variables: I would have the value 100; J, the value 50; K the value 1000; A, the value 3.141592; B, the value 3.141592; and C, the value 4600.

Q. The value read for B in the example has its decimal point positioned _____ places from the right by the FORMAT statement conversion code.

●●●

A. 6 (since no decimal point is punched, the FORMAT integer will place one in the indicated position)

152 Notice that all quantities are punched in the rightmost portion of their allotted fields. 1130 FORTRAN does not allow a number to contain any blanks, either within the number itself or to the right of the number. (An error halt will occur if a number contains a blank.) Blanks to the left of a number, however, are permissible.

Q. (True or False) If a card contained the number 5 punched in column 1, a correct FORMAT statement would be 1 FORMAT (I3).

●●●

A. False. This FORMAT statement implies a number field of three positions, which, in this case, would cause the number 5bb to be read. This number is not acceptable to 1130 FORTRAN because of the blanks.

153 The FORMAT statement in the preceding question could be corrected by changing it to 1 FORMAT (I1). Another possibility for correcting the problem would be to leave the FORMAT statement as is, and repunch the data card with the 5 in column 3. Thus, the number would be read correctly as bb5. (This number has the value 5).

Q. Write a FORMAT statement to read a card that contains the number b45.923 in the first 7 card columns.

•••

A. 1 FORMAT (F7.3)

154 Because of the possibility of mispunching the data on the cards, it is strongly advised that you punch the decimal point in all real numbers read from cards. This leaves no doubt as to the placement of the point.

Q. If the statement FORMAT(F5.2) were used for reading a card punched .1234 in the first five columns, what value would be read into the corresponding variable? _____

•••

A. .1234 (as punched; the presence of the decimal point overrides the FORMAT specification)

155 Although the preceding examples did not show this, any of the numbers punched on cards for reading by the computer may have plus or minus signs preceding the punched digits. If no sign is punched, it is assumed to be positive. Make sure that the field width specified by the FORMAT is large enough to include the sign, if used.

156 Perform Exercise 3.11 in your problem book.

The basic FORMAT statement as described in the last several frames defines the layout of a single card; when the list in the statement using the FORMAT requires more data than can be contained in one card, the same layout is repeated card after card until the list is satisfied. As shown in the next few frames, FORMAT statements may specify more than a single card layout in a single statement.

157 The character "/" (slash) is used in FORMAT statements to separate groups of conversion codes. These groups then specify separate card layouts. For example, the statement FORMAT(6F12.4/12F6.4) defines two different card layouts; one is (6F12.4), the other is (12F6.4).

Q. The statement FORMAT(I5/6F12.4/12F6.4) contains (how many?) _____ different card layouts.

●●●

A. three

158 The statement FORMAT(6F12.4/12F6.4) merely tells the computer that the first card read with this FORMAT will have six real numbers, twelve columns each, etc., and the second card read will have twelve numbers, also real, six column fields, etc.

Q. The FORMAT statement shown above specifies conversion codes for a total of _____ numbers.

●●●

A. eighteen

159 A statement such as FORMAT(6F12.4/12F6.4) is used in cases where the cards to be read are not laid out in the same manner on successive cards. If the list of an Input statement using this FORMAT contained more than eighteen variables, the specifications would repeat at the beginning of this FORMAT (first card layout).

Q. The third card read (if any) with the FORMAT shown above would have to contain (how many?) _____ numbers.

●●●

A. six (in accordance with the first card layout)

160 Remember, all the basic FORMAT statements of recent examples would repeat their specifications with each new card that was read when the Input list was long enough to require more than one card. The same condition exists in FORMAT statements defining more than one card layout.

Q. (True or False) If 100 cards were read using FORMAT (6F12.4/12F6.4) 50 of the cards would have to contain six numbers each and the other 50 would have to contain twelve numbers each.

●●●

A. True

161 Consider an example: READ(2,1)(A(I),I=1,1800); 1 FORMAT (6F12.3/12F6.4) would cause the reading of 200 cards: the first card would have six numbers; the second, twelve; the third, six; the fourth, twelve; the fifth, six; etc. The FORMAT would always repeat its specifications at the beginning of the statement, alternating the card layouts.

Q. The contents of the third card read in the above example would become the values of positions (number) _____ through _____ of the A array.

●●●

A. 19, 24

162 If you correctly answered the previous questions, you are getting the knack of list-FORMAT interaction very well. As long as the list keeps calling for more input, the FORMAT keeps specifying the layouts of the cards repeating itself as many times as necessary.

Q. The statements READ(2,1)X; 1 FORMAT(6F12.4/12F6.4) would cause the computer to read (how many?) _____ card(s) (careful now!).

●●●

A. one (the definition still holds that reading stops when the Input list is satisfied)

163 A particularly desirable use of the "/" in FORMAT statements provides for the reading of a single card according to the first card layout in the FORMAT, and then reading several cards according to the second card layout, without repeating the FORMAT specifications back at the beginning. The explanation follows.

164 The statement FORMAT(I6/(6F12.5)) would cause the computer to read a card and convert a single number in the integer mode, then read a second card and convert six real numbers; assuming that the list is long enough, a third card would be read, still reading six real numbers. The first card layout in this FORMAT is used only on the first card read! Why? Read on ...

165 Notice anything unusual about the statement FORMAT (I6/(6F12.5))? There is an extra pair of parentheses surrounding the second card-layout specification. These parentheses cause the FORMAT specifications to repeat only the second card-layout if the Input list requires more than two cards to be read.

Q. How many number(s) would be read from the first card using the FORMAT shown above?

●●●

A. one

166 The general rule about repeating FORMAT specifications for long Input lists is: when the end of the FORMAT statement is encountered, having used all its specifications once through, and the Input list is not yet satisfied, the reading will continue, repeating the specifications at the last open-parentheses in the FORMAT statement.

Q. The statement FORMAT(I5/2E6.1/(12F6.4)) would require a total of _____ numbers on the fourth card read with this FORMAT.

●●●

A. twelve (assuming, of course, that the list were long enough)

167 An example of this useful feature might be READ(2,1)N,(A(I),I=1,N) ; 1 FORMAT(I5/(6F12.4)) in which the integer read from the first card becomes the index limit for the self-indexed list. Thus the first card read determines how many successive cards will be read, repeating the (6F12.4) specification.

Q. If the value read above for N on the first card were 600, a total (be alert) of _____ cards would be read.

●●●

A. 101 (100 cards containing the 600 numbers, six per card, plus the first card with the value of N)

168 This last example demonstrates the most useful of the "fancy" features of FORMAT statements. You will probably find that the use of the "/" seldom occurs in input problems except for the variable length array as in that example.

Q. (Yes or No) Assuming that N is greater than 5, is anything wrong with the combination READ(2,1)N,(A(I),I=1,N) ; 1 FORMAT(I5,5F12.4)?

●●●

A. Yes (the FORMAT will repeat from the beginning, resulting in incorrect conversion of some numbers)

169 The use of parentheses in the fashion just described is limited to one level; that is, parentheses within parentheses are not permitted in FORMAT statements. When used properly, as in FORMAT(I5/(6F12.4)), a single card with a unique layout is read, followed by the reading of several cards with the second layout.

Q. (True or False) Columns 6 to 80 of the first card would be ignored if read with the FORMAT shown above.

●●●

A. True

170 Parentheses may be used in another way in FORMAT statements where a particular sequence of conversion codes is to be repeated in the same card layout, as in: FORMAT(2(I12,2F12.4)). The conversion sequence (I12,2F12.4) comprising 36 columns will be repeated according to the integer preceding the parentheses.

Q. (True or False) The FORMAT shown above appears to be the same as FORMAT(I12,2F12.4,I12,2F12.4)

●●●

A. True

171 Any group of conversion codes which has a repeating pattern may be enclosed in parentheses preceded by an integer, indicating the number of times that pattern is to be repeated. The total number of card columns so allotted must not exceed the size permitted for a single card, however.

Q. The statement FORMAT(I12,E12.4,I12,E12.4,I12,E12.4) could be re-written with parentheses as:
FORMAT(_____).

●●●

A. 3(I12,E12.4)

172 The last several frames have gone through an awful lot of details about FORMAT statements. It would be wise at this point to tally the score while you catch your breath. If you are not interested in the review on the next seven frames, you may skip ahead to frame 180.

173 Basically, FORMAT statements supply the computer with information about the expected layout of data on cards or card images being read, regardless of whether cards, tape or disk are being read. To read a list of any length, six numbers per card, 12 columns per number, with 4 decimal places, FORMAT (6F12.4) may be used.

Q. Write a statement to specify reading of twelve integer numbers per card, six columns each.

●●●

A. 1 FORMAT(12I6)

174 If a statement such as READ(2,1)(K(J),J=1,36) were used with the preceding FORMAT, a total of three cards would be read, each one with the layout (12I6). The FORMAT specifications will repeat where the list in the Input statement requires the contents of more than one card.

Q. If the statement READ(2,1)(NUM(M), M=1,40) were used with the preceding FORMAT, (how many?) _____ cards would be read.

●●●

A. four (the fourth card would not be completely read)

175 Using the "/" to separate conversion code groups, a single FORMAT statement can specify more than one card layout for successive cards. If no interior parentheses are present in such a FORMAT, the specifications will repeat completely from the beginning if the Input list is long enough.

Q. The combination READ(2,1)(A(K),K=1,100) ; 1 FORMAT (6F12.2/6F12.4) will result in (how many?) _____ cards being read with the conversion code 6F12.4.

●●●

A. eight (seventeen cards in all; nine cards with the first card layout, and eight with the second.)

176 A multiple-card FORMAT statement such as the one described in the preceding frame can have a set of interior parentheses which enclose a complete card layout. When the Input list is long enough to require the FORMAT to repeat, the repetition begins at the last open parenthesis in the FORMAT.

Q. If 1001 numbers were read with the statement FORMAT(I5/(6F12.4)), (how many?) _____ numbers would be converted according to the specification (6F12.4).

●●●

A. 1000

177 Still another use of parentheses within FORMAT statements is to set off groups of conversion codes which make up repeating patterns within a single card layout. For example, FORMAT (3(I5,F10.5)) is the same effective FORMAT statement as FORMAT(I5,F10.5,I5,F10.5,I5,F10.5).

Q. Write out the equivalent FORMAT (without interior parentheses) for the statement FORMAT(4(I5,I12).

●●●

A. FORMAT(I5,I12,I5,I12,I5,I12,I5,I12)

178 Q. Write a suitable READ and FORMAT combination to enable the computer to:

1. Read an integer (ten column field) and five real numbers in E notation on a card, then
2. Read a group of cards, each containing ten real numbers (no E notation)
   a. all of the real numbers are to be eight columns in length with four decimal places
   b. the total number of real numbers is determined by the first integer read

Choose appropriate identifiers for the integer variable, the array, the FORMAT number, and the input reference number.

READ	
1	FORMAT

●●●

A.
```
      READ(2,1)N,(A(I),I=1,N)
    1 FORMAT(I10,5E8.4/(10F8.4))
```

179 Your answer, short of having different variable names, should be pretty close to this one. If you note any discrepancies, check the problem statement and the suggested solution above and make sure you understand all the principles applied here.

180 Perform Exercise 3.12 in your problem book.

You have learned how to make the computer read from cards and paper tape, but another very important part of a program is the part which gets the computer to exhibit its answers: the Output statements. The following frames will cover this group of statements.

181 On the 1130, the most common form of output will be the printed page (printed either by a typewriter or a printer), punched cards, or punched paper tape. The 1130 FORTRAN language contains output statements for all of these devices.

182 To have the computer write with an attached printing device, the WRITE statement is used. This consists of the word "WRITE" followed by a set of parentheses enclosing both an output reference number and a FORMAT statement number, and finally, a list of variables, subject to the same set of rules as input lists. Sounds familiar, doesn't it? Actually, all we have done is to exchange the word "WRITE" for the word "READ", thus converting an input statement to an output statement.

183 Our discussion of output statements will concentrate mainly on using the printer, since this is where results are ultimately displayed. If your 1130 system has a typewriter instead of a printer, all of the following concepts are just as valid to you, with the only difference being the choice of the output reference number. You will recall that the number "3" is our output number to reference the printer, while "1" is the logical unit number for the typewriter. Also, we will continue using statement number 1 as our FORMAT statement reference unless otherwise indicated.

Q. Write the output statements which will print the variable X, first on the printer, and then on the typewriter.

●●●

A.
```
WRITE (3,1) X
WRITE (1,1) X
```

184 Even though we will be referring to the printer as our output device, keep in mind that we can select different output media such as card punches, or paper tape punches, by assigning our output reference number to these units. In this case, the data coming out of the computer would be punched instead of printed.

Q. (True or False) The statement WRITE(4,1)A,ALPHA,ZED could transmit data to a paper tape punch.

●●●

A. True

185 The WRITE statement for printing operates as follows: the values of the variables in the list are output in the order listed and printed on the attached printing device, according to the indicated FORMAT statement. Like the Input statements, the list of variables serves only to supply the computer with the quantities involved.

Q. The statement WRITE(3,1)(A(I),I=1,100) will print (how many?) _____ values of the A array.

●●●

A. 100

186 The variable list in the WRITE statement performs the usual function. It defines the variables whose values are to be printed (transmitted). The list may contain variables of either mode and may make use of the self-indexed form; the variables may have subscripts as long as they are not mathematical expressions.

Q. The statement WRITE(3,1)A,B,C,I,J,K will cause the computer to print (how many?) _____ values.

●●●

A. 6

187 The FORMAT statement is used in WRITE statements in much the same way as in Input statements. It defines the layout of the numbers in the external medium (the printed line or the punched card layout) and it uses the same conversion codes as the Input FORMAT statements.

Q. (True or False) The FORMAT statements used with WRITE statements can use I, E, and F conversions.

●●●

A. True

188 The operation of a WRITE and FORMAT combination is best illustrated by an example: WRITE(3,1)A,B,C and 1 FORMAT(F20.4) will cause the computer to print three lines, each line containing a single number within a space of 20 print columns with 4 decimal places to the right of the decimal point.

Q. If statement 1 in the above example had been 1 FORMAT (2F20.4) the computer would print (how many?) _____ lines.

●●●

A. two (two numbers on the first line and one number on the second)

189 The FORMAT statement used with an Output statement will define the layout of a single line, and the writing process continues until the list is satisfied, repeating line after line in the same layout, similar to the card reading process covered earlier.

Q. The combination WRITE(3,1)(A(K),K=1,100) ; 1 FORMAT(10F12.5) will print (how many?) _____ lines.

●●●

A. ten (100 numbers, 10 numbers per line)

190 The line printed by the printer attached to the 1130 computer contains 120 characters. This means that FORMAT statements for WRITE statements can specify a maximum of 120 print positions. This is also the size of the typewriter line.

Q. (True or False) The statement 1 FORMAT(20E20.5) is a legal FORMAT for printing.

●●●

A. False (the total field width specified is 400 columns: too long)

191 Incidentally, the decimal point is printed for real numbers, and its position is rigidly dictated by the second integer after the F or E in the conversion code. On input, you will remember, the decimal point was permitted to be anywhere as long as it was punched, or the FORMAT information was used if the decimal point was not punched.

Q. The statement FORMAT(12F10.5) will print a line of twelve numbers, each with _____ decimal places.

●●●

A. five

192 Remember, the computer does not know what digits are significant with respect to the computation being done. It is up to the programmer to analyze the printed information for accuracy as to the number of decimal places being printed. The computer will print exactly what you call for, even if some of the digits are not significant.

Q. (True or False) If you are computing with two-place accuracy, you would not use a statement like FORMAT(F20.5).

●●●

A. True (why print five decimal places when you are only sure of two or less in your results?)

193 Another point: the printed numbers always appear right-adjusted (as far to the right as possible) in their allotted fields. This means that you can space the numbers on the printed page by making the assigned field width accordingly wider than is needed to contain the printed number. The excess space in the field will be left blank, thus providing spacing.

Q. If you use FORMAT(F20.4) to print the value 123.4567 (including the point) you will have (how many?) _____ blanks to the left of the printed number.

●●●

A. 12

194 The use of WRITE statements is actually that simple. Once you have mastered Input statements, many of the same features and conventions carry over into the Output statement family. To illustrate, a statement combination such as WRITE(3,1)(A(I), I=1,1000) and 1 FORMAT(10F12.4) will print out 100 lines, each line containing 10 values from the A array.

Q. The first printed line in the example above will contain values from elements _____ through _____ of the A array.

●●●

A. 1, 10

195 The output FORMAT statements may include conversion codes of either or both modes in a single FORMAT. As was the case with input data, the FORMAT code determines the mode of conversion and, if the variable whose value is being printed is of the wrong mode, the printed result is meaningless (often a rather wild number).

Q. The combination WRITE(3,1)A ; 1 FORMAT(I5)will result in the value of A being printed in the _____ mode.

●●●

A. integer

196 The problem of matching the modes of the quantities being read or written and the modes of the corresponding conversion codes is brought up again to emphasize its importance. Many error conditions can be caught and diagnosed by the computer, but this is not one of them. Programs with these bugs often behave strangely.

Q. (True or False) The combination
```
WRITE(3,1)A,B,C,I,J
1 FORMAT(3E20.6,2I10)
```
is perfectly valid, because the modes of the variables and conversion codes agree.

●●●

A. True

197 You can easily check your programs to ensure that this error condition does not exist. Any input or output list has a definite order in which the variables' value are to be input or output, and every FORMAT statement has a definite cycle order of conversion codes, even if repetition is called for. By taking a dry run, and reading the program like the computer would, you will often find situations where such errors exist.

Q. (True or False) The combination
```
WRITE(3,1)N,(A(I),I=1,1000)
1 FORMAT(I5,5F20.6)
```
will convert some of the data incorrectly.

●●●

A. True

198 The use of the WRITE statement, then, is very similar to the use of any of the Input statements. The general form of construction is the same: identifying word (WRITE), an output reference number, a FORMAT number, and a suitable list of variables whose values are to be printed. The FORMAT statement performs the same function for the WRITE statement as it does for Input statements. It specifies the layout of the data in the external medium.

199 Turn to Panel 3.10. This panel shows where the values are placed on the printed page as a result of the following statements.

```
      WRITE(3,15)(X(I),I=1,20)
15    FORMAT(4F10.4)
```

200 There are several items of interest on Panel 3.10. For one thing, you should notice that we have shown numbers, as they appear in storage, as well as their printed representation. Because the FORMAT statement specified four numbers per line, we see that we need five lines to print 20 numbers.

Q. How many lines would print if 1 FORMAT (2F10.4) had been used instead?

●●●

A. 10

201 The printed numbers on Panel 3.10 are particularly interesting. Let's take things one at a time, starting with the printing of X(1). Notice that two blanks are supplied by the computer to fill out the left-hand side of the number. This will always be done if the integer portion of the number is smaller than the space allotted to it.

Q. How would 2.123 print, using F6.3 as the FORMAT code? (Use b for blank)

●●●

A. b2.123

202 X(2) is printed with two zeroes supplied in the decimal portion of the number. Zeroes will always be supplied to fill out the unused decimal positions of the number being printed. Notice also that the decimal point is always printed, and it will always occupy 1 position of the print field.

Q. How would 3.14159 print, using F10.6 as the FORMAT code?

●●●

A. bb3.141590

203 X(3) is pretty straightforward, but be sure to note that if the number is negative, one printing space should be reserved for the minus sign. The plus sign is not printed for positive numbers.

Q. Write a format code to print the value 6.36, which may be either positive or negative.

●●●

A. F5.2

204 Nothing is new in X(4), but X(5) is worthy of note because we have lost one decimal position! This happened because the size of the decimal field was not large enough to accommodate our internal number. Notice that the printed number is not shifted left to compensate for this oversight. Excess decimal positions are simply dropped (truncated), and that's that. Care must be taken to avoid this common programming error.

Q. How would .16789 print, using F3.3 as the FORMAT code?

●●●

A. .16

205 X(6) is normal, but look what happens in X(7)! Here we have the unfortunate result of the integer portion of the number exceeding its allotted print space. When this occurs, the 1130 computer considers this situation to be an error condition, and will fill in the entire print field with asterisks. Incidentally, there must always be one space allowed for the sign, even though the number is positive. For example, 45.689 with the code F6.3 will print as ****** because there is no room allowed for the sign position.

Q. How would 98765.31 print, using F4.1 as the FORMAT code?

●●●

A. ****

206 X(8) illustrates that if there are no integer positions in a negative number, a minus sign will print to the left of the decimal point. A print position is taken, of course.

Q. How would -.3456 print, using F5.2 as the FORMAT code?

●●●

A. b-.34

207 X(9) illustrates the error condition caused by insufficient space allowed for a number and its sign. The moral is, always allow enough space!

Q. Careful now. Write a FORMAT statement which will print on one line three 7-digit signed numbers with 4 decimal places. Allow 8 blank positions between each of the printed numbers.

●●●

A. 1 FORMAT(3F17.4) The 17 positions would contain 8 blanks, a sign position, 3 integer positions, a decimal point, and 4 decimal positions. A total of 51 print positions would be used for all 3 numbers.

208 Q. Write a combination of statements which will print, using 120 print positions, the entire contents of the 600-number BLOCK array, six numbers per line, and two decimal places for each number.

●●●

A.
```
      WRITE(3,1)(BLOCK(I),I=1,600)
1     FORMAT(6F20.2)
```

209 If you answer agrees with this one (excepting, of course, the choice of index name and FORMAT number), you may skip to frame 214. If you did not obtain the correct result, continue to the next frame.

210 First of all, the WRITE statement (if you had no trouble with this, go on to the next frame now). After the word WRITE, place your output reference number and selected FORMAT statement number and follow this with the list (it was assumed that you would use self-indexed form rather than write out a list of 600 variable names with subscripts). The self-indexed form consists of the name (BLOCK), its dummy subscript, and the index definition (1 to 600).

Q. Write a suitable WRITE statement to print out a 20-number array called MATRX: _____

●●●

A. WRITE(3,1)(MATRX(I),I=1,20)

211 As for the FORMAT, you were asked to specify 120 print positions to print six numbers (obviously there must be 20 positions per number provided) with two decimal places. The mode is plainly real for the variable BLOCK. The only possible statement to fit these conditions is 1 FORMAT(6F20.2).

Q. Write a suitable statement to specify the layout of a printed line containing ten numbers of seven positions each, all real, with six decimal places.

●●●

A. FORMAT(10F7.6)

212 Q. Try another similar problem. Write a set of statements to control the writing of the array called VECT which contains 1000 values. These are to be printed, twenty per line on a 120-position line, with one decimal place.
_____.

●●●

A.
```
  WRITE(3,1)(VECT(I),I=1,1000)
1 FORMAT(20F6.1)
```

213 If your answer agrees with the one shown, you are getting the hang of this output business. If not, better go back to frame 180 and review before continuing.

214 Perform Exercise 3.13 in your problem book.

The preceding frames have illustrated printed output using the "F" conversion code. Real numbers may also be printed by using the "E" code. The next few frames will discuss this form of output.

Q. Real numbers may be printed under the control of either _____ or _____ codes.

●●●

A. E,F

215 Let's assume we have the number 123.46 stored in the computer. Under control of the FORMAT code E12.5 this number would print as bb.12346Eb03, where b=blank. Turn to Panel 3.11 for an illustration of this example, which is explained in the next few frames.

Q. In the above example, what is the total field width?

●●●

A. 12

216 Let's examine in detail how our number in storage has been converted to printed output. For one thing, the decimal point has been shifted to the left of the first significant digit. This adjustment of the number is known as "normalizing", and this normalized form is always used with E output.

Q. E conversion always normalizes numbers on output, which means that to the left of the first significant digit, a _____ will be printed.

●●●

A. decimal point

217 Although the decimal point has been shifted, notice that the E exponent (03) will correctly place the decimal point to give the true value of the number. The computer automatically provides the correct value for the exponent.

Q. To compensate for the normalized decimal point, a suitable value for the _____ is used.

●●●

A. E exponent

218 The number of decimal places printed is dependent on the decimal specification in the E conversion code. In this example, we specified 5 decimal places, and thus got 5 digits printed. If fewer decimal places had been specified, we would print fewer digits.

Q. If the conversion code had been E10.2 in this example, how many decimal places would the printed number contain?

●●●

A. 2 (bbb.12Eb03)

219 Notice that the E exponent in our printed number occupies four print positions. We have a position for the letter "E", a position for the sign of the exponent (blank in this example because the exponent is positive), and two positions for the exponent itself. These four positions must always be reserved when E output is used in addition to whatever space is taken by the number itself.

Q. (True or False) The output conversion code E3.1 is valid if used with a three-position number with one decimal.

●●●

A. False (the exponent portion of the number alone will use four positions of field width)

220 The one remaining position of our printed number to be discussed is the blank which immediately precedes the decimal point. This position is used for a minus sign if the number to be converted is negative. As you can see from this example, positive numbers do not have a printed sign.

Q. (True or False) The minus sign of a negative number will require one print position.

●●●

A. True

221 Since we've talked a good deal about reserving printed positions, let's total them all up. We have 4 positions for the E exponent, one for the decimal, and one for the minus sign. Using the same example, bb.12346Eb03, we have underlined the 6 reserved positions. As you can see, in addition to those 6 positions, we also need space for the significant digits of the number itself, which, in this case, has 5 positions.

Q. If we wanted to print a number with 2 significant digits, what is the minimum number of print positions needed?

●●●

A. 8 (6 + 2)

222 Just as with F conversion, additional space to the left of the printed number is advisable for readability. This can be accomplished in E conversion by making the field width code larger than the minimum. When this is done, blanks will be inserted to the left of the number to take up the excess positions.

Q. If in our example on Panel 3.11 we had used E14.5 as the conversion code, what would the printed number look like?

●●●

A. bbbb.12346Eb03

223 You may be wondering what advantage there is in using E format printing. Its value is that it is possible to specify a general printing layout that is valid for any internal real number, regardless of its size. For example, a commonly used E code is E20.7, where space for the maximum 7 significant positions is reserved, 6 positions are used for the sign, decimal point and E exponent, and 7 spaces are reserved for easy reading. (7+6+7=20)

Q. (True or False) Regardless of the size of the internal number, format code E20.7 would convert it correctly.

●●●

A. True

224 Turn to Panel 3.12 and study it carefully. Here you see the six-number array ALPHA printed as the result of the following statements:

```
1  FORMAT(3E20.7)
   WRITE(3,1)(ALPHA(I),I=1,6)
```

Q. If the FORMAT statement had been 1 FORMAT(E20.7), how many lines would print?

●●●

A. 6

225 Notice that on Panel 3.12 the size of the internal number does not affect the number of print positions allowed, since E20.7 will handle any number. This can be very helpful, because sometimes the size of numbers to be printed is not known.

226 Let's take a look at some of the printed numbers on the panel. ALPHA(1) shows the normalized printing of 7 digits, with a suitable exponent adjustment.

Q. No sign prints for ALPHA(1) because the internal number is _____.

●●●

A. positive

227 ALPHA(2) is pretty straightforward, except that two zeros are filled in the decimal portion of the number. This will happen if there are more decimal print positions than significant digits in the internal number. Notice also the minus sign.

Q. How many zeros would be filled in if we were printing a 1 digit number with the FORMAT code E20.7?

●●●

A. 6

228 ALPHA(3) is interesting because it illustrates the printing of a number of very small magnitude. Notice that the exponent E-10 will correctly express the magnitude of this number.

Q. The computer can print numbers of great or small magnitude in E format by adjusting the _____.

●●●

A. exponent

229 You have been shown printing under the control of the general FORMAT code E20.7. Other E FORMAT codes may be used, of course, as long as you keep in mind the size of the internal numbers. If insufficient space is reserved for E printing, truncation will occur just as it did when using the F code incorrectly. For example, the number -21.0057 will print as b-.210Eb02 under control of E10.3. Thus, three digits (057) have been lost.

Q. What will print if E11.2 is used with the number 1.1642?

●●●

A. bbbb.11Eb01

230 If you were able to answer the last question correctly, you have a pretty good grasp of E conversion. If you had trouble, you should return to frame 214 and review.

231 So far, the discussion of output statements has concerned itself only with printing. Another means of output is the use of punched cards or punched paper tape.

232 You will remember that numbers on cards or paper tape were read by similar statements with the input reference number identifying the input device to be used. The same form is used for card or paper tape punching statements. They are made up exactly like WRITE statements for printing, except that the output reference number will be assigned to a card or tape punch.

Q. Assume the output reference number 2 will reference the card punch. The statement number WRITE(2,1)A,B,C causes the computer to punch cards, according to FORMAT statement number _____.

●●●

A. 1

233 The only difference between the WRITE statement used for printing and the WRITE statement used for punching is that a different output device will be referenced by the output reference number. Whatever you may write with the printer, you may also punch on cards or paper tape.

Q. Change the statement WRITE(3,1)(A(I),I=1,100) to write the same information on the paper tape punch.

●●●

A. WRITE (4,1)(A(I),I=1,100)

234 Computer systems will vary in the number of input/output devices attached. If you are planning to run your program on someone else's computer, you should check to see what devices are available for your use, and use the input/output reference numbers accordingly.

Q. Write a pair of statements to punch the contents of the BLOCK array on paper tape (output reference number 4) such that each number is seven digits long with 5 decimal places. The BLOCK array has 100 numbers. Use F specification in the format statement. _____.

●●●

A.
```
1 FORMAT(100F7.5)
  WRITE(4,1)(BLOCK(I),I=1,100)
```

235 If your answer agrees with the one shown, skip to frame 240. If your answer does not agree, continue to the next frame.

236 The writing statement should present no real problem. All you need is the word WRITE followed by the paper tape output reference number (4), a FORMAT number (e.g. 1) and the self-indexed list for the BLOCK array (BLOCK(I),I=1,100). With appropriately placed commas and parentheses, the correct statement becomes:

WRITE(4,1)(BLOCK(I),I=1,100)

Q. Write a card punching statement to punch the contents of a 500-number array called GROUP.

●●●

A. WRITE(2,1)(GROUP(I),I=1,500)

237 The FORMAT required in the original problem was to specify one hundred seven-digit numbers with five decimal places. The only combination to achieve this is 1 FORMAT (100F7.5).

Q. Write a FORMAT statement with F specification to specify eight numbers of fifteen columns each, four decimal places:
1 FORMAT (_____)

●●●

A. 1 FORMAT(8F15.4)

238 Write a pair of statements to write the 100-number A array on the typewriter such that, when printed, the numbers will evenly fill ten lines (two decimal places for each number).

●●●

```
A.    1 FORMAT(10F12.2)
        WRITE(1,1)(A(I),I=1,100)
```

239 If your answer agrees with the one shown, you may continue to the next frame. If not, you had better go back to frame 180 and review before continuing.

240 Perform Exercise 3.14 in your problem book.

This chapter has, so far, introduced three new statement types:

READ
WRITE
FORMAT with E, F, and I codes

241 There are other forms of Input and Output statements in the 1130 FORTRAN language, all of which have similar characteristics to those covered so far. You may find them in the appropriate manual and may learn their use easily; consequently, they will not be covered in this manual. Instead the remainder of this chapter will be devoted to additional features of the Output statements covered already.

242 FORMAT statements are constructed identically for Input or Output statement use, with the exception that output FORMAT statements may have longer total field specifications for paper tape, and printer than are allowed for cards. By the same token, many of the features demonstrated for input FORMATS may be used also for Output statements.

243 The use of the "/" in an output FORMAT has the same basic meaning as it did for the input FORMAT: it will signify the end of one line or line image specification and the beginning of the next. Thus, the same statement can specify consecutive lines with different specifications.

Q. The statement FORMAT(5F20.5/6E20.6) specifies _____ numbers on the first line; _____ numbers on the second; if the list contains enough quantities, a third line would contain _____ numbers.

●●●

A. 5,6,5

244 The use of interior parentheses to establish a repetition point is useful for output FORMAT statements too. The example FORMAT(3I10/(6F20.6)) would cause the printing of a line of integer quantities followed by several lines of real numbers (assuming the list is long enough).

Q. The statement WRITE(3,1)I,J,K,(A(I),I=1,60) used with the FORMAT shown above would result in (how many?) _____ line(s) of integer numbers and _____ line(s) of real numbers.

●●●

A. 1,10

245 One interesting effect concerns the use of more than one "/" in an output FORMAT. Normally, one slash between conversion codes causes printed lines to be single spaced. But two slashes "//" cause double spacing and three slashes cause triple spacing. Moreover, each additional slash causes one more space making it easy to obtain quadruple, quintuple, etc., spacing between lines if desired. To put it another way, two slashes (double spacing) mean one blank line between two printed lines; three slashes (triple spacing) mean two blank lines between two printed lines, etc. Thus, FORMAT(3I10///(6F20.4)) specifies triple spacing from the first line to the next line; in other words, two blank lines between the first line and the next line. See examples below:

Single Spacing	Double Spacing	Triple Spacing	Quadruple Spacing
1234	1234	1234	1234
5678			
4321	5678		
2468		5678	
1359	4321		5678

Q. The statement FORMAT (6F20.5////(5F24.6)) will provide _____ blank lines between the first printed line and the next.

●●●

A. 3 (quadruple spacing)

246 If slashes appear at the beginning or end of the FORMAT statement, however, the number of blank lines before or after a printed line will equal the number of slashes. For example, FORMAT (///I5//) will cause three blank lines, before printing a 5-position integer, and two blank lines after printing.

Q. How many blank lines will precede the printing of the real number in the statement FORMAT(////E20.7)?

●●●

A. 4

247 The statement 1 FORMAT(I5////F5.2,I2//) would cause the following output:

Integer
(blank line)
(blank line)
(blank line)
Real, Integer
(blank line)
(blank line)

Q. How many blank lines are specified by the FORMAT statement 1 FORMAT(///E20.7//I5//)?

●●●

A. 6 (3+1+2)

248 Another interesting feature, available on Output statements only, is the ability to print the index value of a self-indexed list without actually generating a separate variable. For example, the statement WRITE(3,1)(I,A(I),I=1,10), which lists the index variable I separately, will print out values in the following order: the number 1, the value of A1, the number 2, the value of A(2), etc.

Q. How many values will be printed by the above statement?

●●●

A. 20 (each of the ten values of the A array with its associated index variable; for example, 1 A1, 2 A2,....10 A10)

249 Input and Output with FORTRAN is actually very straightforward, no tricks, and fairly consistent, even though there are many details to remember. Experience with this language and others makes one appreciate the ease of using FORTRAN input/output, however.

250 One very important feature of Output statements is the ability to print alphabetic characters. This is useful for labeling the answers, titling the output listing pages, etc. The writing of alphabetic information is accomplished by enclosing the data to be written in quotations marks within the FORMAT statement.

Q. (True or False) The computer can write English text through the use of quotation marks with the FORMAT statement.

●●●

A. True (English text is composed of ordinary alphabetic characters)

251 Alphabetic information that is to be printed (or written on tape or disk for later printing) is written verbatim into the FORMAT statement, always preceded by a single quotation mark and followed by another quotation mark.

Q. (True or False) By the above definition FORMAT(' THIS IS ALPHABETIC DATA') is a valid statement.

●●●

A. True

252 Whenever a FORMAT statement containing alphabetic data is used by an Output statement, that information enclosed by the quotation marks is printed (or written on tape or disk) exactly as it was written into the FORMAT at the time the program was written.

Q. (True or False) The statement FORMAT (' FORTRAN') would print the word FORTRAN when used with a WRITE statement for printing. _____

●●●

A. True

253 In other words, if you wanted to title a printed page with a 100-character sentence you would merely enclose the sentence in quotation marks, writing this directly in the FORMAT statement. This alphabetic information is then available for output any time that FORMAT is called upon by an Output statement.

Q. To provide the word OUTPUT as a page heading, you would write FORMAT (_____).

●●●

A. ' OUTPUT'

254 Actually, the information enclosed by quotes is not restricted to alphabetic characters. You may use all 26 alphabetic characters, all ten numeric digits 0 through 9, and the special characters + - * / = ( ) . , $ and blank.

Q. Write a suitable FORMAT statement to print X=$10,000.99

●●●

A. FORMAT (' X=$10,000.99')

255 Incidentally, information contained within quotation marks is called "literal data", because it is printed character for character, or literally.

256 Unlike the F, E, and I conversion codes, the literal data does not have a corresponding variable in an Output list, but rather, the computer simply places the information into the output image wherever it is called for in a FORMAT.

Q. (True or False) The sequence WRITE(3,1) ; 1 FORMAT ('bJOEbDOE,bDEPT.b241') would title a page with the sentence: JOE DOE, DEPT. 241 (b=blank)

●●●

A. True

257 Literal data can be used in the same FORMAT with either F, E or I codes. For example, the statement 1 FORMAT ('bTHEbANSWERbIS', F15.5) would be perfectly valid. If used, say with the statement WRITE(3,1)X, the computer would print first the indicated alphabetic information (14 columns) and then the value of the variable X (the next 15 columns).

Q. If X had a value of 1.23456 in the example above, the printed line would be _____.

●●●

A. bTHEbANSWERbISbbbbbbbb1.23456 (b indicates blank)

258 Literal data, then, can be used anywhere in an Output FORMAT to provide the ability to write words or symbols to clarify your output. It can be preceded or followed by F, E, or I conversion codes, and may be as long as desired, as long as it does not exceed the allowable print-line size. (Also, the literal data field plus any E, F, or I fields must not exceed this figure.)

259 The literal data characters printed may be alphabetic, numeric, or special characters. This type of information is given the name "alphanumeric" (or sometimes, "alphameric") to distinguish between this type and purely numeric data as converted by E, F, and I codes. The former term will be used in this course.

Q. Alphanumeric information can be printed by enclosing it with _____.

●●●

A. quotation marks

260 An interesting combination of literal data and the repeating FORMAT features enables you to title a printed page and follow this with an array of numbers, using just a single FORMAT statement. The sample program on Panel 3.13 will print a line of alphanumeric information, skip a line, and print any array. Turn to Panel 3.13 now.

261 Notice that the line of alphanumeric information will be printed only once; the inner parentheses will keep the Output statement's list of 1000 quantities repeating with the (10F12.4) specification. That inner parentheses set is very important in this example. Without this, the entire FORMAT specification would be repeated, including the alphanumeric information.

Q. (True or False) If the inner parentheses were omitted in the example on the panel, the computer would print in alternating fashion: "A ARRAY, ten numbers; A ARRAY, ten numbers; A ARRAY, ten numbers; etc.

●●●

A. True (complete with the line-spacing)

262 The use of quotation marks, then, is just a handy way of writing words, symbols, and labels to label your output. The rules are fairly simple. Any alphanumeric information that you wish to print is written directly in a FORMAT and enclosed in single quotes. Every time that FORMAT is called upon by an Output statement (or repeated) that field is written on paper, tape, or disk.

Q. Write a set of statements to print the following line:
bRESULTSbOBTAINEDbFROMbNEWTONbMETHODb
Double space after printing the line.
Print 50 numbers in an ANSWR array, ten numbers per full line (single spaced), each number having six decimal places.

●●●

A.
```
      WRITE(3,1)(ANSWR(I),I=1,50)
    1 FORMAT('bRESULTSbOBTAINEDbFROMbNEWTONbMETHODb'//(10F12.6))
```

263 If your answer agrees with the one shown, skip to frame 268. If you do not agree with this answer continue to the next frame.

264 The WRITE statement was probably no trouble. This list was self-indexed for the 50-number ANSWR array. The FORMAT statement, on the other hand, could be tricky. You were asked to start the printing with a line of alphanumeric information. This means that you start the FORMAT statement with literal data: ('bRESULTSbOBTAINEDbFROMbNEWTONbMETHODb'...

265 You were expected to double space after printing this line, so slash marks "//" should follow the alphanumeric field. This will allow the computer to print the alphanumeric field, skip the next line, and then look on further in the FORMAT for the remaining specifications.

Q. Write a FORMAT which will specify the writing of the word bANSWERb, two blank lines, and the writing of a 20-column integer number field.

●●●

A. FORMAT('bANSWERb'///I20)

266 The FORMAT statement requested for the problem of three frames back would be completed by the addition of the F code for the array to be printed. You were told that there would be ten 6-decimal numbers per full line meaning, for a 120-character line, that you would use (10F12.6).

Q. A 50-number array with the specification (10F12.6) would result in (how many?) _____ lines of printing.

●●●

A. 5

267 If you feel that your misunderstandings have been cleared up, you may continue to the next frame and finish the chapter. If questions still remain unanswered, turn to frame 250 and review before continuing.

268 Perform Exercise 3.15 in your problem book.

You have seen in the preceding frames how literal data is used with output statements. Literal data can also be used in a FORMAT statement that controls input. This use of literal data allows the program to accept alphanumeric information from an input device. Until now, we have not had the means for reading alphabetic information into the program.

Q. When a literal data FORMAT statement is used with an input statement, the computer will accept _____ information.

●●●

A. alphanumeric or alphabetic

269 Usually, data read in this manner is used for identification purposes. That is, it may be descriptive information pertinent to the record being read such as headings, names, or comments.

270 The input FORMAT statement with literal data is constructed exactly like the output FORMAT statement with literal data. For instance, if we wanted to read in 10 positions of alphanumeric information, our FORMAT statement would allow 10 positions between quotation marks, i.e.,: FORMAT ('bbbbbbbbbb').

Q. If we were reading 5 positions of alphanumeric data, how many positions should be given between the quotation marks?

●●●

A. 5

271 When an input literal data FORMAT statement is used with a READ instruction, here's what happens: alphanumeric data is read from the input device, with the number of characters read equal to the number of positions between the quotation marks. The data being read will replace, in computer storage, the characters between the quotes.

Q. Assume we are reading cards with the FORMAT statement 1 FORMAT('bNAMEbFIELD'). How many alphanumeric characters will be read from the cards?

●●●

A. 11

272 By using literal data as in the preceding example, we have the means of storing alphabetic information from incoming cards. In effect, you can envision this as though the incoming data is stored in the FORMAT statement, replacing whatever was written between the quotation marks. Then, by using the same FORMAT statement with a WRITE statement for printing, we would print the alphabetic data from the last card to be read.

Q. (True or False) When a FORMAT statement with literal data is used for reading, we have the means for storing alphanumeric information for future use.

●●●

A. True

273 Let's see how this might work. Suppose a card contained bJOHNbDOE,bJR. in columns 1 to 14. If this card were read with the statements 1 FORMAT('bNAMEbFIELDbbb'); READ(2,1) our FORMAT statement becomes, in effect, 1 FORMAT('bJOHNbDOE,bJR.'). If we now executed the print statement WRITE(3,1) the information printed would be JOHN DOE, JR. Notice that in this example we did not need either an input list, or an output list.

Q. Suppose the card above had contained bPETER,PAUL,+1 in columns 1-14. What would have printed?

●●●

A. PETER,PAUL,+1

274 An important point: each time a new input record is read, data from that record will replace the previously read data in the computer. Thus, the literal data in the FORMAT statement is only as current as the last record read. If another READ statement uses the same FORMAT statement, the incoming literal data will replace the old.

Q. The FORMAT statement will contain literal data only from the _____ card read.

●●●

A. last

275 Let's take a moment to review the use of literal data with input statements. We use this feature of FORTRAN so that the program can accept alphanumeric information from an input device. This information is usually of a descriptive nature, and is often associated with the input record from which it came. The number of characters read equals the number of positions between the quotation marks, and the data will remain available as long as no new information is read by the same FORMAT statement.

276 This method of handling alphanumeric information has a limitation, however. Although data can be read and written by this means, the program cannot change the data. This is so because the program has no way of addressing it - that is, the literal data does not have a name like a variable. Remember that there was no name in the input or output list which referred to the alphanumeric data.

Q. Although literal data can be read or written by a program, the program cannot _____ it.

●●●

A. change

277 Sometimes a programmer will want to alter alphanumeric data in some way - perhaps he wants to rearrange it or insert something or delete something. In order to give him this facility, a new conversion code is used. This new FORMAT code is "A", which stands for alphabetic or alphanumeric conversion.

Q. The "A" conversion code is used so that the programmer can change (what kind of?) _____ data.

●●●

A. alphabetic or alphanumeric (actually, A conversion will accept any of the characters available to the computer, including special characters)

278 When used in a FORMAT statement, the letter "A" is followed by an integer. This integer indicates the number of alphanumeric characters to be read or written under A conversion. For instance, the statement 1 FORMAT(A2) would specify that 2 alphanumeric characters are to be read or written.

Q. Would 1 FORMAT(A1) be a legal statement?

●●●

A. Yes

279 Unlike the literal data FORMAT statement, when A conversion is used there must be a corresponding variable name in the READ or WRITE statements. Thus, the statement 1 FORMAT(A2) might be combined with the statement READ(2,1)DESCR to read 2 alphanumeric characters into the variable DESCR.

Q. Given the statements 1 FORMAT(4A3) ; READ(2,1)ALPHA,BETA, GAMMA,DELTA how many characters would be read into each variable?

•••

A. 3. Notice the use of the 4 to cause repetition of the A code. This form is equivalent to (A3,A3,A3,A3).

280 Each real variable name has sufficient space for four A characters, and each integer variable name has space for only two A characters. Thus the statements READ(2,1)A,B,C,I,N,X; 1 FORMAT (3A4,2A2,A4) would cause 20 characters to be read into computer storage. A,B,C, and X would each contain four characters, and I and N would have two characters each.

281 It is quite important that enough space is allocated when reading in characters under A conversion. For instance, if 12 characters were to be read, but only 2 real variable names were given in the input list, the 4 excess characters would be ignored, (skipped) and would not enter computer storage.

Q. If the statements 1 FORMAT(A2,A1,A2,A3) ; READ(2,1)NAME were executed, how many characters would be ignored (skipped)?

•••

A. 6 (NAME would take the first 2 characters, but the remaining A characters have no variable names assigned to them.)

282 Let's recall the reason we have A conversion. We said that it enables the programmer to change alphanumeric data read by his program - something which could not be done with literal data FORMAT statements. The key to this new ability is that we have assigned names to the incoming data. Because the alphanumeric data now has a name it can be manipulated by regular program statements.

Q. One of the main differences between A and literal data conversion is that the A conversion data has a _____.

●●●

A. name

283 Incidentally, A codes, literal data, and E, I, or F codes can all be intermixed within a FORMAT statement. For example, the statement 1 FORMAT ('bMOONbSHOTb',I5,E20.7,'bEQUALS',F8.2,A4///) would be perfectly legal. Turn to Panel 3.14 for an illustration of how this might print, assuming an appropriate WRITE statement.

Q. What is the value of RHO in the illustration?

●●●

A. -34.12

284 Notice that whatever an A, E, I or F conversion code appears, there must be a corresponding variable name in the output list. Literal data, however, does not require a name in the list.

Q. (Yes or No) Would the statements 1 FORMAT ('bHELP?") ; WRITE(3,1) ; cause HELP? to print?

●●●

A. Yes

285 On Panel 3.14, horizontal spacing along the printed line is provided by two methods 1) blank spaces within the literal data, and 2) excess leading positions within the E and F conversion codes.

Q. How many leading blank spaces are provided by the E and F codes, respectively?

●●●

A. 6, 2

286 The methods of horizontal spacing mentioned in the previous frame are somewhat tedious, however. To give the programmer more flexibility in spacing a report, FORTRAN provides another specification code will be covered in the following frames.

287 The additional coversion code for FORMAT statements which proves useful for spacing portions of your output on a given line is the X conversion, consisting of an integer followed by the letter X. This code merely causes a certain number of blank columns to be inserted in the output image, the number of blanks being defined by the integer preceding the letter X.

Q. The statement FORMAT(F14.4,10X,F12.6) will cause the placement of _____ blanks between the two F conversions.

●●●

A. ten

288 Use of the X code in the FORMAT statement would effect the same result as a literal data field with nothing but blank characters. For example, the statements FORMAT(...,20X,...) and FORMAT(...,'bbbbbbbbbbbbbbbbbbb',...) are equivalent, but the former is obviously more compact.

Q. Write a FORMAT statement whose output will consist of: ten blanks, the letter "A", twenty blanks, the letter "B", twenty blanks, the letter "C", forty blanks, and the letters "ERROR", making use of X and literal data conversions FORMAT(_____).

●●●

A. FORMAT(10X,'A',20X,'B',20X,'C',40X,'ERROR')

289 The X specification can also be used with input records. In this case, the X code indicates how many characters of the input record are to be skipped. For example, if a card has six 10-column fields of integers, and the second field is not to be read, the statement: 1 FORMAT(I10,10X,4I10) may be used, along with an appropriate READ statement.

Q. Write a FORMAT statement that will allow reading of a 2 decimal real number (in E notation) in columns 1-10 of a card, and then an integer in columns 71-80.

●●●

A. 1 FORMAT(E10.2,60X,I10)

290 One further example of this very useful feature: if the first 40 columns of a card were to be skipped, the statement 1 FORMAT(40X,...) would handle it nicely.

Q. (True or False) On input, X specification allows input record positions to be skipped, while on output the X specification indicates the number of blanks to be inserted in the output line or image.

●●●

A. True

291 Before we can go further in discussing FORMAT statements for printing, the topic of vertical paper spacing or "carriage control" will be discussed. It's necessary to do this now, because FORMAT statements for printing must be designed with carriage control in mind. So far, we have skirted this issue in the text, but as we talk more in detail about how a printed line actually looks in relation to the FORMAT statement, this topic is a must.

292 The printer for the 1130 computer system does not actually print the first character in the output image. This character is interpreted by the printing device as a "carriage control" character which causes the printer to space down the page in various ways.

Q. In the statement FORMAT('bANSWER') the first character (carriage control character) is _____.

●●●

A. b (blank)

293 This fact holds true whether the first item in the print image is a result of a numeric field (E,F, or I conversion) or alphanumeric field (literal data, or A conversion). The best way to think of this is to picture a printed line image where the first character, whatever it is, will not be printed and the second character in the image will be the first character in the line that is actually printed (you may recall that previous examples have had blanks there).

Q. (True or False) The very first character in any printer-line image is not printed.

●●●

A. True

294 On 1130 computer systems, the following characters, appearing in column one of a printer-line image, will cause the indicated carriage control operation to occur:

b (blank)	Normal (single) space
0 (zero)	Double space
1 (one)	Skip to a new page
+ (plus)	Do not space (write on same line as before)

In all cases, the indicated control is executed before the actual line is printed.

295 Programming the desired carriage control character into the print image (either for the printer or for a tape print-image) will give the programmer a useful control of his printed output. For example, if you wish your output to begin on a fresh page in the printer, FORMAT('1') used with a WRITE statement will accomplish this. Of course, the character 1 could have been part of a larger FORMAT, too.

Q. The carriage control character in FORMAT(10X,'ANSWER') would be _____.

●●●

A. blank (from the X code)

296 If you answered this correctly, you have probably grasped the meaning of this concept pretty well. The statement FORMAT (10X,'ANSWER') sets up a printed-line image of ten blanks and the word ANSWER. Therefore, nine blanks and the word ANSWER will be printed, with the first blank acting as carriage control. Remember, it is the first character in the image which has this function, not the first character in the FORMAT statement itself!

297 While the carriage control feature can be very helpful, it also can do strange things if you do not handle it carefully. There are many ways in which an odd character can unintentionally appear in the first column of the printed-line image. The next few frames will cite a few common examples you should watch out for.

REFERENCE INDEX

R29-0103-0

Printed in U.S.A.

R29-0103-0

IBM

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