COMPUTATION CENTER
Massachusetts Institute of Technology
Cambridge 39, Massachusetts

November, 1964

TO: FMS Users

FROM: Consulting Staff

SUBJECT: COMMON PROGRAMMING ERRORS

The following is a list of common errors made by FORTRAN Monitor System users as compiled by the Consulting Staff at the Computation Center. It is hoped that this list will help the new programmer avoid many of the more common pitfalls and provide a guide for the more experienced programmer while debugging his program.

Input/Output Errors and Common Causes in FORTRAN or MAD

1. DIMENSION statements are not large enough for the data being read.

2. Fixed-point variables read by an I format must be right-adjusted in the field; i.e., blanks to the right of the number in a field are interpreted as zero.

3. Floating-point variables read by an E format must have the exponent punched to the right of a field.

4. Data is read by the computer beginning in column 1, not 7.

5. The control card * DATA must contain an * in column 1 and DATA anywhere between 7 and 72, nothing else.

6. Array names read in without the use of subscript must be read for the maximum size of the array as it appears in the DIMENSION statement.

7. Data read with an Ø format can contain only numbers 0 through 7.

8. COMMON statements must agree between subprograms.

9. The number of data cards to be read must agree exactly with the number of data cards in the deck. (Note: F2PM Request Cards are not considered part of the binary deck.)

10. Binary data cards cannot be read by FORTRAN programs.

11. There should be only one * DATA card. (Errors usually occur in reproduced binary decks where the DATA card is at the end of the binary deck.)

12. There can be no comment cards among data cards.

13. When reading data cards, if the list goes beyond or up to a slash, the slash will be interpreted and an additional data card will be read.

14. Arguments for subprograms must agree exactly with the arguments specified in the calling subprogram.

15. Floating-point variables cannot be read properly with an I format.

16. If a card or cards from the binary deck are missing an illegal character in format statement may arise.

17. If P is used to scale floating-point variables, the power is used throughout the remainder of the format unless otherwise indicated or unless compatible with MAD. See CC-186.

18. The count for the number of Hollerith characters in a format statement must be correct; i.e., it should not include as Hollerith any other part of the format.

19. Column 1 of the off-line printer is for program control only.

20. Fixed-point variables cannot be read properly according to an E or F format.

21. The largest number of characters which can be read into one machine location according to an A or C format is 6. If more than 6 characters are to be read there must be more than one machine location.

22. Programmers should not read a tape other than 4, and/or write a tape other than 2, for normal input and/or output.

23. For output of an E format a minimum of 6 spaces plus the number of decimal places is required to get the complete number.

24. When the specification X is used for blank fields, the number of blanks desired must precede the X, not follow it. One should note that even the number one must be present.

25. When parentheses are used in a format statement in FORTRAN to denote repetition, the repetition occurs from the last left parenthesis to the last right parenthesis unless the parenthesis is preceded by a number.

26. When there hasn't been enough space specified for printing of a variable, the most significant portion of the variable, including the sign, is dropped and the last significant portion of the variable is printed if IOHSIZ has been called. See CC-186.

27. The FORTRAN statement BACKSPACE backspaces 1 logical record, not a physical record, on binary tape.

28. When reading data, a slash at the beginning of a format causes every other card to be read, not every card.

29. Only 131 characters plus the program control character can appear on-line printed by an IBM 1401.

FORTRAN Post-Mortem and Common Errors

1. When requesting a post-mortem of the main subprogram, the name MAIN must be enclosed in brackets, as (MAIN).

2. When requesting a post-mortem of octal information, a slash must precede each octal location requested.

3. When requesting a post-mortem of an (ENTIRE) subprogram, the post-mortem will not include those variables which appear in a COMMONstatement.

4. When requesting a post-mortem of a FORTRAN tape, a programmer must remember that the first record on tape is record 0, file 0.

Missing Subprograms and Common Causes

1. The subprogram listed as missing was actually an array not included in a DIMENSION statement.

2. The subprogram listed as missing was actually a variable which was intended to be multiplied by another variable, but the * was omitted.

3. The subprogram listed as missing was actually an open or built-in library function which when requested had the F at the end of the name missing.

4. The subprogram listed as missing was actually after the * DATA card and not before it.

5. The subprogram listed as missing was actually not included on the library tape or in the deck.

6. If the letter Ø or number 0 is used for a subprogram name, sometimes they are incorrectly interchanged.

7. Key punch error giving a different name for array. It is assumed by the compiler to be a function.

8. The binary deck may be out of order; for example, the program card may not be the first card of a subprogram.

Incorrect Results and Common Causes

1. Because FORTRAN sets subscript indices as early as possible on a program, changing a fixed-point variable appearing in COMMON in a called subprogram may not change the value used for that variable in subsequent indexing operations in the calling subprogram. See Part IV of the FORTRAN II Programming Manual.

2. The original equation or thesis is incorrect.

3. Parentheses in a FORTRAN statement are incorrectly placed.

4. DIMENSION statements are not large enough for the data read or calculated.

5. If an EQUIVALENCE statement is used for variables in COMMON, then that EQUIVALENCE statement must appear in all subprograms which contain the same COMMON statement.

6. Somewhere in key punching the letter Ø was mistaken for the number 0 in a variable name.

7. The logical sequence of instructions was not programmed as intended. A flow chart is helpful in such a case. The path of program and flow chart can then be compared.

8. Incorrect octal correction cards.

9. Incorrect binary cards: Check binary deck - every card should contain a 7 or 9 punch in column 1 and either an 11 and/or 12 punch in column 1, nothing else.

10. Fixed-point variables in COMMON when used as subscript in DO loops cannot be transmitted to CALLED subprograms unless mentioned on the left-hand side of an = sign within the DO loop.

11. When raising variables to a negative power by means of **, the power preceded by a minus sign must be enclosed by parenthses.

12. No two operators can appear in sequence. For example, A/-B will not work properly, but A/(-B) will work properly.

13. If several DO loops are nested within each other the indexing parameter must not be reset by one of the nested DO statements. Note that arrays indexed in an input/output statement look like a DO loop to FORTRAN.

14. A function subprogram compiled by FORTRAN cannot be called with a terminal F if the function name is more than 3 letters.

15. In a computed GØ TØ statement the value of the fixed-point variable must be less than or equal to the number of statement numbers specified.

16. If an array is indexed in an input/output list, it must also appear in a DIMENSION statement. FORTRAN does not check for this.

17. If possible, programmers should use fixed-point variables to control an iterative process where the only way to get out is through an IF statement. If it is necessary to use a floating-point variable, one should not be testing for an exact zero, but something close. The possiblity of round off error causing an infinite loop can then be avoided.

18. When calling a subprogram with fixed arguments, such as 1. or 2 rather than A or I, one cannot reset these arguments within the subprogram without causing trouble.

Program Stops and Common Causes in FORTRAN Compiled Programs

1. DIMENSION statements are not large enough for data being read or calculated.

2. The number of arguments between a calling and called subprogram do not agree.

3. Incorrect octal correction cards.

4. Incorrect binary cards. Check binary deck - every card should contain a 7 and 9 punch in column 1 and either an 11 and/or 12 punch in column 1 nothing else, except possibly a zero punch which indicates that the checksum should be ignored.

5. If the library functions for square root and arctangent are requested as follows: SQR (argument) or ATN (argument), without the terminal F, a program stop will probably occur. Such is the case for any library function which should end with an F.

6. If the program stops with a select of the card reader, it is because the MAIN program did not terminate with a GØ TØ statement, a CALL EXIT statement, a STOP statement, or a PAUSE statement or because a subprogram did not terminate with any of the above statements, in addition to a RETURN statement. Note that STOP and PAUSE statements should not be used when the program is executed under the FORTRAN Monitor System.

7. A card in the binary deck is lost or misplaced. Check the card labelling, if present.

8. All subroutine arguments which are arrays must appear in a DIMENSION statement. If not, and the variable appears to the right of an arithmetic statement, it will look like a dummy subprogram name (F card type) to FORTRAN.

9. It is possible that a card in the binary deck is missing. Check the labelling in columns 73-80 of the binary deck.

Double-Precision and Complex Arithmetic Errors

1. If a program using double-precision or complex arithmetic wishes to link his subprograms with COMMON statements and some of the variables which appear in COMMON are single-valued double-precision or complex names not used in the subprogram, then these variables must appear in a DIMENSION statement with a D in column 1 in that subprogram.

   For example:

```
D       DIMENSION   A (1), B (1), C (100)
```

   Where A and B are referred to as single-valued floating-point names and C is referred to as an array.

2. Double-precision and complex arithmetic cannot be used on fixed-point variables.

MAD Errors

1. Mode of functions must be specified if different from normal mode.

2. Limit of THROUGH statements must be a Boolean expression.

3. Cards with an * in column 1 cannot be included in a MAD subprogram.

4. Special care should be taken when a statement label is used in an expression.

5. When interchanging FAP and MAD one must remember that MAD only saves index register 4, and returns to 1,4.

Errors in the Usage of Scratch or Special Tapes

1. All scratch tapes should be rewound at the beginning of a program.

2. Once a program has begun writing on a tape, the tape must be rewound before any reading can begin.

3. The density of a tape must be properly specified; i.e., if a tape is written in low density, then it must be read in low density, and simillarly if a tape is written in high density, then it must be read in high density. Note that low density is 200 cpi and high density is 556 or 800 cpi.

Program is too Large to Fit into Core

1. If program and loading tables overlap it is necessary that large arrays, the equivalent of about 5000 locations, be placed in a COMMON statement.

2. If program and data overlap, it is necessary to compute the size of each user subprogram and each library subprogram. (This can be done by using the FMS Library Tape listing, available in Room 26-058). When this is done the following alternatives are available:

   a) Decrease the size of your program by reducing the size of DIMENSION statements.

   b) Break the program up into CHAIN links.

   c) Make use of the EQUIVALENCE statement, if possible.

d) Use dummy (F2PM) which saves about 2800 locations. One should bear in mind that the facilities of (F2PM) are then no longer available.

e) Obtain the following non-library routines from the open files in Room 26-058: IØH, SCH, CSHM, F2PM, EXIT, EXE, and CHAIN and DUMP, if used. One should bear in mind that there will be no blocked output on logical tape 2 if these routines are used. The result will be an increase in the running time of the program.