A111207 263405
nbs special publication 399
Volume 1
U.S. DEPARTMENT OF COMMERCE / National Bureau of Standards

National Bureau of Standards Library, E. 01 Admin. Bldg.

OCT 11981
19102!

# NBS FORTRAN Test Programs 

## Volume 1-Documentation for Versions 1 and 3

```
QC 100 . 457 no. 399
v. 1
1974
```


## NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards ${ }^{1}$ was established by an act of Congress March 3, 1901. The Bureau's overall goal is to strengthen and advance the Nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the Nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau consists of the Institute for Basic Standards, the Institute for Materials Research, the Institute for Applied Technology, the Institute for Computer Sciences and Technology, and the Office for Information Programs.

THE INSTITUTE FOR BASIC STANDARDS provides the central basis within the United States of a complete and consistent system of physical measurement; coordinates that system with measurement systems of other nations; and furnishes essential services leading to accurate and uniform physical measurements throughout the Nation's scientific community, industry, and commerce. The Institute consists of a Center for Radiation Research, an Office of Measurement Services and the following divisions:

Applied Mathematics - Electricity - Mechanics - Heat - Optical Physics - Nuclear Sciences ${ }^{2}$ - Applied Radiation ${ }^{2}$ - Quantum Electronics ${ }^{3}$ - Electromagnetics ${ }^{3}$ - Time and Frequency ${ }^{3}$ - Laboratory Astrophysics ${ }^{3}$ - Cryogenics ${ }^{3}$.

THE INSTITUTE FOR MATERIALS RESEARCH conducts materials research leading to improved methods of measurement, standards, and data on the properties of well-characterized materials needed by industry, commerce, educational institutions, and Government; provides advisory and research services to other Government agencies; and develops, produces, and distributes standard reference materials. The Institute consists of the Office of Standard Reference Materials and the following divisions:

$$
\begin{aligned}
& \text { Analytical Chemistry - Polymers - Metallurgy - Inorganic Materials - Reactor } \\
& \text { Radiation - Physical Chemistry. }
\end{aligned}
$$

THE INSTITUTE FOR APPLIED TECHNOLOGY provides technical services to promote the use of available technology and to facilitate technological innovation in industry and Government; cooperates with public and private organizations leading to the development of technological standards (including mandatory safety standards), codes and methods of test; and provides technical advice and services to Government agencies upon request. The Institute consists of a Center for Building Technology and the following divisions and offices:

Engineering and Product Standards - Weights and Measures - Invention and Innovation - Product Evaluation Technology - Electronic Technology - Technical Analysis - Measurement Engineering - Structures, Materials, and Life Safety * Building Environment ${ }^{4}$ - Technical Evaluation and Application ${ }^{4}$ - Fire Technology.

THE INSTITUTE FOR COMPUTER SCIENCES AND TECHNOLOGY conducts research and provides technical services designed to aid Government agencies in improving cost effectiveness in the conduct of their programs through the selection, acquisition, and effective utilization of automatic data processing equipment; and serves as the principal focus within the executive branch for the development of Federal standards for automatic data processing equipment, techniques, and computer languages. The Institute consists of the following divisions:

Computer Services - Systems and Software - Computer Systems Engineering - Information Technology.

THE OFFICE FOR INFORMATION PROGRAMS promotes optimum dissemination and accessibility of scientific information generated within NBS and other agencies of the Federal Government; promotes the development of the National Standard Reference Data System and a system of information analysis centers dealing with the broader aspects of the National Measurement System; provides appropriate services to ensure that the NBS staff has optimum accessibility to the scientific information of the world. The Office consists of the following organizational units:

Office of Standard Reference Data - Office of Information Activities - Office of Technical
Publications - Library - Office of International Relations.

[^0]
## NBS FORTRAN Test Programs

## Volume 1-Documentation for Versions 1 and 3

Frances E. Holberton and
Elizabeth G. Parker

Institute for Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

## ${ }^{\text {t. Special }}$ publication no. 399 -Volume 1


U.S. DEPARTMENT OF COMMERCE, Frederick B. Dent, Secretary NATIONAL BUREAU OF STANDARDS, Richard W. Roberts, Director

# Library of Congress Cataloging in Publication Data 

Holberton, Frances E
NBS FORTRAN test programs.
National Bureau of Standards Special Publications 399.
CONTENTS: V. l. Documentation for versions 1 and 3.--V. 2.
Listings for version l.-V. 3. Listings for version 3.
Supt. of Docs. No.: C 13.10:399.

1. Computer programs-Testing. 2. FORTRAN (Computer program language) I. Parker, Elizabeth G., joint author. II. United States. National Bureau of Standards. III. Title. IV. Series: United States. National Bureau of Standards. Special Publication 399.

QCl00.U57 no. 399 [QA76.6] $389^{\prime} .08 \mathrm{~s}$ [001.6'425]
74-12314

# National Bureau of Standards Special Publication 399 

Nat. Bur. Stand. (U.S.), Spec. Publ. 399, 171pages (Oct. 1974)
CODEN: XNBSAV

## FOREWORD

It has now become imperative, because of the multitude of computers on the market and the corresponding multitude of FORTRAN compilers, to develop a means of testing the overall quality of these compilers, thereby making meaningful comparisons possible. The National Bureau of Standands has sponsored a project to develop methods and tools to assist in the evaluation process. Before the evaluation process can be undertaken, it is necessary to develop a primary tool, such as a set of FORTRAN programs which can validate whether a FORTRAN compiler is in compiance with the FORTRAN specification as described in the American Standand FORTRAN document X3.9-1966.

In 1966 the National Bureau of Standards formulated the design criteria and specifications for the development of such a set of FORTRAN programs. The initial implementation of this design was performed, under contract, by the Advanced Computer Techniques Corporation in 1967. Since this time, these programs have been desk checked, computer checked, revised, extended, many test units replaced, and the system reorganized to improve the tests and decrease the difficulty of performing the actual validation process.

Version 2 of these test programs was prepared by NBS, under contract, for the Joint Technical Support Activity of the Defense Communications Agency.

The purpose of these FORTRAN programs is to assist in the validation of FORTRAN compilers. There is no attempt to measure the performance of the compiler or the object program efficiency.

Currently, the FORTRAN Standard, ASA X3.9-1966, is undergoing revision and the FORTRAN language is being extended by the X3J3 technical conmittee of the American National Standards Institute (formerly identified as the American Standards Association). The revised FORTRAN Standard will be considered for Federal adoption. This will, if approved, require that the test programs be revised accordingly.

The National Bureau of Standards wishes to thank the Bell Telephone Laboratories for the preparation of the camera-ready copy of the program listings derived from the NBS FORTRAN Test Program Distribution Tape. These listings appear in Volumes 2 and 3 of this report.

## NBS FORTRAN TEST PROGRAMS

The NBS FORTRAN test programs, written in Standard FORTRAN, are designed to test whether a FORTRAN compiler accepts the forms and interpretations of the FORTRAN language as described in the American National Standard FORTRAN document X3.9-1966. The test programs, comprised of ll6 test units, are structured into two versions, each containing approximately 14,500 punch card images. The test units may be used as separate executable FORTRAN programs, or may be linked end to end with other test units, with a minimum of user effort, to improve operating efficiency. Version $l$ is structured into ll6 executable FORTRAN programs, and Version 3, containing the same 116 test units, iss structured into 14 executable FORTRAN programs for use on lange FORTRAN processors.

The test program design criteria was to:

- Constrain all test programs to the FORTRAN Standard X3.9-1966.
- Reduce the effect of those areas in which the FORTRAN Standard does not prescribe a method or solution, e.g., range, precision, size of computer, etc
- Simplify the use of the FORTRAN test programs.
- Test FORTRAN language elements before they are used in support of other tests.
- Maintain an open ended system so that tests may be changed or added.

The test programs require the use of a cand reader, printer and one intermediate tape unit.

During the development of the test systems ten different computing systems were used, and the current set of tests were run on five major systems. The largest test unit required less than 3,000 words of memory and when structured into 14 executable programs the largest program required less than 6,000 wonds of memory to execute the compiled prognams. The test units, for the most part, are straight line programs and during the debugging of the tests, less than 15 minutes was required to compile and execute the set of 14 structured FORTRAN programs, excluding cand read and printer time.

The magnetic tape, containing the NBS FORTRAN Test Programs, Version 1 and Version 3, together with the documentation ( 3 volumes) is available from:

> National Technical Information Service
> Department of Commerce
> 5285 Port Royal Road
> Springfield, Virginia 22151

The magnetic tape is available in 800 cpi recording density in the following forms:

> 7 track even parity BCD, recorded from the FORTRAN H set punch card code (see FORTRAN Standard, Appendix D X3.9-1966)
> 9 track odd parity EBCDIC code
> 9 track odd parity ASCII code

Key words: Computer programming language; EORTRAN: FORTRAN validation; language validation; standards FORTRAN; test program design.

## INTRODUCTION

This document, Volume 1 of three volumes, contains the documentation, including the systems design, operating procedures, expected test results and distribution tape organization for a set of FORTRAN Test Programs developed by the Institute for Computer Sciences and Technology, National Bureau of Standards. These test programs are written in ASA Standard FORTRAN and test the language elements described in the ASA Standard FORTRAN document X3.9-1966.

The NBS FORTRAN Test Programs, Version l, contain 116 test units, each structured as an executable FORTRAN program.

The NBS FORTRAN Test Programs, Version 3, containing the same ll6 test units, structured into 14 executable FORTRAN Programs, have been organized for use on large FORTRAN Processors for the purpose of reducing the number of systems control cards needed to perform the tests.

The NBS FORTRAN Test Programs, recorded on magnetic tape as approximately 14,500 punch cand images for each version, contain the FORTRAN source language programs and data. Extensive FORTRAN comment lines are interspersed throughout the programs to enable the user to both run the programs and determine the nature of the tests without the need for additional documentation. The test results contain information related to the expected results.

Volume l, Section I describes the system design, the programming techniques and conventions used in the program development and should enable the user to extend, alter or reorganize the test programs.

Volume l, Section II defines the organization and operating procedure for performing the tests and contains a set of representative results obtained from actual running of the test programs on several FORTRAN processors.

Volume l, Section III describes the order and location of each test unit and data as recorded on magnetic tape for distribution.

Volume 2 contains the program listings for the NBS FORTRAN Test Programs, Version 1.

Volume 3 contains the program listings for the NBS FORTRAN Test Programs, Version 3.

## Table of Contents

Foreword


#### Abstract

Introduction


Section I SYSTEMS MANUAL
A. FORTRAN Test Program Design I-A-1

A1. Objective I-A-1
A2. Design Criteria I-A-1
A3. Design Considerations I-A-1
A4. Design Implementation I-A-4
A5. FORTRAN Concepts Excluded from the Test Programs I-A-6
A6. Interpretations Made to the FORTRAN Standard I-A-8
B. Description of Each Segment I-B-1
C. Test Unit Segments Indexed to the FORTRAN Standard I-C-1 Document ASA X3.9-1966
D. Program Information I-D-1

D1. Conventions Used in the Test Programs I-D-1
D2. Assumed Levels for Non-specified FORTRAN Areas I-D-4 D3. Names and Statement Numbers Used in the Test I-D-6 Programs
E. Structuring, Restructuring and Extending the Test Programs I-E-1

E1. Program Structure I-E-1
E2. Consolidating Test Program Units Using Version 1 I-E-1
E3. Deleting a Section of a Test Unit I-E-4
E4. Deleting an Entire Test Unit I-E-4
E5. Adding to a Test Unit I-E-4
E6. Adding New Test Units I-E-5
F。 Difficulties Encountered During the Test Program Development I-F-1
F1. Interpreting the FORTRAN Standard I-F-1
F2. Precision, Conversion and Maximum Value of I-F-1 Numeric Data
F3. Meaningful Tests and Comprehensible Results I-F-1
F4. FORTRAN Compilers with Language Extensions I-F-3
F5. Performing the Tests I-F-3
G. References

Section II USERS MANUAL
A. Operating Procedures II-A-1

A1. Organization of Tests and Facilities Requirements II-A-1 A2. Input Data Preparation
A3. List of Test Programs for Version 1 II-A-3

A4. List of Test Units by Parts for Version 3
II-A-4 II-A-11
B. Procedures for Isolating Test Unit Failures from Version 3

II-B-1
B1. Deleting a Test Unit
B2. Creating a Single Test from a Deleted Unit II-B-1
C. Sample Test Results

II-C-1
C1. Interpreting the Test Results
II-C-1
C2. Test Results II-C-2
Section III DISTRIBUTION. TAPE ORGANIZATION
A. General Description III-A-1

A1. FORTRAN Test Programs and Data Version 1
III-A-3
A2. FORTRAN Test Programs and Data Version 3

| 008 - FMTRW | Formatted Input/Output | II- C-3 |
| :---: | :---: | :---: |
| 009 - AFRMT | A-Conversion | II-C-7 |
| 010 - DATA2 | DATA Statement Use | II- C- 8 |
| 011 - AASGN | Real and Integer Arith Assignment. Stmnts. | II- C-9 |
| 013 - DASGN | Simple D.P. Assignment Statements | II- C-11 |
| 015 - CASGN | Simple Complex Assignment Statements | II- C-15 |
| 016 - LASGN | Logical Assignment Statements | II- C-19 |
| 017 - INTRL | Arithmetic Assignment Statements | II- C-20 |
| 020 - UGOTO | Unconditional GO TO Statements | II-C-22 |
| 021 - AGOTO | GO TO Assignment Statements | II-C-22 |
| 022 - CGOTO | Computed GO TO Statements | II- C-23 |
| 030 - ARBAD | Basic Addition | II-C-23 |
| 031 - ARFAD | Double Precision Addition | II- C-24 |
| 032 - ARBSB | Basic Subtraction | II-C-24 |
| 033 - ARFSB | Double Precision Subtraction | II-C-24 |
| 034 - ARBAS | Basic Addition and Subtraction | II- C-24 |
| 035 - ARFAS | Addition and Subtraction of D.P. Values | II- C-25 |
| 036 - ARBMI | Multiplication of Integer Values | II- C-25 |
| 037 - ARBMR | Multiplication of Real Values | II- C-25 |
| 038 - ARFMD | Multiplication of D.P. Values | II-C-25 |
| 039 - ARBDV | Division of Integer and Real Values | II-C-26 |
| 040 - ARFDV | Division of D.P. Values | II-C-26 |
| 041 - ARBEX | Exponentiation of Integer and Real Values | II- C-26 |
| 042 - ARFEX | Exponentiation of D.P. Values | II-C-26 |
| 043 - ARBHI | Hierarchy of Operators and Parentheses | II-C-27 |
| 050 - SBB67 | Subscripts of Integer, Real Arrays $\mathrm{v}, \mathrm{k}$ | II-C-27 |
| 051 - SBB45 | Subscripts of Int., Real Arrays $\mathrm{v}+\mathrm{k}, \mathrm{v}-\mathrm{k}$ | II- C-27 |
| 052 - SBB13 | Subscripts of Int., Real Arrays $c^{*} v, c^{*} v+k$, $\mathrm{c} * \mathrm{v}-\mathrm{k}$ | II- C-28 |
| 053 - SBF17 | Subscripts of D.P. Arrays $v, k, c^{*} v, c^{*} v+k$, $\mathrm{c} * \mathrm{v}-\mathrm{k}, \mathrm{v}+\mathrm{k}, \mathrm{v}-\mathrm{k}$ | II- C-28 |
| 054 - SIMIF | Arith. IF, Logical IF followed by GO TO | II- C-28 |
| 055 - IFABS | Intrinsic Functions ABS, IABS | II- C-29 |
| 056 - IFFLT | Intrinsic Function FLOAT | II- C-29 |
| 057 - IFFIX | Intrinsic Function IFIX | II- C-29 |
| 058 - IFSGN | Intrinsic Functions SIGN, ISIGN | II-C-29 |
| 059 - IFDAB | Intrinsic Function DABS | II- C-29 |
| 060 - IFTRN | Intrinsic Functions AINT, INT, IDINT | II- C-30 |
| 061 - IFMOD | Intrinsic Functions AMOD, MOD | II- C-30 |
| 062 - IFMAX | Intr. Funct. AMAX0, AMAX1, MAX0, MAX1, DMAX1 | II- C-31 |
| 063 - IFMIN | Intr. Funct. AMIN0, AMIN1, MIN0, MIN1, DMIN1 | II-C-32 |
| 064 - IFDSG | Intrinsic Function DSIGN | II- C-32 |

## NBS FORTRAN Test Program Results

Table of Contents

| 065 - IFDIM | Intrinsic Functions DIM, IDIM | II-C-33 |
| :---: | :---: | :---: |
| 066 - IFSGL | Intrinsic Function SNGL | II-C-33 |
| 067 - IFREL | Intrinsic Function REAL | II- C-34 |
| 068 - IFIMG | Intrinsic Function AIMAG | II- C- 34 |
| 069 - IFDBL | Intrinsic Function DBLE | II-C-35 |
| 070 - IFCPX | Intrinsic Function CMPLX | II- C- 35 |
| 071 - IFCJG | Intrinsic Function CONJG | II-C 35 |
| 072 - IFBMS | Integer and Real Intrinsic Functions | II- C-36 |
| 073 - IFFMS | Int., Real and D.P. Intrinsic Functions | II-C-36 |
| 080 - EXPON | Basic External Function EXP | II- C-37 |
| 081 - DEXPO | Basic External Function DEXP | II- C-38 |
| 082 - CEXPO | Basic External Function CEXP | II- C-38 |
| 083 - LOGTM | Basic External Function ALOG | II- C-40 |
| 084 - DPLOG | Basic External Function DLOG | II- C-40 |
| 085 - CXLOG | Basic External Function CLOG | II-C-41 |
| 086 - COLOG | Basic External Function ALOG10 | II-C-42 |
| 087 - DCLOG | Basic External Function DLOG10 | II-C-43 |
| 088 - SINUS | Basic External Function SIN | II- C-43 |
| 089 - DPSIN | Basic External Function DSIN | II- C-44 |
| 090 - CSICO | Basic External Functions CSIN, CCOS | II- C-44 |
| 091 - COSNS | Basic External Function COS | II- C-45 |
| 092 - DPCOS | Basic External Function DCOS | II- C-45 |
| 094 - TANGH | Basic External Function TANH | II- C-46 |
| 095 - SQROT | Basic External Function SQRT | II- C-46 |
| 096 - DSQRO | Basic External Function DSQRT | II-C-47 |
| 097 - CSQRO | Basic External Function CSQRT | II- C-47 |
| 098 - ARCTG | Basic External Function ATAN | II- C-48 |
| 099 - DACTG | Basic External Function DATAN | II- C-48 |
| 100 - ACTG2 | Basic External Function ATAN2 | II- C-49 |
| 101 - DATN2 | Basic External Function DATAN2 | II- C-49 |
| 102 - DMODA | Basic External Function DMOD | II- C-50 |
| 103 - CABSA | Basic External Function CABS | II-C-50 |
| 110 - BSFTS | Statement Functions - Integer and Real | II- C-51 |
| 111 - FSFTS | Statement Funct. - D.P., Complex, Logical | II-C-51 |
| 140 - CPXAD | Addition and Subtraction of Complex | II- C-52 |
| 141 - CPXMU | Multiplication of Complex Numbers | II- C-52 |
| 142 - CPXDV | Division of Complex Numbers | II-C-53 |

## Table of Contents

| 143 | , |
| :---: | :---: |
| 144 | - CPXOP |
| 145 | - CREAD |
| 146 | - CREMU |
| 147 | - CREDV |
| 148 | - CREOP |
| 149 | - MISC3 |
| 150 | - MISC4 |
| 160 | - BRFCP |
| 161 | - BIFCP |
| 162 | - FRFCP |
| 163 | - FIFCP |
| 164 | - CFCCP |
| 165 | - DPFCP |
| 166 | - BFCCP |
| 167 | - SBRTN |
| 168 | - FSBRT |
| 169 | - BLKDT |
| 179 | - BLKDA |
| 180 | - UNFRW |
| 182 | - BACUP |
| 190 | - DOTRM |
| 191 | - DOLMT |
| 192 | - DONSC |
| 193 | - DONSI |
| 194 | - DONSX |
| 195 | - DONML |
| 196 | - DONIO |
| 197 | - MORDO |
| 200 | - SUBRI |
| 300 | - LOGIF |
| 301 | - BARIF |
| 302 | - FARIF |
| 310 | - IOFMT |
| 312 | - RDFMT |
| 350 | - MISC5 |
| 351 | - FUNMX |
| 352 | - NAMES |
|  | S |

Exponentiation of Complex Numbers
Arithmetic Operations on Complex
Add and Subtract Coinplex and Real Numbers
Multiply Complex by Real Numbers
Divide Complex by Real and the Reverse
Combined Operations on Complex and Real
Blanks in, Cont. of Statement to Max Lines
Special Characters for Continuations
Real External Functions
Integer External Functions
Real External Functions
Integer External Functions
Complex External Function
Double Precision External Functions
Logical External Functions
Subroutine Subprogram
Subroutine Subprogram
BLOCK DATA Test
BLOCK DATA Test
Unformatted WRITE and READ
BACKSPACE Tape
DO Loops - Terminal Statements
DO Loops - Parameters as Variable Names
DO Loops - Completely Nested Nest
DO Loops - Incomplete DO, Exit by GO TO
DO Loops - Extended Range
D0 Loops - Nested Nest
DO Loops - I/O Terminal Statements
DO Loops - I/O, Statmt. Ft., Intr Ft., CALL
Subroutine - Operations Done at Sub Level
Logical IF Statements
Arithmetic IF Statements - Integer, Real
Arithmetic IF Statements - D.P.
Formatted READ/WRITE - Additional Features
Formats in Arrays
Specifications for Program Form
Basic External Functions - Trig Formulae
Names Resemble FORTRAN Verbs, Functions
COMMON, DIMENSION, EQUIVALENCE

Page

II-C-53
II-C-54
II-C-54
II-C-54
II-C-54
II-C-55
II-C-55
II-C-55
II-C-55
II-C-56
II-C-56
II-C-56
II-C-56
II-C-57
II-C-57
II-C-57
II-C 58
II-C-58
II-C-59
II-C-59
II-C-60
II-C-60
II-C-60
II-C-60
II-C-60
II-C-61
II-C-61
II-C-61
II-C-62
II-C-62
II-C-62
II-C-63
II-C-63
II-C-63
II-C-65
II-C-66
II-C-67
II-C-67
II-C-67

A1. Objective: To develop a set of FORTRAN test programs, available to a wide range of FORTRAN processors with a minimum of user effort required to perform the tests. These tests shall conform to the ASA FORTRAN Standard X3.9-1966.[1]

## A2. Design Criteria

a) To constrain all test programs to the FORTRAN language described in the ASA FORTRAN Standard X3.9-1966.
b) To reduce the effect of those areas in which the FORTRAN standard does not prescribe a method or solution; the programs must be adaptable to differing environments such as:

- Size of computer and I/O facilities.
- Power of the FORTRAN compiler as reflected in the size and complexity of a FORTRAN program.
- Variations in the range and precision of numeric values.
- Differences in form and media for submitting a program and data.
- Differences in procedures for compiling and running a FORTRAN program.
c) To simplify the use of the FORTRAN test programs.
- The cost of computer time for compiling and running must be kept to a minimum.
- The cost of human resources for the analysis of test results, determination of test failures and the comprehension of the test design must be taken into consideration in the system design.
d) To test FORTRAN language elements before they are used in support of other tests.
e) To maintain an open ended system so that tests may be changed or added.

A3. Design Considerations
It is recognized that any set of programs which is designed to test a complex set of specifications, such as ASA FORTRAN (X3.9-1966) can never test every interaction of every FORTRAN statement, with all permissible forms, in all permissible positions in an executable program. However, it is desirable to design a system such that those parts of the Ianguage which have been tested are relatively easy to determine and at the same time permit extensions to the system without extensive knowledge of the entire system.

The test programs must be designed with the realization that a FORTRAN processor might not accept various elements of the language and the action could be identified at one or more of the following times or conditions:
a) Compile time.

- The compiler might terminate without completion of the compilation and with insufficient information for the user to determine the cause.
- The compilation may be completed with diagnostic messages on the program listing, which as a general rule (although outside of the FORTRAN standard specification) assist in locating the trouble.
b) Link Edit and Load Time.
- The executable program may fail to meet the loader, etc., requirements-which may or may not be identified in the program listing.
c) Execution Time.

Conditions in the computer or compiler may produce improper machine code which causes the test program to be aborted before completion. (Any one or more of these conditions could occur prior to obtaining the test program results.)
d) Unexpected Test Results.

The running of the test programs could produce printed results which were different from the expected results. This can occur if:

- Some well defined element of the Standard FORTRAN language was implemented in the compiler in a variant way.
- Some ill defined part of the language was interpreted by the compiler writer different from the test program writers.
- An improper interpretation of the standard by the test program writers.
- An actual bug in the test programs.
- An actual bug in the compiler.

Because many unforseen difficulties can occur during the running of the test programs, where it will be necessary for the user to refer to the program listing to determine what elements of the language are being tested as distinct from those elements which must be used to support the test, it is imperative that the program listing be liberally interspersed with FORTRAN comment lines to assist the reader.

Because the FORTRAN standard document is a semi-technical specifications document without a rigid definition of the semantics of the language, the document is subject to interpretation differently by different individuals.

The ASA FORTRAN Standard is a reference standard and does not address the medium or its coded characteristics, so that the form of the FORTRAN program on a medium such as punched cards is outside the scope of the standard. However, because a common medium is punched cards, and the H-set punch card code was designed for FORTRAN, the H-set is deemed the most universally
accepted card code on which to prepare the FORTRAN test programs and data for a processor. If a processor does not accept this card code it is reasoned that a conversion routine probably does exist which could convert this set to the processor punch card code.

If the programs are to be available to both large and small FORTRAN processors the I/O facilities must be kept to a minimum. If the processor has a card reader then most likely a printer and either one tape unit or a disc would also be available, so that the test programs could be confined to these I/O devices.

In order to determine what capabilities existed for FORTRAN or FORTRAN-like compilers in 1966 when this project was initiated, a survey of the literature was made and specifications for forty compilers were compared. From this unpublished study, a "FORTRAN processor" was defined to contain the minimum range and precision of numeric values and the most limited program size which could be found among the forty compilers examined. This lead to the constraints used in the test programs which are described under Program Information Section I-D.

The assumption was made, because of the nature of FORTRAN, that all processors probably had something akin to "Compile-Load-and Go" as a form of operations.

Each test program, if it were to run on a small computer, must be limited in size. It is theoretically possible to test almost all characteristics of the language in a single executable program if a processor were large enough. However, it might be desirable to test a new compiler on a large computer for the first time with small test elements, so that any difficulties might be recognized more rapidly, while any later running of the test programs or updated versions of the compiler could be performed more economically if the test elements were combined into larger executable programs.

In order for the test units to be run independently and later combined into larger executable programs, as well as changed or expanded it was necessary to consider the following:

- The required positioning of certain FORTRAN statements such as specification statements and statement functions.
- The choice of symbolic names, such that they did not constrain the testing of elements of the language, and at the same time would not require the knowledge by the user of symbolic names which had been used when changes or expansion of the test were necessary.
- The allocation of statement labels so that duplication would not result.
- The handling of those aspects of a FORTRAN program which are not covered by the standard such as precision, size of program, number of arguments, depth of DO nesting, the number of subprograms, etc.

The FORTRAN test programs are not designed for use in debugging of a FORTRAN compiler. In fact, the assumption has been made that the compiler, for the most part, is working but may not have all of the FORTRAN language features available in the system.

Those elements of the language which are used in support of test units are limited to what can be considered "defacto FORTRAN". That is, language features which were not universally implemented in 1966 but which appear in the standard are tested but are never used in support of other tests. Therefore, such features as: extended range of a DO, the Gw.d format field descriptor, a constant of the form 26E1 containing no decimal point, etc., are not used after their appearance in a test unit.

The test program units, for the most part, are small main programs with straight line logic. Each test unit is implemented to be run as a separate test or linked end to end with another test. All data used within a progra test unit is defined within that unit, except the tests for the FORMAT statement which require external input data to be read.

The selected order of the test units is dictated by the need for testing the basic fundamentals of the language so that these features may be used $t$ support later tests. Certain elements of formatted READ and WRITE are tested first, so that test results can be written out.

The initial test of the DATA statement appears as an early test sequence because a constraint would be placed upon the use of symbolic names in othe test units prior to the occurrence of the DATA statement test if the test appeared later in the set. Other appearances of the DATA statement are in a subprogram and as a format specification. These are for the purpose of the tests and no further use is made of this statement.

All testing is performed at the main program level except those concepts and associations which are unique to a subprogram. One test unit which is performed at the main program level containing a variety of FORTRAN statements is basically duplicated in a test unit which performs the same statements at the subprogram level. Other appearances of subprograms in the test set are basically for the purpose of argument association testing and for those FORTRAN statements which may occur only in a subprogram.

The FORTRAN statements used in the test units may appear, at first glance, to be nonsense operations. To comprehend the true meaning of the statement in a test unit, it is necessary to read the statement transformin the variable name or constant used into its attributes and utilization associated with an operator. Such an example might be: A one dimensional array element appearing in a common block is raised to the power of an unsigned integer constant.

To assist the test program implementors as well as the reader of the test programs, naming conventions described in Program Information,

$$
\mathrm{I}-\mathrm{A}-4
$$

Section I-D, have been used throughout the programs to convey the attributes of the name, which appear in specification statements, directly in the name itself, so that no reference need be made to the specification statements to comprehend this information. In addition, comment cards have been used freely in the test units to convey the nature of the test and the operations being performed.

The design of a computer program system for automatic insertion of operating system control cards and the linking of test program units was initiated. Further analysis into the problems has brought to light the potential difficulties of using the output of such a system and its doubtful economics. For the following reasons, this system has not been implemented:

- The lack of common terminology for similar functions among various operating systems control languages would cause difficulty in communicating with a wide audience the information required to be inserted into an automated system for producing the desired effect. For example, what is called a JOB card in one system is called a RUN card in another, while what is called a RUN card in another system may be called an EXECUTE card in the first system. Because similar terminology for operating system control functions is used for functions of the system at different levels of control, it would be necessary to describe levels of functions to a user, who might not be aware of this logic.
- The FORTRAN standard does not define the order of presentation to a compiler of program units, so that this becomes an additional burden to the user to comprehend when this order may not affect the majority of FORTRAN processors.
- Operating systems control cards may require special control punch codes which are outside the codes defined for data use. For example, a control card which contains a code containing the digits 6, 7, 8 and 9 in a single column on the card can be obtained only by a keypunch device with provisions for over striking in a single column.
- To produce punched cards from an automated system with special codes outside the normal punch card character set would require the software-hardware system to permit column binary cards to be punched. This facility, although available in the hardware of some systems is not available to the user because of software constraints. Of the computer systems surveyed, only one system permitted column binary cards to be produced and this facility is available only to the assembly language programming system.
- If cards can be produced by the colunn binary operations from the system, the device which interprets and prints on the card would not necessarily print the appropriate symbols, because codes for certain FORTRAN characters and the control card codes may have different graphic associations or no valid association.
- If the test programs with their interpersed operating systems control cards were placed on tape, there is no assurance that the receiving installation has provisions for using or even obtaining punched cards from such a tape. Although the images on tape would be in coded character representation for the receiving installation, the operating system may not permit the reading of control card sensitive information and the passing of it to an applications program for the purpose of producing punch cards. If the tape is read as a binary tape, the parity bit, record size, pulse code and blocking characteristics might not conform to the receiving computer's requirements.

Because of these numerous difficulties which may affect the user and the potential additional costs which may be encountered in preparing the test program information at the receiving installation, these test programs have been prepared for use without the inclusion of systems control cards.

To simplify the task of grouping test units together into larger programs for testing, and thus eliminate the need for an abundance of systems control cards to operate each test unit as a separate computer run, those cards which must be revised are identified in the test units as comment cards containing the characters " $\mathrm{C}=$ " in the first two columns. The FORTRAN specification statements taken from different test units require the elimination of duplicate names to conform to the language definition. To simiplify this task, symbolic names appearing in a "C=" specification statement will always appear in the same type of specification statement throughout the entire program test set, so that elimination of duplicate names is achieved by inspection of a collection of a similar type of specification statement. That is, if an array declarator in one program test unit is contained in an INTEGER statement, all other occurrences of that symbolic name in a specification statement will be in an INTEGER statement and not in a DIMENSION statement. See Structuring, Restructuring and Extending Test Programs, Section I-E.

A5. FORTRAN Concepts Excluded from the Test Programs.
Because the FORTRAN Clarification Reports [2, 3] do not have the status of updating the current ASA FORTRAN document X3.9-1966, extreme caution was exercised in making use of some of the interpretations in the FORTRAN test programs. The following FORTRAN Statements and concepts have been excluded from the FORTRAN Test Programs:
a) An I/O unit number specified by an unsigned integer constant. All I/O statements express the unit numbers as integer variable nanes which are assigned values in the first executable statements. This increases program portability.
b) PAUSE and PAUSE n. These statements are excluded from the test because many systems do not permit them and, action by an operator would be required to resume the program test.
c) The name of a Basic External Function specified as a user subprogram name. This action would not permit the inclusion of a Basic External Function so defined to appear in any test unit which was combined with other test units.
d) An external procedure written in a language other than FORTRAN. Unless Basic External Functions can be considered in this class of procedures no test is made of this facility.
e) As currently structured in Version 3, with test units 169 and 179 in Parts 11 and 12, respectively, a single labeled common block does not receive initialized data from more than one Block Data Subprogram. The proposed revised FORTRAN Standard tentatively places such a restriction upon Block Data Subprograms.

Combining these test units would test the ability to initialize data from more than a single Block Data Subprogram to a specified labeled common block. Individual data elements, however, are not initialized more than once.
f) Formatted and Unformatted records on the same I/O device within the same test unit. This concept is the subject of a FORTRAN clarification. Because a unit may be declared by the implementor not to contain this property, because this concept conflicts with the Magnetic Tape Label for Information Interchange Standard (X3.27-1969) and because this concept does not enhance program interchange, this feature was excluded from a single test unit. However, when test units 196 and 197 are combined in an executable program as in Version 3 Part 12 this feature is tested.
g) A Formatted external output field whose width does not contain enough character positions to include a positive sign and a leading zero. This concept is the subject of a FORTRAN clarification. Because these optional character positions are described in the FORTRAN Standard in the same paragraph which describes the optional external exponent form (implementor option), it is unclear whether the optional character positions are an implementor or a user option.
h) A subprogram name passed as an actual argument, and then a corresponding dummy name appearing in an argument list of a function reference or CALL to a lower level subprogram. The rules of the FORTRAN standard are incomplete. Because a dummy subprogram name may not appear in an EXTERNAL Statement it is unclear how a subprogram name may be passed more than one level and maintain a proper association as a subprogram.
i) A labeled FORMAT statement which is not referenced in an I/O statement. It is unclear in the FORTRAN Standard whether a standard conforming FORTRAN program may contain such a statement which is not referenced.
j) Hollerith constants are constrained to the FORTRAN character set, and therefore the character set is a subset of the characters capable of representation by the processor. This increases program portability.
k) The ENDFILE statement appears in a test unit but cannot be tested, because the action is undefined when an endfile record is encountered during execution of a READ statement.

A6. Interpretations Made to the FORTRAN Standard
The following interpretations have been made to the FORTRAN Standard:
a) Those items identified in the FORTRAN Clarification Reports as "Correction to Typographical and Transcription Errors" and "Corrections to Mistakes" in the FORTRAN document X3.9-1966 have been recognized and the interpretation to the standard is as if these items had actually been corrected in the original document.
b) A relational operator is not immediately followed by a signed constant. A left parenthesis appears between the relational operator and the signed constant. The FORTRAN standard does not appear to permit two adjacent operators.
c) Hollerith data does not appear "under the guise" of a complex or double precision type.
d) The word "range" may not be broadened to include "extended range" and therefore a GO TO or arithmetic IF statement in an "extended range" may not reenter the DO nest at a common terminal statement.
e) The FORTRAN Standard does not state how a Hollerith constant is positioned in a storage unit. In order for a Format Specification to be introduced into an array by way of a DATA Statement, the following assumption has been made based upon the Aw Format field descriptor, 'Let 'g' be the number of characters represented in a storage unit", and "w" be the value of $n$ in the $n H$ form of a Hollerith constant, then:" If the field width is less than $g$, the w characters will appear left justified with g-w trailing blanks in the internal representation [1, page 22L22].
f) There are no separate class rules for Basic External Functions and therefore referencing of these is handled under Class $V$, an external function. By these rules a Basic External Function may be passed as the actual argument of an external procedure reference provided the symbolic name appears also in an EXTERNAL statement.
g) The unit of angular measurement for the trigonometric functions is assumed to be expressed in radians.
h) "The value zero is considered neither positive nor negative", does not constrain the appearance as a constant to be an unsigned zero, but may appear with either a plus (+) or minus (-) sign.
i) The FORTRAN Standard does not describe the condition of non-nested DO loops contained in an outer DO loop, nor is this condition described in earlier FORTRAN implementation manuals. However, this concept is fundamental to the DO loop and is considered defacto.

In the following picture each bracket is considered to be a DO loop:

The FORTRAN Test Programs are made up from 185 segments containing sequences of FORTRAN statements. There are 116 main program sequences, whose segment number and name are each printed with the test program results, 63 subprograms which are each associated with a single test, and 6 sequences, one of which (segment 007) is always associated with each executable program. Elements from the other five sequences (segments $000,001,003,005,006$ ) are included when appropriate. The F(ORTRAN Test Programs Version 1 and 3 are structured to include the necessary elements from these segments. The following is a brief description of each segment:

000, , (non-executable) contains a Directory of Test Programs introduced by comment lines before the first executable program on Version 1. In Version 3, the Directory appropriate to each of the 14 parts is inserted before each of the 14 executable programs.

001, SPECS, (non-executable) declares variable types, function types, and array sizes and types for use in later segments of the test programs. This segment is not executable since it contains only specification statements, but statements from this segment are included in other segments, as required, to furnish the necessary specifications for an executable program.

003, DATA1, (non-executable) examines the format of the DATA statement, which causes variables and array elements to be initially defined. It is run with segment 010, DATA2.

005, BSFDF, (non-executable) defines arithmetic statement functions of type integer and real. Segment 005 is run with segment 110 and 197.

006, FSFDF, (non-executable) defines statement functions of type double precision, logical, and complex.

The expressions contain constants, variables and intrinsic function references, references to previously defined statement functions and to external functions. Segment 006 is run with segment 111.

007, IODEF, (included in all executable programs) defines the system input, system output and a work unit to be used in the testing programs. Three integer variables are given values in simple assignment statements, to be associated with those units, which must be included with each program that requires such definitions. However, the values assigned to these variables may be changed to satisfy specific computer systems. These units are referred to by the following variable names:

NUVI - for results, usually a line printer defined as unit 6.
IRVI - for input, usually a card reader defined as unit 5.
INVI - for intermediate input/output data, usually a magnetic tape defined as unit 9. This unit is used only in Segments $180,182,196,197$, and 200.

In Versions 1 and 3, 6 input cards (three of which are prepared by the user) are associated with this segment but run with segment 008. In Version 3 also each one of the executable programs (14 Parts) contains these cards so that the user can identify the environment of the execution of the tests.

008, FMTRW, (executable) tests the FORMAT and formatted I/O statements. Under control of the FORMAT statements in the segment, 40 data cards are read in from the system input unit, and written to the system output unit. The reading into and writing from a FORMAT specification as well as the symmetry of the terminal slash in a FORMAT specification is inserted into segment 007 but executed as part of segment 008. Also written to the output unit, are lines of data produced by Hollerith information showing how the data should appear. Additional tests are performed in Segment 310.

009, AFRMT, (executable) tests FORMAT and formatted I/O statements as related to A-conversion. It tests that the Aw descriptor causes w Hollerith characters to be read into or written from a single list item, provided w does not exceed the number of characters representable in a single storage unit. The last line of the test results should print the last letters of the alphabet equal to the number of Hollerith characters contained in a storage unit. If the number of characters is less than 4, the first three test lines will contain missing characters, but the corresponding Hollerith information should be aligned.

010, DATA2, (executable) tests the contents of variables and array elements which were initialized by way of the DATA statement, in segment 003. Via formatted output, the contents of the initialized variables and array elements are written out. The values are integer, real, double precision, complex and Hollerith. The FORMAT statements are varied, and contain descriptors, repeated by parentheses and constants.

011, AASGN, (executable) tests simple arithmetic assignment statements with the formation of integer and real constants.

013, DASGN, (executable) tests the formation of double precision constants, the referencing of double precision array elements and the assignment of values to this type in arithmetic assignment statements. The proper application of the unary sign to double precision is also tested.

015, CASGN, (executable) tests the formation of complex constants, the referencing of complex variables and array elements and the assignment of values to this type in arithmetic assignment statements. The proper application of the unary sign to complex types is also tested.

016, LASGN, (executable) tests logical assignment statements. Values are assigned to integer variables used in relational expressions of logical assignment statements. Variables and array elements are declared logical in type statements, then used in mixtures of relational expressions and logical expressions which are assigned to variables and array elements. Logical values are either true or false.

017, INTRL, (executable) tests arithmetic assignment statements in which each side of the equation is of a different type. Integer values are assigned to real and double precision variables and arrays; real values are assigned to integer and to double precision variables and arrays.

020, UGOTO, (executable) tests the unconditional GOTO statement. Branching into labeled executable statements, in both a forward direction and a backward direction and to statements immediately following the GOTO. Each set of statements causes an integer to be generated. The test is designed to cause the unconditional transfers to be executed in such an order as to produce a consecutive set of integer values.

021, AGOTO, (executable) tests the GOTO assignment statement. The integer variable used in an ASSIGN statement is referenced only in an assigned GOTO statement, while defined as a statement label. Assigned GOTO statements branch only to excutable statements; they have a maximum of nine branches, though the ANSI standard does not specify a maximum. The value of the integer variable after the execution of the ASSIGN statement is designed to correspond to a statement label in the list of the assigned GOTO statement.

022, CGOTO, (executable) tests the computed GOTO statement. Lists in the statements have nine or fewer statement labels, which are within the same program unit. The integer variable referenced is always greater than zero and does not exceed the number of statement labels in the list.

030, 031, 032, 033, examine the formation of expressions with the addition or subtraction operator. Expressions involve variables, array elements and constants in varying orders, such as:
> variable $\pm$ array element $\pm$ constant variable $\pm$ constant
> array element $\pm$ constant
> array element $\ddagger$ variable.

In each of these segments, numeric values are assigned to the variables and array elements which are then referenced in simple arithmetic statements.

030, ARBAD, (executable) forms expressions in which real values or integer values are added together. Expressions contain two to eight terms. One expression contains only variables, one contains only array elements and an other contains only constants.

031, ARFAD, (executable) combines double precision values with the addition operator. Values are positive or negative variables and array elements. Two, four or five terms make up each expression.

032, ARBSB, (executable) forms expressions in which real or integer values are subtracted. Values are positive or negative variables and array elements. Expressions contain two to four variables, array elements and constants.

033, ARFSB, (executable) examines expressions involving the subtraction of double precision values. Values are positive and negative. Elements are variables, array elements and constants. Statements contain two to four variables, array elements and constants.

034, ARBAS, (executable) combines both addition and subtraction in expressions containing real or integer values. Variables, array elements and constants appear in various combinations and orders. Numeric values which are assigned are positive and negative. Expressions contain two to six elements.

035, ARFAS, (executable) combines subtraction and addition in expressions with double precision values. Some expressions contain parenthesized expressions within parenthesized expressions, others contain variables, array elements and constants without parentheses.

036, ARBMI, (executable) tests the multiplication of integer values, which are both positive and negative. One to six multiplication operations occur within a single expression.

037, ARBMR, (executable) tests the multiplication of real values. Expressions contain two to seven terms. Values are positive and negative.

038, ARFMD, (executable) tests expressions which involve the multiplication of double precision values. Variables, array elements and constants occur in various orders in expressions which contain from two to seven terms.

039, ARBDV, (executable) tests expressions of type real or integer in which variables and constants are divided by variables and constants. Some expressions contain successive division operations, in order to examine the order of evaluation of the terms.

040, ARFDV, (executable) tests the division of double precision variables, array elements and constants. Within an expression, values are of the same type and divisors are never zero. Expressions contain one to four division operations.

041, ARBEX, (executable) tests expressions in which integer or real values are raised to integer or real powers. The exponent assumes values which include zero and a negative one. Successive exponentiation occurs in some expressions so that the order of evaluation might be examined.

$$
\begin{aligned}
& A * * B \\
& (A * * B) * * C \\
& (A * * B) * *(C * * D)
\end{aligned}
$$

042, ARFEX, (executable) tests expressions in which double precision values are raised to real and double precision powers. Exponentiated values are raised to exponentiated values. Expressions contain variables, array elements, and constants.

043, ARBHI, (executable) tests the hierarchy of operators and parentheses. Only integer expressions are used in this segment which also tests that the laws of association and commutation may be applied. Integer terms containing division, do not follow these laws. The order of evaluation, generally, is according to the following hierarchy:

1. exponentiation
2. multiplication/division
3. addition/subtraction.

The elements of the expressions are then regrouped, using parentheses, to cause new orders of evaluation.

050, SBB67, (executable) tests the formation of subscripts for integer and real arrays, where the form of the subscript is either an integer variable, $v$, or an integer constant, $k$. Arrays are one, two or three dimensions, and the variables in the subscripts are given values in simple arithmetic assignment statements.

051, SBB45, (executable) tests the formation of subscripts for integer and real arrays, where the form of the subscript is either a variable plus a constant, $v+k$, or a variable minus a constant, $v-k$. Expressions also contain array elements with constant subscripts. Variables and constants in subscripts are of integer type.

052, SBB13, (executable) tests the formation of subscripts for integer and real arrays where the form of the subscript is a variable multiplied by a constant, $c^{*} v$, or a variable multiplied by a constant plus a constant, $c^{*} v+k$, or a variable multiplied by a constant minus a constant, $c^{*} v-k$. Through simple arithmetic statements, real and integer values are assigned to variables and array elements. Integer values are assigned to the variables occuring in subscripts of array elements, which are then computed; the array elements are then used in the evaluation of the expression in which they occur.

053, SBF17, (executable) tests the formation of subscripts for double precision arrays using the allowable subscript constructs: v, k, v+k, $v-k, c^{*} v, c^{*} v+k, c * v-k$, where $c$ and $k$ are integer constants and $v$ is an integer variable. Arrays are one, two or three dimensional; subscript expressions are of integer type and the values assigned to array elements are of double precision type.

054, SIMIF, (executable) tests simple forms of expressions in an arithmetic IF statement and a logical IF statement followed by a GOTO, so that these statements may be used in subsequent tests, the logical IF is further tested in segment 300, and the arithmetic IF in segments 301 and 302 .

055, IFABS, (executable) references the intrinsic functions, ABS, and IABS, which obtain the value of the argument, disregarding the sign. The arguments are integer, real variable names, and expressions.

056, IFFLT, (executable) references the intrinsic function, FLOAT, which is to convert an integer to the real form. Arguments are integer variable names and expressions.

057, IFFIX, (executable) references the intrinsic function, IFIX, which is to convert a real value to the integer form. Arguments are real variable names and expressions.

058, IFSGN, (executable) references the intrinsic functions, SIGN and ISIGN which are to transfer the sign of the second argument to the first argument. Arguments are integer or real variable names or expressions.

059, IFDAB, (executable) references the intrinsic function, DABS, which obtains the value of a double precision argument, disregarding the sign. Arguments are double precision variable names and expressions.

060, IFTRN, (executable) references the intrinsic functions, AINT, INT, and IDINT which are to truncate real and double precision values. Arguments are variable names.

061, IFMOD, (executable) references the intrinsic functions AMOD and MOD, defined as remaindering. The arguments are real and integer variables, respectively.

062, IFMAX, (executable) references the intrinsic functions AMAXO, AMAX1, MAXO, MAXI, DMAX1, which are to choose the largest argument of a set of arguments. Arguments are real, integer, and double precision variables. There are two to five arguments in each argument list, though the ANSI standard does not set a limit on the number of arguments.

063, IFMIN, (executable) references the intrinsic functions AMIN0, AMIN1, MIN0, MIN1, DMIN1, which are to choose the smallest value of a set of arguments Arguments are integer, real, or double precision variables. There are two to five arguments in each list.

064, IFDSG, (executable) references the intrinsic function DSIGN, which is the transfer of sign from the second argument to the first. The two arguments are double precision variables.

065, IFDIM, (executable) references the intrinsic functions DIM and IDIM which are to obtain the positive difference. Arguments are real and integer variables, resp.

066, IFSGL, (executable) references the intrinsic function SNGL, which is to obtain the most significant part of a double precision value. Arguments are variables and expressions. The first and the last result should be the same value.

067, IFREL, (executable) references the intrinsic function REAL which is to obtain the real part of a complex quantity. Arguments are variables.

068, IFIMG, (executable) references the intrinsic function AIMAG, which obtains the imaginary part of a complex value. Arguments are constants and variables.

069, IFDBL (executable) references the intrinsic function DBLE, which expresses a single precision argument in double precision form. Arguments are variables and intrinsic function references.

070, IFCPX, (executable) references the intrinsic function CMPLX, which is to form a complex value from two real arguments. Arguments are constants and variables.

071, IFCJG, (executable) references the intrinsic function CONJG, which is to obtain the conjugate of a complex value. Arguments are constants and variables.

072, IFBMS, (executable) tests the use of arithmetic expressions of several terms or containing references to intrinsic functions as arguments to other intrinsic functions.

073, IFFMS, (executable) references many of the intrinsic functions. The arguments to them consist of all the primaries.

080, EXPON, (executable) references Basic External Function, EXP, the exponential function of type real. The arguments which are powers of 2 , are real variables and expressions containing intrinsic functions.

The expected results printed to a precision greater than the computed results in the Basic External Function tests, are obtained from Table values.[4]

081, DEXPO, (executable) references Basic External Function, DEXP, the double precision exponential function. Arguments are powers of 2, ranging from -16.0D0 to +16.0D0. Some arguments are expressions containing intrinsic functions.

082, CEXPO, (executable) references Basic External Function, CEXP, the complex exponential function. The testing range extends from 0 to 16 by steps of PI/3.

083, LOGTM, (executable) references Basic External Function, ALOG, the natural logarithm function of type real. Arguments are real variables and expressions containing intrinsic functions.

084, DPLOG, (executable) tests Basic External Function, DLOG, the double precision natural logarithm function. Arguments are double precision variables and expressions containing intrinsic functions.

085, CXLOG, (executable) references Basic External function, CLOG, the complex logarithm function. The testing range extends from 0 to $5 . E 7$ by steps of PI/3.

086, COLOG, (executable) references Basic External Function, ALOG10, the common logarithm function of type real. Arguments are real variables and expressions containing intrinsic functions.

087, DCLOG, (executable) references Basic External Function, DLOG10, the double precision logarithm function. Arguments are double precision variables and expressions containing intrinsic functions.

088, SINUS, (executable) references Basic External Function, SIN, the trigonometric sine function of type real. The arguments which range from 0 to 2 PI , are real variables and expressions containing intrinsic functions.

089, DPSIN, (executable) references Basic External Function, DSIN, the double precision trigonometric sine function. The arguments which range from 0 to 2 PI are double precision variables and expressions containing intrinsic functions.

090, CSICO, (executable) references Basic External Functions, CSIN and CCOS, the complex trigonometric sine and cosine functions. Arguments are complex variables.

091, COSNS, (executable) references Basic External Function, COS, the trigonometric cosine function of type real. The arguments range from 0 to 2 PI , and are real variables and expressions.

092, DPCOS, (executable) references Basic External Function, DCOS, the trigonometric cosine function of type double precision. Arguments are double precision variables and expressions which range from 0 to 2 PI.

094, TANGH, (executable) references Basic External Function, TANH, the hyperbolic tangent function of type real. Arguments are real variables and expressions containing intrinsic functions.

095, SQROT, (executable) references Basic External Function, SQRT, the square root function of type real. Arguments are real variables and expressions whose values are prime numbers.

096, DSQRO, (executab1e) references Basic External Function, DSQRT, the double precision square root function. Arguments are double precision variables and expressions whose values are prime numbers.

097, CSQRO, (executable) references Basic External Function, CSQRT, the complex square root function. Arguments are complex expressions.

098, ARCTG, (executable) references Basic External Function, ATAN, the trigonometric arctangent function of type real. Arguments are real variables and expressions containing intrinsic functions and whose values are powers or sums of 2 .

099, DACTG, (executable) references Basic External Function, DATAN, the single argument trigonometric arctangent of type double precision. Arguments are real variables and simple arithmetic expressions containing intrinsic functions, whose values are powers or sums of 2 .

100, ACTG2, (executable) references Basic External Function, ATAN2, the two argument trigonometric arctangent function of type real. Arguments are real variables and expressions containing intrinsic functions, whose values are powers or sums of 2 .

101, DATN2, (executable) references Basic External Function, DATAN2, the two argument trigonometric arctangent function of type double precision. Arguments are double precision variables and expressions containing intrinsic functions, whose values are powers or sums of 2 .

102, DMODA, (executable) references Basic External Function, DMOD, the remaindering function of type double precision. Arguments are double precision variables.

103, CABSA, (executable) references Basic External Function, CABS, the modulus function. Arguments are the elements of an array of type complex.

110, BSFTS, (executable) references statement functions defined in an earlier segment, 005. The arguments are integer or real constants, variables and arithmetic expressions. Type statements are used to reaffirm the type of some intrinsic functions.

111, FSFTS, (executable) references statement functions in which the arguments are logical, double precision or complex constants, variables, and logical or arithmetic expressions. The statement functions were defined in segment 006. Type statements are used to reaffirm the type of some intrinsic functions.

140, CPXAD, (executable) tests expressions in which complex values are added or subtracted. Complex variables and constants occur in various orders and combinations, with two to nine elements in each expression.

141, CPXMU, (executable) tests expressions in which complex values are multiplied by complex values. Expressions contain from two to ten terms in various orders and combinations of complex variables and complex constants.

142, CPXDV, (executable) contains expressions in which complex values are divided by complex values. Variables and constants appear both as dividends and divisors. Some expressions involve only complex variables, some only complex constants, and others a combination of both.

143, CPXEX, (executable) involves the exponentiation of complex values. The value of the integer power varies from 3 to 100. Expressions contain variable and constant values raised to variable or constant powers. Each expression contains a single term.

144, CPXOP, (executable) performs several arithmetic operations within an expression containing complex values. Each of the arithmetic statements performs addition, subtraction, multiplication, division, and exponentiation. Only the exponents are of integer type.

145, CREAD, (executable) performs addition and subtraction within an expression containing complex and real values. Other than in exponentiation, complex values may only be combined with real values.

146, CREMU, (executable) performs multiplication of real and complex values within an expression. The number of terms in an expression varies from two to four.

147, CREDV, (executable) performs division of complex values by real values and of real values by complex values. Expressions contain terms in which values are variables or constants.

148, CREOP, (executable) performs, within an expression, addition, subtraction, multiplication and division of complex and real values, and exponentiation of complex values. Exponents are integer values, only. The hierarchy rules determine the order of evaluation.

149, MISC3, (executable) contains arithmetic assignment statements in which the statements are continued for several lines and are interspersed with many blanks. Blanks occur within variable names and throughout the statements which are one to twenty lines in length. The statements involve real and integer values only. The digits, the letters and the special FORTRAN characters make up the list of continuation characters for the multiple line statements. The digit, zero, and the character, blank, are not legitimate continuation characters, but are used in the initial line of a statement.

150, MISC4, (executable) has interspersed blanks within arithmetic assignment statements containing complex values. Statements are one to twenty lines with letters and special characters to indicate the continuation. Statements occur which have a single character on a line; others have one or two terms of the expression on a line. Uncounted blanks do not appear in the midst of Hollerith information. Continuation lines of both a FORMAT statement and an assignment statement contain non space characters in columns 2 through 5. The arithmetic assignments used in this segment are similar to those used in segment 148.

160 , BRFCP, (executable) references REAL functions, contained in segments 400 , $420,430,440,450,460$. The arguments of the functions are either integer or real variable names, array names, array element names, and arithmetic expressions. Arguments are given numerical values in arithmetic assignment statements, and their names, values, or expressions appear in the argument list of the function reference. Function references contain one or two arguments in the argument list with only one list containing many arguments

161, BIFCP, (executable) references INTEGER functions contained in segments 401, $421,431,441,451,461$. Arguments are integer or real variable names, array names, array element names and arithmetic expressions. Argument lists contain as few as one argument and as many as twenty arguments, though no limit is imposed by the ANSI standard. The expression in which the references occur are of the same type as the function value to be returned.

162, FRFCP, (executable) references REAL functions; the arguments are the types integer, real, double precision, complex and logical, and are variable names, array names, array element names, and external procedure names. The functions referenced are contained in segments 402, 422, 432, 442, 452. Reference is also made to two intrinsic functions, REAL and AIMAG, which return the real part and the imaginary part of complex values, resp. to the expressions in which they occur. Common storage is shared by the referencing program and a function.

163, FIFCP, (executab1e) references INTEGER functions with arguments of types integer, real, double precision, complex and logical. Variable names, array names, array element names and external procedure names appear in the argument lists. Common storage is shared by the referencing program and a function. The functions referenced are in segments $403,423,433,443,453$. One argument list contains twentyone arguments; all others contain one or two arguments.

164, CFCCP, (executable) references COMPLEX functions with arguments of types integer, real, double precision, complex, and logical. The argument lists include variable names, array names, array element names and external procedure names. The functions referenced are contained in segments $404,414,424,434,444,454,464$. Common storage is shared by the referencing program and a function.

165, DPFCP - (executab1e) references DOUBLE PRECISION functions with arguments of types integer, real, double precision, complex and logical. Variable names, array names, array element names, and external procedure names appear in the argument lists. Common storage is shared by the referencing program and a function. The functions referenced are in segments $405,415,425,435,445,455,465,475$. These functions return a value which is of the same type as the expressions in which they occur within the calling program.

166, BFCCP, (executable) references LOGICAL functions with arguments of types integer, real, double precision, complex and logical. The argument lists include variable names, array names, array element names, and external procedure names. Referenced functions are in segments 406, 416, 426, 436, $446,456,466,476$; the value of the function returned from each reference is of type logical. Common storage is shared by the referencing program and a function.

167, SBRTN, (executab1e) calls subroutine subprograms. Arguments are the types integer and real and include variable names, array names, expressions and a Basic External Function. A CALL from a subroutine is made to another subroutine. One subroutine CALL contains no argument list. Subroutines called are in segments 407, 417, 427, and one of them shares common storage with the calling program.

168, FSBRT, (executable) calls subroutine subprograms. Arguments are the types integer, real, double precision, complex and logical and include variable names, array names, and expressions. A CALL from one subroutine is made to another subroutine; one subroutine CALL contains no argument
list. Subroutines called are in segments $408,418,428$, and share common storage with the calling program. Values are returned via the argument list of the CALL.

169, BLKDT, (executable) uses a block data subprogram. Labeled common blocks contain variable names and dimensioned arrays. Implicit types of variables and arrays are overridden by double precision, complex and logical statements. The block data subprogram used to suppiy the initial values of the labeled common blocks is contained in segment 409. This segment writes out the values which are contained in the labeled common blocks.

179, BLKDA, (executable) uses three block data subprograms, which contain six labeled common blocks with elements to be initialized. Elements of any block are initialized through only one of the block data subprograms contained in segments 419, 429, 439. Implicit typing is sometimes overridden by double precision, complex, and logical statements. This segment writes out the values which are contained in the labeled common blocks. They correspond to the labeled common blocks of the block data subprogram.

180, UNFRW, (executable) tests the unformatted WRITE statement and the unformatted READ statement with and without a list. Included in the segment is an ENDFILE statement. This segment uses an intermediate tape.

182, BACUP, (executable) examines the backspace statement. Data is created in memory, written to tape, then changed in memory. The tape is then backspaced, and the data read to memory in a forward direction. The data block is 1024 words in length and is written and read by way of unformatted input/output statements. This segment uses an intermediate tape.

190, DOTRM, (executable) examines DO statements and DO ranges which terminate with a CONTINUE, ASSIGN, or logical IF statement. DO statements meet the requirements that parameters of the DO must be greater than zero, and must not be redefined during the execution of the range of that DO. In some DO statements, the incrementation parameter appears; in others, it does not appear and has an implied value of one.

191, DOLMT, (executable) examines a DO statement and its range, in which the parameters are integer variable names. Numerical values are given for them in arithmetic assignment statements occurring before the DO statement. The DO range consists of an arithmetic assignment statement involving the induction variable and the terminal statement which is a CONTINUE.

192, DONSC, (executable) examines DO ranges contained within other DO ranges, the parameters of which are integer constants and variables. Each range of a DO within the nest has its own terminal statement. Another nest of DO's has a single terminal statement. Nests contain two to five DO statements and the DO range includes arithmetic IF statements and GO TO statements。

193, DONSI, (executable) examines a DO statement and its associated range, in which an exit is made from the range of a DO before the DO has been satisfied. The induction variable is used both within and outside of the range of the DO.

194, DONSX, (executable) examines a DO nest which has an extended range. Exit from the innermost DO is by way of an unconditional transfer, reentry is by way of an arithmetic IF statement.

195, DONML, (executable) examines the ranges of DO's which are within the range of another DO, but are not nested. All parameters are integer constants and the ranges contain arithmetic assignment statements.

196, DONIO, (executable) examines the ranges of DO's which have input or output statements as the terminal statement. The terminal statements include a READ, a REWIND and a WRITE statement, each of which is also the only statement within the range of that DO. This segment uses an intermediate tape.

197, MORDO, (executable) examines DO ranges which have within, references to statement functions and intrinsic functions, CALLS to subprograms, and DO's with extended ranges. Input, output and rewind statements are also within these DO ranges. This segment uses an intermediate tape.

200, SUBR1, (executable) passes the I/O assignments through common then calls a subroutine subprogram without an argument list, and returns to an unlabeled CONTINUE statement. The subroutine called is contained in segment 410. This segment uses an intermediate tape.

300, LOGIF, (executable) examines the logical IF statement. Variables and array elements, declared logical, are assigned values of true or false. These values are then used in the logical IF statement, which includes an executable statement which is not a DO statement nor an other logical IF statement. A signed zero constant is tested in a relational expression.

301, BARIF, (executable) examines the arithmetic IF statement which contains integer or real values and references to intrinsic functions. The effect of the sign of zero is tested.

302, FARIF, (executable) examines arithmetic IF statements in which the expressions contain double precision values and references to intrinsic functions.

310, IOFMT, (executable) examines the formatted READ and WRITE statements and FORMAT statements as they relate to fields of input card images. There are 38 card images read as input to this segment; the formats under which the variables and array elements are read and written include each of the descriptors. Formats occur in which there is a one to one correspondence between elements in the list and descriptors; other formats occur which do not have the same number of descriptors as there are elements in the lists. Segment 310 examines additional features not contained in segment 008 .

312, RDFMT, (executable) examines formatted READ and WRITE statements in which the format specifications are contained within arrays. Reference is to an array name, in place of the reference to a format statement label. The format specifications contained in arrays do not have nH field descriptors. FORMAT specifications are defined in DATA statements, read in as elements of an array, and passed as an argument to a subroutine. There are 13 card images read in this segment.

350, MISC5, (executable) examines the specifications of the program form. This includes verifying that comments are not executed, that every statement within the unit, can be reached, that all characters in a line are accepted, that labels can be one to 5 characters long and may be placed anywhere in columns one to five. Other features of program form are also examined.

351, FUNMX, (executable) further tests some Basic External Functions by using trigonometric formulas.

352, NAMES, (executable) determines whether the compiler can distinquish predefined function names and data names from FORTRAN verbs. The names of intrinsic functions and FORTRAN verbs appear as variable names and array names in a program unit. In other units of the same program, these names appear as intrinsic function names and as FORTRAN verbs. Subprogram units are segments $413,463,473,483$.

360, SPEC2, (executable) examines the use of integer variables and arrays and real variables and arrays, which are either in COMMON, or appear in EQUIVALENCE statements, or both. All array names are in DIMENSION statements; some have two or three dimensions, which are written as one dimensional arrays in the EQUIVALENCE statement. The array element successor function defines a relation by which a multi-dimensional array can be made equivalent to a one dimensional array. The order of the specifications is DIMENSION, COMMON, EQUIVALENCE and no dummy arguments appear in COMMON or EQUIVALENCE statements. Numeric values are assigned to variables and array elements to which other variables and array elements have been equivalenced. The associated variables and array elements are then used in arithmetic assignment statements, DO statements, IF statements and computed GOTO statements. A special blank common arrangement is used in this segment and this segment may not be combined with other segments requiring blank common.

Segments beginning with segment 400 are subprograms.
400, AFS, to be run with main program segment 160 , is a real function of one real argument. The value of the function is the result of multiplying the dummy argument by a constant.

420 , BFS, to be run with main program segment 160 , is a real function of two real arguments which are added together to produce the value of the function.

430, CFS, to be run with segment 160 , is a real function of an integer argument, which is the power to which a constant is raised, to produce the value of the function.

440, DFS, is a real function of two integer arguments, one of which is subtracted from the other producing the power to which a real constant is raised. The result is the value of the function. This function is referenced in segment 160.

450, EFS, is a real function of a real array, the size of which is declared in a DIMENSION statement. The value of the function is the sum of the elements of the array. This function is referenced in segment 160 .

460, FFS, is a real function with twenty arguments of integer and real variables and integer and real arrays. The expression defining the function contains addition, subtraction, multiplication and exponentiation of arguments. This subprogram is referenced in segment 160.

401, IAFI, is an integer function of a single real argument. The value of the function is the product of a constant and the argument. This subprogram is referenced in the main program contained in segment 161.

421, IBFI, is an integer function of two real variables. The value of the function is the sum of the two arguments. This subprogram is referenced in the main program contained in segment 161.

431, ICFI, is an integer function of an integer variable. The value of the function is obtained by exponentiating a real constant. This segment is referenced in the main program contained in segment 161.

441, IDFI, is an integer function of two integer arguments. The value of the function is obtained by raising a real value to the power which is the difference between the two arguments. The real variable is defined in a DATA statement. Segment 441 is referenced in segment 161.

451, IEFI, is an integer function with a single argument consisting of an integer array. The size of the array is declared in a DIMENSION statement and the elements of the array are added together to produce the value of the function. Segment 451 is referenced in connection with segment 161.

461, IFFI, is an integer function with twenty arguments of real variables and arrays and integer variables and arrays. The dimensionality of each array is declared within the subprogram. The value of the function is obtained by evaluating the equation which contains addition, subtraction, multiplication and exponentiation, of variables and array elements. This segment is referenced in segment 161.

402, GFS, is a real function of a double precision argument. The argument is assigned to the function name. This subprogram name is passed as an argument in segment 162 to segment 442, JRFS, which references it.

422, HFS, is a real function of two complex variables. The value of the function is obtained by assigning the imaginary part of the product of the complex values to the function name. This segment is referenced in segment 162.

432, IRFS, is an explicitly typed real function of a logical variable. The function value is defined by one of two logical IF statements, depending upon the value of the argument. This segment is referenced twice in segment 162.

442, JRFS, is an explicitly typed real function of an external procedure (segment 402) and a double precision variable. The value of the function is the value of the external procedure of which the double precision value is the argument. This segment is referenced in segment 162.

452, RFS, is a real function with twenty-one arguments of all the types of variables and arrays and an external procedure which is not referenced. Array and variable types are declared in logical, complex and double precision statements. Adjustable arrays appear in this subprogram. This segment is referenced in segment 162.

403, IFI, is an integer function of a double precision variable. The variable is assigned to the function name to produce the value of the function. This segment is referenced in segment 163 and also passed as an argument from segment 163 to segment 453 and segment 443.

423, JFI, is an integer function of two complex arguments. The value of the function is the imaginary part of the product of the two arguments. This segment is referenced in segment 163.

433, KFI, is an integer function of a logical argument. The value of the function is determined by one of two logical IF statements, depending upon the value of the argument. This segment is referenced twice in segment 163.

443, LFI, is an integer function of the external procedure IFI (segment 403) and a double precision variable. The value of the function is the value of the external procedure of which the variable is the argument. This segment is referenced in segment 163.

453, MFI, is an integer function with twenty-one arguments of all the types of variables and arrays and an external procedure. An adjustable array and its adjustable dimensions are dummy arguments of this subprogram. This segment is referenced in segment 163, and is similar to segment 452 except for function type, and the dummy function is referenced.

404, AFC, is a complex function, explicitly typed, of a real variable. The sum of the real variable and a complex value is the value of the function. This segment is referenced in segment 164.

414，BFC，is a complex function of an integer argument．A complex value is raised to an integer power to produce the function value．This segment is referenced in segment 164.

424，CFC，is a complex function of a real array．The elements of the array are subtracted from a complex constant to produce the function value．This segment is referenced in segment 164.

434，DFC，is a complex function of a double precision variable．The value of the function is obtained by subtracting a complex constant from the product of a complex constant and a real variable．This segment is referenced in segment 164.

444，EFC，is a complex function of a complex variable．The function value is the complex argument minus a constant．This segment is referenced in segment 164 。

454，FFC，is a complex function of a logical variable．The value of the function is determined by one of two logical IF statements，depending upon the value of the argument．This segment is referenced twice in segment 164.

464，HFC，is a complex function with twenty－one arguments of all the types of variables and arrays and a complex function which is not referenced． Variable and array types are declared in type statements in the subprogram．Adjustable arrays are arguments in this subprogram．A value is passed through common and is redefined within the subprogram． This segment is referenced in segment 164 and is similar to segment 452 。

405，AFD，is a double precision function of a real argument．The value of the function is set equal to the argument．This subprogram is referenced in segment 165 and also passed as an argument from segment 165 to segment 455 。

415，BFD，is a double precision function of an integer variable。 A double precision constant is raised to the power of the integer variable． This segment is referenced in segment 165.

425，CFD，is a double precision function of a double precision argument． The value of the function is the value of the argument．This segment is referenced in segment 165.

435，DFD，is a double precision function of two complex variables．The value of the function is the imaginary part of the product of the two complex variables．This segment is referenced in segment 165.

445，EFD，is a double precision function of a logical variable．The value of the function is determined by one of two logical IF statements， depending upon the value of the argument．This segment is referenced twice in segment 165.

455，FFD，is a double precision function of an external procedure（segment 405 ）and a double precision variable．This segment is referenced in segment 165.

465, GFD, is a double precision function of a double precision array. The elements of the array are added together to produce the value of the function. This segment is referenced in segment 165.

475, HFD, is a double precision function with twenty-one arguments of all the types of variables and arrays and a double precision function which is not referenced. Adjustable arrays are arguments in this segment. A value is passed through common and redefined in the function subprogram. This segment is similar to segment 452 and is referenced in segment 165 .

406, AFB, is a logical function of a real variable. This function is referenced in segment 166.

416, BFB, is a logical function of an integer argument. This segment is referenced in segment 166.

426, CFB, is a logical function of a double precision argument. This segment is referenced in segment 166.

436, DFB, is a logical function of a logical variable. The value of the function is the value of the argument. This segment is referenced in segment 166.

446, EFB, is a logical function of a complex variable. This segment is referenced in segment 166.

456, FFB, is a logical function of a real array. This segment is referenced in segment 166.

466, GFB, is a logical function of a real variable and a logical external procedure (segment 406). This segment is referenced in segment 166.

476, HFB, is a logical function with twenty-one arguments of all the types of variables and array elements and an external function which is referenced. This segment is referenced in segment 166.

407, AAQ, is a subroutine subprogram with integer and real variable and array names and a function in the argument list. This subprogram, called in segment 167, calls another subprogram (segment 417), whose argument list cuntains integer and real array names.

417, ABQ , is a subroutine subprogram called from another subroutine subprogram (segment 407) which is called in segment 167.

427, ACQ, is a subroutine subprogram which has no argument list. Variables and arrays are passed through common; some are redefined within the subprogram. This segment is referenced in segment 167.

408, ADQ , is a subroutine subprogram with twenty-four arguments of type integer, real, double precision, complex, and logical variables and arrays. This subprogram, called in segment 168, calls another subprogram (segment 418), whose arguments are integer and real variables and arrays.

418, $A E Q$, is a subroutine subprogram called from another subroutine subprogram (segment 408). The arguments are integer and real variables and arrays. This subroutine is used with segment 168.

428, AFQ, is a subroutine subprogram which has no arguments. Variables and arrays are passed through common; some are redefined within the subprogram. This segment is referenced in segment 168.

409, BLOKD, is a block data subprogram, which contains type, EQUIVALENCE, DATA, DIMENSION, and COMMON statements. These are the allowable statements in a block data subprogram, in which data statements assign values to variables and array elements which are in labeled common blocks. Hollerith data is assigned to each type of array, which are one, two, and three dimensional. This segment is to be run with segment 169.

410, SUBRQ, is a subroutine subprogram which contains no argument list and returns no values to the calling program. Arguments are passed through blank common. The subprogram contains FORTRAN statements, including input/output statements and references to intrinsic functions. This subroutine is called in segment 200. This segment is similar to main program segment 197.

Segment 419 BLAKD,
Segment 429 BLBKD,
Segment 439 BLCKD, are three block data subprograms, each of which, through data statements, assigns values to a different labeled common block. Each of these subprograms contains all of the statements allowed in a block data subprogram and each contains arrays of one, two, and three dimensions. These segments are run with segment 179.

411, SMCQ, is a subroutine subprogram called from a logical IF statement in the calling program, segment 300 .

412, MDQ, is a subroutine subprogram called from within a DO of the calling program. It is called from segment 197.

462, FMTQ, is a subroutine subprogram called by segment 312. FORMAT specifications and Hollerith constants are passed as arguments of the subroutine. An empty FORMAT specification is also tested.

413, MAQQ, is a subroutine subprogram in which an intrinsic function name is used as a variable name and a second intrinsic function name is referenced。 This subroutine is called from segment 352.

463, MBQQ, is a subroutine subprogram in which an intrinsic function name is used as a variable name. It is called from segment 352.

473, AMQQ, is a subroutine subprogram in which an intrinsic function name is used as a variable name. This subroutine is called from segment 352.

483, BMQQ, is a subroutine subprogram in which several intrinsic function references are nested and one intrinsic function name is used as a variable name. This subroutine is called from segment 352.
C. TEST UNIT SEGMENTS INDEXED TO THE FORTRAN STANDARD DOCUMENT ASA X3.9-1966

The following is the table of contents to the FORTRAN document X3.9-1966 with the corresponding FORTRAN Test Program Segments identified.

ASA X3.9-1966
Section Number and Title
FORTRAN Test Program Segment

1. Purpose and Scope
2. Basic Terminology
3. Program Form
3.1 The FORTRAN Character Set 008,149
3.1.1 Digits
3.1.2 Letters

008,360
3.1.3 Alphabetic Characters

008,009
3.1.4 Special Characters
3.1.4.1 B1ank Character
3.2 Lines
3.2.1 Comment Line
3.2.2 End Line
3.2.3 Initial Line
3.2.4 Continuation Line
3.3 Statements
3.4 Statement Label
3.5 Symbolic Names
3.6 Ordering of Characters
4. Data Types
4.1 Data Type Association
4.2 Data Type Properties
4.2.1 Integer Type

003,010,149,ALL
054,301, 302
4.2.2 Real Type

011
4.2.3 Double Precision Type 013
4.2.4 Complex Type 015
4.2.5 Logical Type 016
4.2.6 Hollerith Type

003,010
5. Data and Procedure Identification
5.1 Data and Procedure Names
5.1.1 Constants
5.1.1.1 Integer Constant
5.1.1.2 Real Constant
5.1.1.3 Double Precision Constant
5.1.1.4 Complex Constant
5.1.1.5 Logical Constant 5.1.1.6 Hollerith Constant

## 017

003,010,011
003,010,011
003,010,013
003,010,015,067
003,010,016
$003,010,312,462$

```
    5.1.2 Variable 003,010,312
    5.1.3 Array
        5.1.3.1 Array Element
        5.1.3.2 Subscript
        5.1.3.3 Subscript Expressions
    5.1.4 Procedures
5.2 Function Reference
5.3 Type Rules for Data and
        Procedure Identifiers
    5.4 Dummy Arguments
    6. Expressions
    6.1 Arithmetic Expressions
    6.2 Relational Expressions
    6.3 Logical Expressions
    6.4 Evaluation of Expressions
    7. Statements
    7.1 Executable Statements
        7.1.1 Assignment Statements
        7.1.1.1 Arithmetic Assignment 011,013,015,017,030-043,140-148
        7.1.1.2 Logical Assignment 016
                Statement
    7.1.1.3 GO TO Assignment
                Statement
    7.1.2 Control Statement
        7.1.2.1 GO TO Statements
        7.1.2.1.1 Unconditional GO TO 020
        Statement
        7.1.2.1.2 Assigned GO TO 021
        Statement
        7.1.2.1.3 Computed GO TO 022
                Statement
        7.1.2.2 Arithmetic If Statement 054,301,302
        7.1.2.3 Logical If Statement 054,300
        7.1.2.4 CALL Statement
        7.1.2.5 RETURN Statement
        7.1.2.6 CONTINUE Statement
        150,190-197,200
        7.1.2.7 Program Control
            Statements
        7.1.2.7.1 STOP Statement ALL,360
        7.1.2.7.2 PAUSE Statement (Omitted)
        7.1.2.8 DO Statement 190-197
        7.1.3 Input/Output Statements
        7.1.3.1 (initial record, next 180,182
        and preceding record)
    7.1.3.2 READ and WRITE Statements
        7.1.3.2.1 Input/Output Lists 008,310
        7.1.3.2.2 Formatted READ 008,009,196,310,312
        7.1.3.2.3 Formatted WRITE 008,009,196,310,312
        7.1.3.2.4 Unformatted READ 180,182
        7.1.3.2.5 Unformatted WRITE 180,182
```

7.1.3.3 Auxiliary Input/OutputStatements
7.1.3.3.1 REWIND Statement ..... 180,196
7.1.3.3.2 BACKSPACE Statement ..... 182
7.1.3.3.3 ENDFILE Statement ..... 180
7.1.3.4 Printing of Formatted ..... 008
Records
7.2 Nonexecutable Statements
7.2.1 Specification Statements
7.2.1.1 Array-Declarator ..... ALL
7.2.1.1.1 Array Element ..... 003,008,180,192Successor Functionand value of asubscript
7.2.1.1.2 Adjustab1e Dimension 162-166,452,453,464,475,476
003,008,169,409,360 (all parts)
162-169,200,360,40 ..... 409003,022,110,111003,010,312,441
008,009, 310, 312,410,462
008,009,312,462
169,409,360 ..... 162-167 ..... 008,009
008,009
008,310
2.1.2 DIMENSION Statement
7.2.1.4 EQUIVALENCE Statement
7.2.1.4 EQUIVALENCE Statement ..... 00,310
action with Input/OutputList
7.2.3.5 Scale Factor ..... 008,310
7.2.3.5.1 Scale Factor Effects ..... 008,310
7.2.3.6 Numeric Conversions ..... 008,310
7.2.3.6.1 Integer Conversion ..... 008,011
7.2.3.6.2 Real Conversions ..... 008,011
7.2.3.6.3 Double Precision ..... 008,013Conversions
7.2.3.6.4 Complex Conversion ..... 008,015
7.2.3.7 Logical Conversion ..... 008,016
7.2.3.8 Hollerith Field ..... 009Descriptor
7.2.3.9 B1ank Fie1d Descriptor ..... 008
7.2.3.10 Format Specification in ..... 312,462Arrays
8. Procedures and Subprograms
8.1 Statement Functions
8.1.1 Defining Statement Functions ..... 005,006,410
8.1.2 Referencing StatementFunctions
8.2 Intrinsic Functions and Their ..... 055-073,352
I-C-3
8.3 External Functions
8.3.1 Defining Function Subprograms
8.3.2 Referencing External Functions
8.3.3 Basic External Functions
8.4 Subroutine
8.4.1 Defining Subroutine Subprograms
8.4.2 Referencing Subroutines
8.5 Block Data Subprogram
9. Programs
10. Intra-and Inter-Program Relationships

400, 420, 430,etc.
160-166
080-103,351
$408,410,418,427,428,462$
$167,168,200,312$
$169,179,409,419,429,439$
Rules stated are included under tests related to Section 3 through Section 8 of the FORTRAN Standard
D. PROGRAM INFORMATION

The following points describe the organization of FOPTRAN test programs:
The programs are divided into a number of small segments.
Most segments, except for specification statements, I/O assignment statements, statement functions, subprograms and DATA statements, are completely self-contained.

Most segments are very simply written with the testing devoted to related features described in the ASA standard. The number of FORTRAN statement types is minimized in order to make each test less dependent on other language features.

Every segment begins with a heading of comment lines which gives the segment name, segment number, pertinent ASA references, purpose of the segment and restrictions observed in the segment.

The last line in every segment is marked by a comment line with the message 'END OF TEST SEGMENT xxx."

Comments, throughout each segment, give detailed ASA references and additional explanations of the coding.

D1. Conventions Used in the Test Programs
Certain conventions have been adopted and are used throughout the document, the program code and the test results. These conventions provide the user with a means to:
identify types of data,
determine the number of dimensions associated with a given array,
distinguish program elements,
correlate references between the ASA FORTRAN standard document and the pertinent test segments.

The conventions are described below.
a) Segment Identification

Each segment is identified in the following two ways:
By a 3- to 5- character (A-Z, 0-9) descriptive name (e.g., DPLOG, SBRTN).

By a unique 3-digit (0-9) number.
Both the segment name and number appear in the program listing, the documentation and the generated test results.

Line numbers, columns 73-80, are outside of the standard, but are usually available in an implementation of the FORTRAN Standard, when the source statements are introduced to the processor from punched cards.

The scheme used to identify FORTRAN lines is a compromise between the ability to associate the program listing with this document and the card handling problem. The FORTRAN test program listing represents both a statement of the program for the processor and a document for the user. The program listing also assists in the consolidation and isolation of test units. Although each line number is unique, a test program unit may contain FORTRAN lines with columns 74-76 (segment number) with segment numbers 001-007 inserted within the test units. Columns $73-80$ are coded in the following fashion:

Column 73 contains

Columns 74, 75, 76 contain nnn

Columns 77, 78, 79 contain

Column 80 contains

P for FORTRAN Test Version 1.
H for FORTRAN Test Version 3.
nnn where 'nnn' are 3 unique digits (0-9) which identify the program segment. (The greatest segment number allowed is 699).
mmm where'mmm' are 3 digits (0-9) representing a line number within the program segment.
where x is either zero or five and allows for the insertion of lines at a later time.

In Version 1, the sequence numbers (columns 77-80 for segments 001 and 007 start with 0010 and are incremented by 5 , with each new segment number (columns 74-76) forcing the beginning sequence number to be even.

In Version 3, the sequence numbers (columns 77-80) for segments 000, 001 , and 007 are increased by 5 in column 80 and each Part is initiated by the following sequence number: Part 1, 0010 ; Part 2, 0400; Part 3, 0700; Part 4, 1200; Part 5, 1800; Part 6, 2300; Part 7, 2700; Part 8, 3200; Part 9, 3700; Part 10, 4300; Part 11, 4800; Part 12, 5400; Part 13, 6000; Part 14, 6400; and the statement function definition segment 005 imbedded in segment 197 begins at 0500 .
c) Statement Labe1s

Each statement label is a string of four digits ( $0-9$ ) 。 To avoid duplicate labels in the test program, the first three digits of the string contains the number of the segment in which the statement label is found. (See the description of columns 74-76 above). The fourth digit is used to make the string unique within that particular segment.

This convention provides ten unique labels per program segment. When more than ten labels are needed in any segment, digits 1-3 of the extra labels contain a unique number between 700 and 999, instead of the program segment number. For this reason, the greatest program number allowed is 699. A table of currently used additional statement numbers is contained in Section I-D3e.
d) Format of Comments

Every comment line contains ' $C$ ' in column 1, followed by five asterisks (*) or a "C=" in columns 1 and 2.

Each segment is preceded by a heading of comment cards which give the segment name, segment number, purpose of the segment, restrictions observed, ASA references and miscellaneous comments.

Additional comment lines, interspersed with the actual coding describe the specific purpose of the coding which follows and give pertinent ASA references.

Comment lines containing " $\mathrm{C}=$ " in columns 1 and 2 denote the required Specification statements, I/O Assignment statements, STOP statement and END line needed to construct a FORTRAN program if each main program segment is to be run as a separate test unit.
e) Format of the Generated Test Results

The generated test results of every segment start on a new page and are headed by several lines which give the segment name, segment number, purpose of the segment (very briefly stated), and ASA references. The printed area is constrained to an $81 / 2$ by 11 inch page, with a maximum of 57 lines printed per page.
f) Naming Conventions

A unique 3- to 5-character designation is used to identify a variable, array, function or subprogram. The combination of the last two characters in the name indicates the type and category. The character preceding the last two flags items which appear in COMMON or EQUIVALENCE statements. One or two optional characters may begin each name to make it unique. The conventions are as follows:

If character 5 is (or last character)

I the type is integer;
$S$ the type is real;
D the type is double precision;
C the type is complex;
B the type is logical;
H the actual argument is Hollerith; Q the string represents a subroutine.

If character 4 is
(or next to last character)
a where 'a' is any other letter (A-E, G-U, W-Z) for cases in which none of the other codes are applicable.

W the name is a dummy argument;
$X$ the name appears in a COMMON statement;
Y the name appears in an EQUIVALENCE statement;
$Z$ the name appears in both COMMON and EQUIVALENCE statements;
a where 'a' is any other letter ( $\mathrm{A}-\mathrm{V}$ ), the name appears neither in COMMON nor in an EQUIVALENCE statement.

Characters 1 and 2 are aa where 'aa' are any letters ( $A-Z$ ) associated with the string. These two characters are used only to insure that each name is unique. Either or both of them may be omitted, if desired.

Examples of this convention are A3I, BBXVD, CBFS, PAAQ where the strings represent a 3-dimensional integer array, a double precision variable (used in a COMMON statement), a real function name and a subroutine name, respectively.

D2. Assumed Leve1s for Non-specified FORTRAN Areas
The ASA standard does not impose specifications in many areas that are clearly subject to limitations in actual FORTRAN compilers. Therefore, in order to design meaningful tests, some additional specifications have been established. These limits are described below.
a) Level of Nesting

The DO loop segments of the program contain a maximum of FIVE nested loops.
b) Number of Arguments

The test program contains subprograms with up to TWENTY-FIVE arguments.
c) Size of Arrays

The size of arrays is generally very small, i.e., usually less than TWENTY words.
d) FORMAT Standards

FORMAT statements never cause more than FORTY characters on a line to be generated in the output.
e) Number of Parentheses

Expressions in the test program never exceed TEN levels of parentheses.
f) GO TO Branches

The number of branches in assigned and computed GO TO statements never exceeds TWELVE branches.
g) Constant Length

Constants are kept small in order not to exceed the storage unit length capacity of some computers. The limits on constant length are set as follows:

Integer constants - 5 digits
Real constants - 7 digits
Double precision constants

- 14 digits

Complex constants
(each half)
Hollerith constants

- 7 digits
- 2 characters except in segment 009 which tests A-conversion for 1 to 4 characters and 26 characters for the truncation test.

D3. Names and Statement Numbers Used in the Test Programs
Only those names which are used as array names, external function and subroutine names, common block names, and variable names appearing in a DATA statement appear in the following lists. The list of array declarators appearing in type statements and COMMON statements is supplied to assist the user when he wishes to extend or revise the test programs.
a) Subprogram Names Used in the Test Program Set and the Number of Arguments

Integer Functions

| IAFI | 1 |
| ---: | ---: |
| IBFI | 2 |
| ICFI | 1 |
| IDFI | 2 |
| IEFI | 1 |
| IFFI | 20 |
| IFI | 1 |
| JFI | 2 |
| KFI | 1 |
| LFI | 2 |
| MFI | 21 |

Double Precision
Functions

| AFD | 1 |
| :--- | ---: |
| BFD | 1 |
| CFD | 1 |
| DFD | 2 |
| EFD | 1 |
| FFD | 2 |
| GFD | 1 |
| HFD | 21 |

Real Functions

| AFS | 1 |
| ---: | ---: |
| BFS | 2 |
| CFS | 1 |
| DFS | 2 |
| EFS | 1 |
| FFS | 20 |
| GFS | 1 |
| HFS | 2 |
| IRFS | 1 |
| JRFS | 2 |
| RFS | 21 |

Logical Functions

| AFB | 1 |
| :--- | ---: |
| BFB | 1 |
| CFB | 1 |
| DFB | 1 |
| EFB | 1 |
| FFB | 1 |
| GFB | 2 |
| HFB | 21 |

Complex Functions

| AFC | 1 |
| :--- | ---: |
| BFC | 1 |
| CFC | 1 |
| DFC | 1 |
| EFC | 1 |
| FFC | 1 |
| HFC | 21 |

Block Data Subprograms-No Names Permitted In FORTRAN Language But Identified by Comment Cards As:

Subroutines
BLOKD

| AAQ | 9 |
| ---: | ---: |
| ABQ | 3 |
| ACQ | 0 |
| ADQ | 24 |
| AEQ | 8 |
| AFQ | 0 |
| MDQ | 2 |
| SMCQ | 1 |
| FMTQ | 22 |
| SUBRQ | 0 |
| MAQQ | 2 |
| MBQQ | 2 |
| AMQQ | 2 |
| BMQQ | 2 |

b) Array Declarators in Type Statements and COMMON Statements

Double Precision

## AC1D (10)

AlD (4)
A2D $(2,2)$
A3D $(2,2,2)$
BC2D $(7,4)$
$\operatorname{CC} 3 \mathrm{D}(7,2,2)$
DPA1D (5)
DPA2D $(2,2)$
EP1D (43)
FC2D $(5,5)$
$\operatorname{MCA} 3 \mathrm{D}(1,4,2)$
$\operatorname{RC} 3 \mathrm{D}(3,3,3)$
AX1D
AX2D
AX3D
DX1D
DX2D
DX3D
Logical
A1B(2)
A2B $(2,2)$
A3B $(2,2,2)$
GG1B(2)
$\operatorname{GH} 2 \mathrm{~B}(1,2)$
$\operatorname{GI3B}(1,1,2)$
MCA1B(7)
L1B(10)
AX1B
AX2B
AX3B
DX1B
DX2B
DX3B

## Complex

A1C (12)
A2C $(2,2)$
A3C $(2,2,1)$
B1C (8)
B2C $(4,2)$
B3C $(2,2,2)$
LLIC (32)
$\operatorname{LM2C}(8,4)$
$\operatorname{LN} 3 \mathrm{C}(9,2,2)$
EP1C(30)

AX1C
AX2C
AX3C
DX1C
DX2C
DZ3C

Integer
III(5)
I2I $(2,2)$
I3I $(2,2,2)$
$\operatorname{MCA} 3 \mathrm{I}(2,3,3)$
IU2I $(4,2)$
$\operatorname{IT} 3 \mathrm{I}(4,2,2)$
IU3I $(2,3,3)$

```
AC1S(25)
AC2S (5,6)
AC3S (1, 1,3)
A1S(5)
A2S (2,2)
A3S (3, 3, 3)
CMA1S(5)
CMB1S (5)
EP1S(33)
IAC1I(5)
IAC2 I (2,7)
MCA1I(5)
L1I (10)
IAB1I(4)
IAB2I (3,3)
IAB3I (2,2,2)
AB1S (4)
AB2S (3,3)
AB3S (2,2,2)
IV1I(1024)
ZU1S(12)
ZU3S (3,2,2)
ZU2S (4,2)
ZT1S(4)
YER1S(7)
J (2)
JJ (1, 1)
JJJ (1,1,1)
JJJJ (1,1)
JJJJJ(1)
JJJJJJ(1)
GOTO (2,2)
IF (5)
MX1I (3)
TX1S(3)
MMY1I (400)
NNY3I (20,10,2)
MX2I (2,3)
TX2S (2,2)
WAZ2S(3,2)
RVY1S(2)
RVY2S (1,2)
JY2I (2,2)
JY1I(5)
NZ1I(4)
NZ2I (4,2)
wAZ1S(2)
```

c) Blank Common Organization and Block Names

There are two separate mappings of COMMON in the Program Set. Segment 360, the last test in the program set, tests COMMON, EQUIVALENCE, and DIMENSION using a special organization of blank COMMON not associated with any other program segment. For this reason segment 360 may not be combined with any of the segments listed below which make use of a different arrangement.

The following ordering of blank COMMON is used in Segments 162, $163,164,165,166,167,168$ and 200.

AXVS
CXVS
IXVI
IAX1I (4)
$\operatorname{IAX} 2 \mathrm{I}(3,3)$
IAX3I $(2,2,2)$
BXVS
AX1S (4)
$\operatorname{AX2S}(3,3)$
AX $3 \mathrm{~S}(2,2,2)$
AXVD
AX1D (2)
$\operatorname{AX2D}(2,2)$
$\operatorname{AX} 3 \mathrm{D}(2,2,2)$
AXVC
AX1C (2)
AX2C $(2,2)$
AX3C $(2,2,2)$
AXVB
AX1B(2)
$\operatorname{AX} 2 \mathrm{~B}(2,2)$
$\operatorname{AX} 3 \mathrm{~B}(2,2,2)$
The six labeled COMMON blocks are identified by the names:
BLKn where n is 1 to 6
The organization of the data in the labeled COMMON blocks is specified in Segment 179.
d) Variables and Array Elements Defined in DATA Statements

Symbolic names of variables and array elements with their corresponding values are defined in DATA statements in segment 003 and tested in segment 010. When augmenting the test programs the following variable names and array element names may not appear in subsequent DATA statements nor be redefined in tests which precede segment 010 (e.g., 008 or 009). No restriction is placed upon the redefinition of these variables or array elements in test segments which follow segment 010 .

DATA Statement 1
Symbolic Name Form and Value of the Entry
$\operatorname{I1I}(1)$
$\operatorname{MCA} I(1,2,1)$
$\operatorname{MCA} 3 I(2,2,2)$
$\operatorname{IAC} 21(2,5)$
$\operatorname{IAC} 2 \operatorname{I}(2,6)$
$\operatorname{MCA} 1(2,1,1)$

DATA Statement 2
Symbolic Name
EP1S(8)
EP1S(10)
EP1S (12)
$\operatorname{AC} 2 \mathrm{~S}(5,5)$
EP1S(11)
$\operatorname{AC} 2 S(5,3)$
$\operatorname{AC} 2 S(5,2)$
DATA Statement 3
Symbolic Name
BVD
DPA2D $(2,1)$
CVD
DPA2D $(1,2)$
DVD
DPA2D $(2,2)$
DATA Statement 4
Symbolic Name
ADSVC
$\operatorname{LN} 3 \mathrm{C}(9,1,2)$
LL1C (30)
LN3C $(8,2,2)$
$\operatorname{LM2C}(8,3)$
$\operatorname{LN} 3 \mathrm{C}(9,1,1)$
LL1C (32)
$\operatorname{LN} 3 \mathrm{C}(8,1,2)$

## 0

2* 10
3* 246

Form and Value of the Entry

$$
2 * 0.0
$$

2*-750.05
.24615E3
2.4615E2
3.54674E+3

Form and Value of the Entry
+34567890.1D-3
$345.678901 \mathrm{D}+2$
112233.5D-08
11.22335D-4
3.4D12
0.34 D 13

Form and Value of the Entry

```
2*(11.1, 22.22)
(-3.45E1, -67.8E-1)
(-34.5E0, -6.78E0)
(10.E0, -20.E0)
(1.0E1, -2.0E1)
(-20.0E1, +4.E3)
(-200.E0, +4000.E0)
```

DATA Statement 5 Symbolic Name

## MAVB

MCA1B (6)
MBVB
DATA Statement 6 Symbolic Name

GI 3B $(1,1,2)$
GG1B(1)
EP1S(15)
DATA Statement 7
Symbolic Name
III(2)
IAC2I $(1,5)$
IAC2 I $(1,3)$
I1I(5)
IAC2I $(2,4)$
$\operatorname{MCA} 3(1,1,2)$
AVI(Integer type)
EP1S (13)
$\operatorname{AC} 2 S(2,6)$
$\operatorname{AC2S}(1,6)$
$\operatorname{AC3S}(1,1,1)$
$\operatorname{AC} 2 S(3,6)$
$\operatorname{AC} 3 S(1,1,2)$
$\operatorname{AC2S}(4,6)$
AVD
A1D (1)
DPA2D $(1,1)$
$\operatorname{MCA} 3 \mathrm{D}(1,1,1)$
AlD (2)
$\operatorname{MCA} \mathrm{D}(1,1,2)$
LL1C (29)
$\operatorname{LN3C}(8,2,1)$
BCVC
LM2C $(8,4)$
GH2 B $(1,1)$
$\operatorname{GI} 3 \mathrm{~B}(1,1,1)$
MCVB
III (3)
I1I (4)
MCA3I $(1,2,2)$
$\operatorname{AC} 2 \mathrm{~S}(5,6)$
JVS (type REAL)
EPIS (14)
AC3S ( $1,1,3$ )
IAC2 I $(1,4)$
CHEVC

Form and Value of the Entry
2* .TRUE.
.FALSE.

Hollerith Data Form
2HNO
2* 2HAD

Form and Value of the Entry
3* 0

4* -750

2* 0 。
2* 246.15
354674.E-2
$354.674 \mathrm{E}+1$
35467.4E-01

3* -. 295D5
$-29.5 \mathrm{D}+3$
3456. $78901 \mathrm{D}+01$
$0.345678901 \mathrm{D}+5$
2* (1.11E1, +222.2E-1)
$(-34.5,-6.78)$
(-.345E2, -678.E-2)
2* .TRUE.
. FALSE.
2* 10
+246
-. 75005E03
$-7.5005 \mathrm{E}+02$
2 HBC
$2 \mathrm{H}^{*}=$
2H P
2* (10., -20.)

LL1C (31)
DCVC $\quad(-200 .,+4000$.
LM2C $(8,2)$
(-2000.E-1, +400.E1)
A1D (3)
$\operatorname{MCA} 3 \mathrm{D}(1,3,1)$
+1122.335D-6
A1D (4)
$\operatorname{MCA} 3 \mathrm{D}(1,4,1)$
$0.00001122335 \mathrm{D}+2$
34.0 D 11

MCA1B (7)
GH2B(1,2)
0.034 D14

2* .FALSE.
e) Statement Numbers Used (and Not Used) Between 7000-9999 With the Segment Numbers Associated


[^1]E. STRUCTURING, RESTRUCTURING AND EXTENDING THE TEST PROGRAMS

E1. Program Structure
Version 1 has been structured as 116 executable FORTRAN programs with provisions for linking test units end to end. Version 3 has been structured into 14 executable FORTRAN programs.

Every main program test unit contains at least two segment numbers, the first executable statements which assign the I/O unit numbers, identified as segment 007 in columns 74-76, and the test segment identified by the 3 -digit identification 008 to 360 .

An executable program includes some of the following segment numbers:

Specification Statements
DATA Statements
Statement Function Definitions
I/O Assignment Statements
Main program segments
Subprograms

Segment 001
Segment 003
Segment 005 or 006
Segment 007
Segment 008-360
Segment 400-483

Because test units may be linked end to end, the segment numbers 001 to 007 are identified by these numbers within the test unit in which they are embedded to facilitate the identification and location of these elements in a FORTRAN program and to aid in the elimination of duplicate elements when test units are consolidated.

Each test unit, even when consolidated with other test units, can be viewed from the program listing as an independent test because the necessary Specification statements, I/O assignment statements, STOP statements, and END lines are inserted as specially structured comment lines in their appropriate locations. Lines beginning with the characters " $\mathrm{C}=$ " identify these otherwide FORTRAN statements.

E2. Consolidating Test Program Units Using Version 1
Version 1 contains a directory of the test segments as a set of 342 comment lines before the first test segment. These are identified as segment 000 and may be used to create a directory to head any consolidation of the test programs. (A directory of only those test units appearing in a specific part heads each executable program in Version 3.)

In both versions, comment lines have been inserted to ease the burden of coupling test units together or isolating them.

Specification statements and END lines have unique position requirements in the FORTRAN standard. Specification statements must precede Statement Function definitions and the first executable statement, and the END line must be the last line of a program unit. Comment lines may be anywhere before the END line.

Each main program unit in Version 1 has been created as if it had been developed from Version 3. That is, the comment lines inserted into each test unit which directs the user how to create a single test program from a consolidated set has actually been performed to create Version 1, leaving the comment lines in place. This permits the user who has consolidated the test programs to later isolate individual test units as needed with directions for the process contained in the program. For example, in segment 008 test unit, the FORTRAN text contains the following message:

C***** WHEN EXECUTING ONLY SEGMENT 008, THE SPECIFICATION STATEMENTS
C***** WHICH APPEAR AS COMMENTS MUST HAVE THE C= IN COLUMNS
C***** 1 AND 2 REMOVED
Below this message is a set of comment lines which, except for columns 1 and 2, look like Specification statements with the segment number 001 in columns 74-76. In Version 1 , this action has already been performed leaving the $\mathrm{C}=$ comment lines in the program and inserting the actual Specification statements below these comment lines but with the segment number changed from 001 to the test segment number, in this case 008. The four digit sequence number, columns $77-80$ is unique for these inserted lines, and is assigned characters and digits which will facilitate the location of these lines. Similar messages appear before the $I / O$ assignment statements and the STOP statement and END line. The following identification code has been assigned for columns 74-80:

## Specification Statements nnnAx $\not \boxed{\varnothing} \not$

I/O Assigment Statements nnnBdøß
STOP and END
Where nnn is the test segment number in which the statements are embedded, $x$ is 1 to 9 and $A$ to $F$, and $d$ is 1 or 2. The last two character positions are blank. Specification statements may contain continuation lines, so that the sequence number is significant.

In order to link test units end to end into a single executable program, it is necessary to eliminate duplicate specifications, STOP and END lines, and I/O assignment statements (if the unit numbers are changed by the user). These appear only in the main program test units. Elimination of duplicate symbolic names from the Specification statements is performed on each of the nine (DIMENSION, COMMON, EQUIVALENCE, EXTERNAL, REAL, INTEGER, DOUBLE PRECISION, COMPLEX, and LOGICAL) statements independently. That is, if dimension information is expressed in a type statement instead of a DIMENSION statement, all test programs which require this specification information for a particular symbolic name will be consistent.

The appropriate directory and the consolidated specifications, identified as segment 001, should be placed in front of the first test unit of the consolidated set, the I/O assignments placed as the first executable statements (segment 007) within the first test unit, and a single STOP statement and END line must appear as the last lines of the main program unit. If Statement Function definitions are a part of a test unit, these
must be placed before the first executable statement. Segments 110 , 111, and 197 contain Statement Function definitions. If segments 110 and 197 are combined into the same executable program, one copy of segment 005 must be removed. Test units should be performed in the order of the directory, particularly segment 010, Data Statement Use, must appear in order, because the potential reuse of data names appearing in a DATA statement in other program test units cannot be guaranteed.

If during the consolidation process, an attempt is made to include more test units than the FORTRAN processor will accept into a single executable program, it will be necessary to return the specification statements, I/O assignment statements and STOP and END lines to the appropriate test units not included with the finally consolidated set for later use in another consolidation. The segment number associated with these lines identified by the letters "A", "B", and "C" in column 77 is contained in columns 74-76.

When test programs are consolidated into larger executable programs, it is desirable to have some means of identifying the test results with some additional information related to the environment of performing the tests, such as computer name, compiler version, operating system version, date, and any additional information which would distinguish successive running of the test programs. This can be achieved by incorporating the FORTRAN lines, identified as segment 007 -which are embedded in segment 008 starting with the comment line "IDENTIFY THE SOURCE OF THE TEST PROGRAMS", into the first test of each consolidated test set following the I/O assignment statements. The last continuation line of the FORMAT statement at Statement Label 0071 should be altered to reflect a unique means of identifying each executable program. In Version 3, this has been done by identifying each executable program as a PART, numbered from 1 to 14. The first six input cards associated with segment 008 will then be required for running each of the consolidated test sets. Cards 1, 3 and 5 are prepared by the user, replacing the dummy information on the card images supplied, with the environmental information. See Section II-A2 Input Data Preparation.

The number of test segments which may be linked end to end is a function of the power of the FORTRAN processor with the following exceptions:
a) Segment 360 may not be linked with any other test segment which uses blank common.
b) Segments 169 and 179 when consolidated into a single program will cause different elements of a specific labeled common block to be initialized from DATA statements in different BLOCK DATA subprograms. While the current FORTRAN standard does not exclude this, it is anticipated that the future revised FORTRAN standard may prohibit the user from so doing.

E3. Deleting a Section of a Test Unit
If certain test elements fail to perform on a system because some elements of the FORTRAN language have not been implemented and the test unit cannot be executed, it will be necessary to inspect the test unit to determine what statements together with the corresponding WRITE statements are affected. When a section of a test unit is altered it is recommended that those statements which are changed or deleted have appropriate comment cards inserted to identify the change. This can be achieved by making the current statements into comment lines with a character other than blank, *, or $=$ as the second digit and a comment card containing the number of lines which follow in the replacement. If a statement which is deleted contains a statement label, it will be necessary to repunch the card with the four digit statement label right justified and replace column 1 with a "C".

If a program test unit is too large for running as a single test unit it may be separated into smaller units for testing. This may be necessary for test segment 008, Formatted READ and WRITE, because of its current size and the number of FORMAT statements included in this test unit. The sample Result Output should be inspected. The breaks in the program should conform to locations where a new page indicator is detected at the beginning of a FORMAT statement. Data cards are identified in the program listing and the card number is given at the point of the appropriate READ statement.

E4. Deleting an Entire Test Unit
All test units are identified by segment numbers in columns 74-76. Test units contain "C=" comment cards for specification statements and I/O assignment statements with segment numbers 001 and 007, respectively. STOP and END cards appear at the end of each test unit as "C=" comment cards with the sequence number the same as the test unit number. All cards related to a test unit may be removed by inspection of the program listing. Any subprogram which is associated with that program test unit will not be assoicated with any other program test unit and may be removed. Distinctive comment cards separate test units.

## E5. Adding to a Test Unit

Any program test unit may be extended by appending statements after the last executable statement in a program test unit. See Program conventions for symbolic name and statement label use Section I-D. Result output pages have been limited to $81 / 2$ by 11 inch pages with a new page indicator for each page. All variables must have their values initialized in the test unit. Any new specifications must be introduced into "C=" cards within the test unit and a check made of the specifications contained at the beginning of the Part in Version 3. Array declarators used in the test set are identified in Section I-D-3.

Be sure that the programming conventions used in this test set are followed. A segment number which has not been used less than 399 may be used for a main program test. Numbers 400-699 which have not been used are available for subprograms. In general, the number chosen should be high enough so that those elements of the language which must be used in the test have already been tested.

Make sure that each new test is self contained. Initialize all values within the new segment itself. Use the same comment line structure to separate the new test unit, and intersperse comments to describe the test. Update the directory, specification section and the comment cards at the beginning of the Part in Version 3 to reflect any new program test units added. The listing of the program is supposed to contain enough comments to permit the programs to be used if additional documentation is not available.

## F. DIFFICULTIES ENCOUNTERED DURING THE TEST PROGRAM DEVELOPMENT

During the implementation of the program design, a number of difficulties other than the normal program debugging arose which required resolution. These difficulties have been classified under the following five categories.

## F1. Interpreting the FORTRAN Standard

> In interpreting the FORTRAN Standard document X3.9-1966, there was a conscious effort to glean from the document only that which was stated, and not to be influenced by earlier implementations of the FORTRAN language. This lead to a long list of questions which needed resolution. The ASA FORTRAN technical committee, X3J3, reconvened to address these and other questions of interpretation of the Standard. The committee published two clarification reports $[2,3]$ concerning the interpretation of the standard. Those questions which could not be resolved without actually revising the standard have been deferred and will be handled in the future revision and extension to the FORTRAN standard. Some of the questions did not arise until some initial test units were run on different processors and the different interpretations of the standard could be asserted and appeared to be justified by the wording in the standard.

F2. Precision, Conversion and Maximum Value of Numeric Data
The choice of the actual values used in arithmetic expressions presented considerable difficulty. The range of the exponent, which is not covered by the FORTRAN standard was kept small so that the variation on different processors would not be reflected in the test results. To overcome some of the precision problems, small fractional powers of two were used in the hope that the conversion of these values would be exact. Recognizing that the FORTRAN standard defines a real constant to be an "approximation to the digit string interpreted as a decimal numeral" the equation $1.3+1.3=2.6$ may not be true if the result were compared to the constant 2.6 because the constant when converted and when doubled may not have the same internal representation as the constant 2.6. Rather than attempt to apply an error tolerance to the results, it was decided to subtract the expression result from the expected result and rely upon the rounding under the Fw.d format field descriptor on output to compensate for a small difference in values. Because the Fw.d format field descriptor cannot be applied to results derived from a double precision operation, it was necessary to stipulate in the test results a reasonable error factor to be applied to the value if the result was not zero.

F3. Meaningful Tests and Comprehensible Results
The development of meaningful test programs of the FORTRAN Standard language cannot be separated from the presentation of the results of the test. If it can be considered that any test result value printed from a specific application of the test programs on a FORTRAN processor could be in error, the means to determine the statements involved in the test result must be readily discernible. This lead to examining, on a case basis, how to present the results. Where ever possible the results obtained from
arithmetic operations were subtracted from the expected result and the expected value of zero printed, so that the user could quickly scan a page of results and determine any errors. When this was not possible, Hollerith information is printed directly above the expected value so that the eye can quickly scan the results for discrepancies, or some appropriate means such as the test number for the value in error, so that reference back to the program listing could be made.

Various elements of the FORTRAN language presented some difficulties in displaying the results.

The effect of the scale factor both on READ and WRITE is such an example. If the FORTRAN processor does not perform this conversion properly, and only the expected result is printed with the processor result, it is tedious to determine from the program listing what actual data and format field descriptor is associated with a value. Because of this, the information which a user would need to determine the operation being performed is presented in a tabular form with the expected result and the actual result.

The Intrinsic Functions SNGL and DBLE presented the problem of storage unit size and how can it be determined whether these functions are actually performed when the maximum real and double precision constant length established for the programs is 7 and 14 digits respectively. The FORTRAN Standard does not define these functions to operate under the same rules as the corresponding assignment statement operation identified in Table 1 of the FORTRAN Standard. The FORTRAN Standard does not address the precision of a value, so that it cannot be determined from the document if a standard conforming program may READ or WRITE values which express a precision in excess of the processor capability. Because of these factors, for FORTRAN processors which can express a REAL value of 14 or more digits in a single storage unit, the printed results may not display the value to a precision large enough to encompass the actual function result. Increasing the number of decimal digits expressed in the format field descriptor should eliminate the difficulty.

The Basic External Functions presented a significant problem for devising test programs because the units of the arguments are not specified in the FORTRAN Standard and there was to be no attempt to address the unspecified range of arguments, precision or accuracy of the function results. This lead to using the "defacto standard" practice for the units of arguments and to select arguments which reduced the probability of variation due to conversion and for which there were also published table values. It is hoped that the arguments selected with their expected results, although not necessarily representative of normal usage, would constitute a basis for the user determining whether the function referenced is, in fact, the function obtained. Table values were not readily available for the complex functions so that a different method had to be employed.

## F4. FORTRAN Compilers with Language Extensions

The difficulty of assuring that a FORTRAN program is confined to that which is defined in the FORTRAN Standard X3.9-1966 is substantial.
Because a program produces the same correct results on many FORTRAN processors does not in itself substantiate the program to be standard conforming. Even though the test programs were desk checked, not all non-standard usage was picked up by this method. Moving from one processor to another brought to light the differences in the extensions or relaxations permitted on various FORTRAN processors. Those programming errors which persisted.undetected through many FORTRAN processors where they were treated as extensions are:

- Missing type declarations for dummy arguments of statement functions
- Missing commas after an nH format field description in FORMAT statements
- Non agreement between format field description and the type of the list element
- Missing decimal point in a real constant in a real expression
- Lack of agreement of type between actual and dummy arguments of a subprogram where the dummy argument is not referenced.


## F5. Performing the Tests

The initial running of the completed test programs on various FORTRAN processors was performed from punched cards. The punch card code used was the BCD-H set which is identified in Appendix D of the FORTRAN Standard X3.9-1966. It was assumed that any computer installation would have a conversion routine for this code to its own, if it were not an option of the compiler. For the most part the testing was performed on the consolidated test set which reduced the number of executable programs from 116 to 14 . This was done to minimize the number of control cards needed to be inserted around the programs. In spite of assistance from systems personnel at the test site in every initial running on a different processor one or more programs had to be resubmitted to the computer because of operating systemis control card errors. The kinds of errors were:

- Failure to identify the FORTRAN Programs as BCD card code producing errors in scanning the FORTRAN statements.
- Failure to identify the data as BCD H set causing the execution to be aborted on improper symbols on input cards.
- Missing or mispunched control cards.
- Inproper sequencing of control cards.

At no time was the alloted time on the computer or the maximum number of pages of printing exceeded. Because many of the test programs may require the same set of control cards, special care must be taken for those programs requiring data, an auxiliary tape unit and subprograms.

The differences in capability of operating systems did not present a difficulty but the lack of standard terminology and definition of functional capability presented barriers in human communications.

When a compiler had an option to check the programs for conformance to the FORTRAN standard and no diagnostic messages resulted, the test program writers were lulled into the belief that the programs met the standard, which later running on a different compiler proved not to be the case. Not all non-standard usage even within a single FORTRAN progran unit were detected.

## G. REFERENCES

1. American Standard FORTRAN X3.9-1966 - since the original publication of the FORTRAN Standard, the standardizing organization has changed its name from American Standards Association to United States of America Standards Institute and recently to American National Standards Institute. Therefore, documents identified as ASA X3.9-1966, USASI X3.9-1966 and ANS X3.9-1966 all refer to the same document.
2. Clarification of FORTRAN Standards - Initial Report. Communications of the ACM Vo1. 12, No. 5, May 1969.
3. Clarification of FORTRAN Standards - Second Report. Communications of the ACM Vo1. 14, No. 10, October 1971.
4. Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. NBS. M. Abramowitz and Irene A. Stegun, editors, Applied Mathematical Series 55, 1966.

## A. OPERATING PROCEDURES

The NBS FORTRAN Test Programs are designed to test the acceptance of the ASA FORTRAN Standard X3.9-1966 language definition by a FORTRAN processor.

The Test Programs are comprised of 116 test units and approximately 14,500 card images.

A1. Organization of Tests and Facilities Requirements
The FORTRAN Test Programs are presented in two forms, one for execution on small FORTRAN processors identified as Version 1, and the other for large FORTRAN processors identified as Version 3.

The tests make use of a maximum of 3 I/O units. These I/O units are identified as integer variable names which are assigned values in the first executable statements in each executable program and the statements may be altered by the user. No subprogram directly references these variable names or values.

The variable names and their current values are:
IRVI - for input, usually a card reader, is defined as unit 5.
NUVI - for test results, usually a line printer is defined as unit 6.
INVI - for intermediate input/output data, usually a magnetic tape, is defined as unit 9.

The test programs should be run in numeric order. Test sequences contained in later test units may depend upon the successful execution of earlier test units.
a) Program Order

The FORTRAN Standard does not define the order of presentation of a main program, BLOCK DATA subprograms, FUNCTION or SUBROUTINE subprograms to a FORTRAN processor. This order is prescribed by the implementor and may vary from system to system. Systems also vary on the need for systems control cards or special cards preceding each subprogram. The test programs have been arranged with no intervening control cards but contain the necessary STOP and END cards as follows:

Main program
Subprograms (if required)
Data (if required)
Some systems may require a specific order for BLOCK DATA subprograms distinct from FUNCTION or SUBROUTINE subprograms.

The subprograms appear after the corresponding main program unit and before the data in the order listed in Section II-A3 for Version 1 and Section II-A4 for Version 3.

Each program is set up (except for the system control cards) for a FORTRAN compile-load-and-go execution.

The user is assumed to be familiar with the operating system control requirements necessary to perform a FORTRAN compilation.

These steps should be followed:

- Choose the appropriate control cards for a FORTRAN compile.
- Check the format and ordering of control cards carefully.
- In particular, check if any control cards are necessary for FUNCTION, BLOCK DATA, and SUBROUTINE subprograms.
- Check the particular FORTRAN system documentation for any special requirements for ordering of subprograms which may differ from the order of the test program.
- Check if the test program requires input data. Version 1 requires data for test segments $008,009,310$ and 312; Version 3, for all parts. Cards 1,3 and 5 of segment 008 for Version 1 and all parts for Version 3 may be prepared by the user and replace the sample cards supplied with the programs. Section II-A2.

Sections II-A3 contains the list of test programs for Version 1. The accompanying table identifies the I/O facilities requirements and other related information.

Sections II-A4 contains the list of test programs for each of the 14 Parts for Version 3 and identifies the I/O facilities requirements as well as a summary sheet related to all Parts.
b) Memory Requirements to Execute the Test Programs

During the development of the test systems ten different computing systems were used and the current set of tests were run on five major systems. Although no requirements for memory can be determined without experimentation, the largest test unit in Version 1 required less than 3,000 words of memory. When structured into 14 executable programs as Version 3, the largest program required less than 6,000 words of memory.
c) Time

The time to compile and execute the test programs varies with the power of the computer and the compiler. The test units, for the most part are straight line programs. During the debugging of the test program set of Version 3 on different large scale systems less than 30 seconds was required to compile and execute any one of the 14 Parts excluding card read and print time.

All data card images associated with the FORTRAN Test Programs are included with the program distribution. It is not essential to the performance of the test programs to prepare any input data, however, provisions have been made to facilitate the identification of the test program results for a given FORTRAN processor.

In Version 1 , test units $008,009,310$, and 312 require input data which is supplied with the programs. The first six (6) cards associated with test unit 008 cause a heading page to be produced for the program set.

In Version 3, all test Parts 1 to 14 include six (6) input cards as the total input data to that part, except Parts 1 and 13 which include additional input data cards supplied with the test programs.

These six cards permit information to be introduced by the user to identify: the computer, FORTRAN compiler identification, operating system level, date, etc., which describe the environment in which the test is performed. Cards 1, 3 and 5 must be replaced and prepared to introduce three (3) lines of print which precedes test unit 008 in Version 1 or is appended to the initial output page of each test part in Version 3.

The first 40 characters from each of three cards (cards 1, 3 and 5) are read and replace the Hollerith information supplied in each of three FORMAT statements. The first character of each card must be blank (for print carriage control) and the other 39 characters must be from the FORTRAN character set. Cards 2, 4, and 6 must remain as prepunched. These six cards are part of the first test unit (008) in Part 1, testing the replacement of Hollerith information in a FORMAT statement by a formatted READ, and the symmetry of interpretation of a terminal slash (/) in a FORMAT statement used for READ and WRITE, causing cards 1, 3, and 5 to be read and written, and cards 2,4 and 6 to be skipped on input and blank lines to be produced on output.

These six cards are not part of the test in parts other than Part 1 but are included for user output documentation only.

WARNING: The following four characters should be avoided in preparation of the three cards, because these characters differ in the punch card code for input preparation devices:

A3. List of Test Programs for Version 1
The I/O Unit numbers used in the Test Programs are:
Input (card reader) 5
Output (printer)6
Intermediate ..... 9

The following table identifies each of the 116 Test Programs for Version 1 and the associated subprograms.

Codes Used to Describe the Information in the Table

Column
1 M Main Program F External Function S Subroutine B BLOCK DATA

2 I Input Required
3 No. of Pages of Output

Column
4 X Intermediate Tape Required
5 C Blank Common Block
/ Special Blank Common
6 D DATA Statement Defined
7 No. of Cards per Segment

| Seg. Name | Test | Table |
| :---: | :---: | :---: |
| 000 | Directory of Test Programs | - - - - - 342 |
| 008 - FMTRW | Formatted Input/Output <br> 6 Identification Cards and 40 Data Cards | $\begin{array}{rlr} \text { M I } 8---529 \\ \text { - - - } \end{array}$ |
| 009 - AFRMT | A-Conversion 3 Data Cards |  |
| 010 - DATA2 | DATA Statement Use | M - 3 - - 74 |
| 003 - DATA1 | Test Format of DATA Statement | M - - - D 84 |
| 011 - AASGN | Real and Integer Arith Assignmt. Stmnts. | M - $3--268$ |
| 013 - DASGN | Simple D.P. Assignment Statements | M - 8-- 420 |
| 015 - CASGN | Simple Complex Assignment Statements | M - 9 - - 469 |
| 016 - LASGN | Logical Assignment Statements | M - 1 - - 106 |
| 017 - INTRL | Arithmetic Assignment Statements | M - 4 - - 185 |
| 020 - UGOTO | Unconditional GO TO Statements | M - 1 - - 69 |


| 021 - AGOTO | GO TO Assignment Statements | M - 1 - - 149 |
| :---: | :---: | :---: |
| 022 - CGOTO | Computed GO TO Statements | M - 1 - - 146 |
| $030-$ ARBAD | Basic Addition | M - 1-- 115 |
| 031 - ARFAD | Double Precision Addition | M - $1-\ldots 57$ |
| 032 - ARBSB | Basic Subtraction | M - $1-\ldots 67$ |
| 033 - ARFSB | Double Precision Subtraction | M - 1 - - 72 |
| 034 - ARBAS | Basic Addition and Subtraction | M - $1-\ldots 79$ |
| 035 - ARFAS | Addition and Subtraction of D.P. Values | M - 1 - - 60 |
| 036 - ARBMI | Multiplication of Integer Values | M - 1 - - 66 |
| 037 - ARBMR | Multiplication of Real Values | M - 1 - - 64 |
| 038 - ARFMD | Multiplication of D.P. Values | M - 1 - - 71 |
| 039 - ARBDV | Division of Integer and Real Values | M - $1-\ldots 78$ |
| 040 - ARFDV | Division of D.P. Values | M - 1 - - 66 |
| 041 - ARBEX | Exponentiation of Integer and Real Values | M - $1-\ldots 90$ |
| 042 - ARFEX | Exponentiation of D.P. Values | M - 1 - - 74 |
| 043 - ARBHI | Hierarchy of Operators and Parentheses | M - 1 - - - 177 |
| 050-SBB67 | Subscripts of Integer, Real Arrays v, k | M - 1 - - 79 |
| 051 - SBB45 | Subscripts of Int., Real Arrays $\mathrm{v}+\mathrm{k}, \mathrm{v}-\mathrm{k}$ | M - $1-$ - 87 |
| 052 - SBB13 | Subscripts of Int., Real Arrays $c^{*} v, c^{*} v+k$, $c^{*} v-k$ | M - 1 - - 112 |
| 053-SBF17 | Subscripts of D.P. Arrays $v, k, c^{*} v, c^{*} v+k$, c*v-k, v+k, v-k | M - 1 - - 79 |
| 054 - SIMIF | Arith. IF, Logical IF followed by GO TO | M - 1 - - 77 |
| 055 - IFABS | Intrinsic Functions ABS, IABS | M - 1 - - 64 |
| 056 - IFFLT | Intrinsic Function FLOAT | M - 1 - - 49 |
| 057 - IFFIX | Intrinsic Function IFIX | M - 1 - - 59 |
| 058 - IFSGN | Intrinsic Functions SIGN, ISIGN | M - 1 - - 82 |
| 059 - IFDAB | Intrinsic Function DABS | M - 1 - - 65 |


| 060 - IFTRN | Intrinsic Functions AINT, INT, IDINT | M - $1--107$ |
| :---: | :---: | :---: |
| 061 - IFMOD | Intrinsic Functions AMOD, MOD | M - 1 - - 84 |
| 062 - IFMAX | Intr. Funct. AMAXO, AMAXI, MAXO, MAXI, DMAXI | M - $2-$ - 248 |
| 063 - IFMIN | Intr. Funct. AMIN0, AMIN1, MIN0, MIN1, DMIN1 | M - $2-$ - 225 |
| 064 - IFDSG | Intrinsic Function DSIGN | M - 1 - - 58 |
| 065 - IFDIM | Intrinsic Functions DIM, IDIM | M - 1 - - 69 |
| 066 - IFSGL | Intrinsic Function SNGL | M - 1 - - 80 |
| 067 - IFREL | Intrinsic Function REAL | M - 1 - - 102 |
| 068 - IFIMG | Intrinsic Function AIMAG | M - 1 - - 129 |
| 069 - IFDBL | Intrinsic Function DBLE | M - 1-- - 57 |
| 070 - IFCPX | Intrinsic Function CMPLX | M - 1-- 61 |
| 071 - IFCJG | Intrinsic Function CONJG | M - 1 - - 66 |
| 072 - IFBMS | Integer and Real Intrinsic Functions | M - 1 - - 129 |
| 073 - IFFMS | Int., Real and D.P. Intrinsic Functions | M - 2 - - 181 |
| 080 - EXPON | Basic External Function EXP | M - 1 - - 60 |
| 081 - DEXPO | Basic External Function DEXP | M - 1 - - 68 |
| 082 - CEXPO | Basic External Function CEXP | M - $3-\ldots 98$ |
| 083 - LOGTM | Basic External Function ALOG | M - $1-\ldots 57$ |
| 084 - DPLOG | Basic External Function DLOG | M - 1 - - 67 |
| 085 - CXLOG | Basic External Function CLOG | M - 3 - - 106 |
| 086 - COLOG | Basic External Function ALOG10 | M - 1 - - 56 |
| 087 - DCLOG | Basic External Function DLOG10 | M - 1 - - 66 |
| 088 - SINUS | Basic External Function SIN | M - 1 - - 81 |
| 089 - DPSIN | Basic External Function DSIN | M 1 - - 82 |
| 090 - CSICO | Basic External Functions CSIN, CCOS | M - 1-- 65 |
| 091 - COSNS | Basic External Function COS | M - 1-- 82 |
| 092 - DPCOS | Basic External Function DCOS | M - 1 - - 81 |


| 094 - TANGH | Basic External Function TANH | M - 1 - - 57 |
| :---: | :---: | :---: |
| 095 - SQROT | Basic External Function SQRT | M - $1-\ldots 55$ |
| 096 - DSQRO | Basic External Function DSQRT | M - 1 - - 63 |
| 097 - CSQRO | Basic External Function CSQRT | M - $1-74$ |
| 098 - ARCTG | Basic External Function ATAN | M - 1 - - 58 |
| 099 - DACTG | Basic External Function DATAN | M - 1 - - 66 |
| $100-\mathrm{ACTG} 2$ | Basic External Function ATAN2 | M - 1 - - 56 |
| 101 - DATN2 | Basic External Function DATAN2 | M - 1 - - 66 |
| 102 - DMODA | Basic External Function DMOD | M - 1 - - 63 |
| 103.- CABSA | Basic External Function CABS | M - 1 - - 84 |
| 110 - BSFTS | Statement Functions - Integer and Real | M - 1 - - 74 |
| 005 - BSFDF | Statement Function Definition | M - - - 35 |
| 111 - FSFTS | Statement Funct. - D.P., Complex, Logical | M - 1 - - 108 |
| 006 - FSFDF | Statement Function Definitions | M - - - 58 |
| $140-\mathrm{CPXAD}$ | Addition and Subtraction of Complex | M 1 - - 76 |
| 141 - CPXMU | Multiplication of Complex Numbers | M - 1 - - 141 |
| 142 - CPXDV | Division of Complex Numbers | M 1 - - 83 |
| 143 - CPXEX | Exponentiation of Complex Numbers | M - 1 - 125 |
| 144 - CPXOP | Arithmetic Operations on Complex | M 1 - - 63 |
| 145 - CREAD | Add and Subtract Complex and Real Numbers | M 1 - - 67 |
| 146 - CREMU | Multiply Complex by Real Numbers | M 1 - - 62 |
| 147 - CREDV | Divide Complex by Real and the Reverse | M-1- - 58 |
| 148 - CREOP | Combined Operations on Complex and Real | M - 1 - - 66 |
| 149 - MISC3 | Blanks in, Cont. of Statement to Max Lines | M 1 - - 97 |
| 150 - MISC4 | Special Characters for Continuations | M - 1 - - 105 |

$160-\mathrm{BRFCP}$
400 - AFS
420 - BFS
430 - CFS
440 - DFS
450 - EFS
460 - FFS
161 - BIFCP
401 - IAFI
421 - IBFI
431 - ICFI
441 - IDFI
451 - IEFI
461 - IFFI
162 - FRFCP
402 - GFS
422 - HFS
432 - IRFS
442 - JRFS
452 - RFS
163 - FIFCP
403- IFI
423-JFI
433-KFI
443- LFI
453-MFI
164 - CFCCP
404 - AFC
414 - BFC
424 - CFC
434 - DFC
444 - EFC
454 - FFC
464 - HFC
165 - DPFCP
405 - AFD
415 - BFI
425 - CFD
435 - DFD
445 - EFD
455 - FFD
465 - GFD
475 - HFD
Real External Functions
M - 1 - - - 82
Real Argument F - - - - 010
Real Arguments
F - - - - - 10
Integer Argument
F - - - - 10
Integer Arguments
Array Name as Argument
F - - - - 11
F - - - - 11
Different Types of Arguments
Integer External Functions
F - - - - 15
M - 1 - - - 87
Real Argument
Real Arguments
Integer Argument
Integer Arguments
Array Name as Argument
Different Types of Arguments
Real External Functions
D.P. Argument
Complex Arguments
Logical Argument
External Procedure
Different Types of Arguments
Integer Extérnal Functions
D.P. Argument
Complex Arguments
Logical Argument
External Procedure
Different Types of Arguments
Complex External Functions
Real Argument
Integer Argument
Array Name as Argument
D.P. Argument
Complex Argument
Logical Argument
Different Types of Arguments
Double Precision External Functions
F - - - - 10
F - - - - - 10
F - - - - - 10
F - - - D 13
F - - - - 11
$\begin{array}{lll}\text { F - - - - - } & 11 \\ \text { F - - - } & 15\end{array}$
M-1 - C - 132
F - - - - - 11
F - - - - 12
F - - - - 16
F - - - - 11
F - - C - 29
M-1 - C - 123
F-1-C-123
F - - - - 11
F - - - - 12
$\begin{array}{lll}\text { F - - - - } & 12 \\ F--- & 16\end{array}$
F - - - - - 11
F-- C - 29
M - 1 - C - 132
F - - - - - 10
F - - - - 10
F - - - - 11
F - - - - 12
F - - - - 11
F - - - - 15
F - - C - 28
M - 1 - C - 135
Real Argument
Integer Argument
D.P. Arguments
Complex Argument
Logical Argument
External Procedure
Array Name as Argument
Different Types of Arguments
F - - - - - 10
F - - - - - 10
F - - - - 11
F - - - - 12
F _ _ _ - 12
F - - - - 16
F - - - - 11
F - - - - 12
F - - C - 32

| 166-BFCCP | Logical External Functions | M - 1-C - 144 |
| :---: | :---: | :---: |
| $406-\mathrm{AFB}$ | Real Argument | F - - - - 10 |
| 416 - BFB | Integer Argument | F - - - - 10 |
| 426 - CFB | D.P. Argument | F - - - - 11 |
| 436 - DFB | Logical Argument | F - - - 11 |
| 446 - EFB | Complex Argument | F - - - 12 |
| 456 - FFB | Array Name as Argument | F - - - - 12 |
| 466 -. GFB | External Procedure | F - - - - 11 |
| 476 - HFB | Different Types of Arguments | F - - C - 25 |
| 167 - SBRTN | Subroutine Subprogram | M-1-C-103 |
| 407 - AAQ | Integer, Real Variables, Array Elements | S - - - 23 |
| 417 - ABQ | Array Elements | S - - - - 13 |
| 427 - ACQ | No Argument List | S - - C - 21 |
| 168 - FSBRT | Subroutine Subprogram | M-1-C-153 |
| 408- ADQ | Different Types of Arguments | S - - - - 39 |
| 418- AEQ | Array Names and Integer Arguments | S - - - - 23 |
| 428- AFQ | No Argument List | S - - C - 41 |
| 169 - BLKDT | BLOCK DATA Test | M - 1 - - 71 |
| 409 - BLOKD | BLOCK DATA Subprogram | B - - - D 36 |
| 179 - BLKDA | BLOCK Data Test | M - 1 - - 70 |
| 419 - BLAKD | BLOCK DATA Subprogram | B - - - D 24 |
| 429 - BLBKD | BLOCK DATA Subprogram | B - - - D 17 |
| 439 - BLCKD | BLOCK DATA Subprogram | B - - - D 20 |
| 180 - UNFRW | Unformatted WRITE and READ | M - 1 X - - 133 |
| 182 - BACUP | BACKSPACE Tape | M - 1 X - - 74 |
| 190 - DOTRM | DO Loops - Terminal Statements | M - 1 - - 135 |
| 191 - DOLMT | DO Loops - Parameters as Variable Names | M - 1 - - 62 |
| 192 - DONSC | DO Loops - Completely Nested Nest | M - 1 - - 166 |
| 193 - DONSI | DO Loops - Incomplete DO, Exit by GO TO | M - 1 - - 60 |
| 194 - DONSX | D0 Loops - Extended Range | M - 1 - - 130 |
| 195 - DONML | DO Loops - Nested Nest | M - 1 - - 65 |
| 196 - DONIO | DO Loops - I/O Terminal Statements | M-1 X - - 91 |
| 197 - MORDO | DO Loops - I/O, Statmı. Ft., Intr Ft., CALL | M - $1 \mathrm{X}-\mathrm{-} 143$ |
| 005 - BSFDF | Statement Functions | M - - - - 35 |
| 412 - MDQ | Subroutine Subprogram | S - - - - 13 |
| 200 - SUBR1 | Subroutine - Operations Done at Sub Level | M-1 $\mathrm{XC}^{\text {C }} 52$ |
| 410 - SUBRQ | Subroutine Subprogram - No Arg. List | S - X C - 101 |


| $300-$ LOGIF | Logical IF Statements | M - 1 - - - 275 |
| :---: | :---: | :---: |
| 411 - SMCQ | Subroutine Subprogram | S - - - - 12 |
| 301 - BARIF | Arithmetic IF Statements - Integer, Real | M - 1 - - 175 |
| 302 - FARIF | Arithmetic IF Statements - D.P. | M - 1 - - 99 |
| 310 - IOFMT | Formatted READ/WRITE - Additional Features 38 Data Cards | $\begin{array}{r} \text { M } 5 \text { - }-310 \\ \text { - }-\ldots \end{array}$ |
| 312 - RDFMT | Formats in Arrays | M I 1 - D 201 |
| 462 - FMTQ | Subroutine Subprogram | S - - - 33 |
|  | 13 Data Cards | - - - - 13 |
| $350-\mathrm{MISC5}$ | Specifications for Program Form | M - 1 - - 156 |
| 351 - FUNMX | Basic External Functions - Trig Formulae | M - 1 - - 58 |
| 352 - NAMES | Names Resemble FORTRAN Verbs, Functions | M - 1 - - 79 |
| 413 - MAQQ | Subroutine (Intrinsic Function Names | S - - - 15 |
| 463 - MBQQ | Subroutine used as Variable Names in | S - - - 15 |
| 473 - AMQQ | Subroutine some Subrts. and as | S - - - D 21 |
| 483- BMQQ | Subroutine Functions in others) | S - - - 16 |
| 360 - SPEC2 | COMMON, DIMENSION, EQUIVALENCE | M - 1-/ - 169 |

Total Cards 14360

A4. List of Test Units by Parts for Version 3

FORTRAN TEST PROGRAMS SUMMARY INFORMATION FOR VERSION 3

| Part \# | \# of TEST UNITS | \# of SUBPROGRAMS | INPUT DATA* | INTERMEDIATE TAPE REQUIRED | Pgs of OUTPUT | \# of CARDS |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 1 | 4 |  | X |  | 16 | 1123 |
| 2 | 2 |  |  |  | 18 | 932 |
| 3 | 10 |  |  |  | 14 | 1076 |
| 4 | 13 |  |  |  | 14 | 1123 |
| 5 | 11 |  |  |  | 14 | 1153 |
| 6 | 9 |  |  |  | 11 | 912 |
| 7 | 13 |  |  |  | 18 | 997 |
| 8 | 12 |  |  |  | 13 | 951 |
| 9 | 11 |  |  |  | 12 | 971 |
| 10 | 5 | 29 |  |  | 6 | 1031 |
| 11 | 5 | 23 |  |  | 6 | 1090 |
| 12 | 12 | 5 |  | X | 13 | 1433 |
| 13 | 5 | 2 | X |  | 10 | 1190 |
| 14 | 4 | 4 |  |  | 5 | 579 |
|  | 116 | 63 |  |  | 170 | 14561 |

*Input data other than the 6 cards which are appended to each Part for user output documentation
input unit \#5 = card reader
output unit $\# 6=$ printer
intermediate unit \#9

Segment \# and Name
000
001 SPECS
003 DATA1

1. 007 IODEF
2. 008 FMTRW*
3. 009 AFRMT*

010 DATA2*
011 AASGN*

## Test

Special Documentation
Specifications needed for Part 1 Test Format of DATA Statement I/O Unit Assignment Statements Formatted Input/Output
A-Conversion
DATA Statement Test
Real and Integer Arithmetic Assignment Statemen

Input 49 cards - prepare 3 cards (cards 1, 3, and 5) Unit \#5
Output - Print 16 pages
*Produce Output
Note 1 The first 6 input cards (user prepared cards 1, 3, and 5) are associated with seg. 007 program element, however, performing tests under segment 008. See Data Preparation Section II-A-2. These 6 cards are part of the test for this part only. Inclusion of these cards in later Part tests is for user output documentation only.

Note 240 input cards - for test of seg. 008
Note 303 input cards - for test of seg. 009

Segment \# and Name
000
001 SPECS

1. 007 IODEF

013 DASGN*
015 CASGN*

Test
Special Documentation
Specifications needed for Part 2
I/O Unit Assignment Statements
Simple Double Precision Assignment Statements
Simple Complex Assignment Statements

Input 6 cards - prepare 3 cards (1, 3, and 5) Unit \#5
Output Print 18 pages Unit \#6
*Produce Output
Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 2 are not part of the test, but are included for user output documentation only.

Segment \# and Name
000
001 SPECS

1. 007 IODEF

016 LASGN*
017 INTRL*
020 UGOTO*
021 AGOTO*
022 CGOTO*
030 ARBAD*
031 ARFAD*
032 ARBSB*
033 ARFSB*
034 ARBAS*

Test
Special Documentation
Specifications needed for Part 3
I/O Unit Assignment Statements
Logical Assignment Statements Arithmetic Assignment Statements Unconditional GO TO Statements
GO TO Assignment Statements
Computed GO TO Statements
Basic Addition-Integer and Real
Double Precision Addition
Basic Subtraction-Integer and Real
Double Precision Subtraction
Basic Addition and Subtraction-Integer and Real

Input 6 cards Unit \#5
Output Print 14 pages
*Produce Output

Unit \#6

Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 3 are not part of the test, but are included for user output documentation only.

Segment\# and Name
000
001 SPECS

1. 007 IODEF

035 ARFAS*

036 ARBMI *
037 ARBMR*
038 ARFMD*
039 ARBDV*
040 ARFDV*
041 ARBEX*
042 ARFEX*
043 ARBHI*
050 SBB67*
051 SBB45*
052 SBB13*
053 SBF17*

Test
Special Documentation
Specifications needed for Part 4
I/O Unit Assignment Statements
Addition and Subtraction of Double Precision Values
Multiplication of Integer Values
Multiplication of Real Values
Multiplication of Double Precision Values
Division of Integer and Real Values
Division of Double Precision Values
Exponentiation of Integer and Real Values
Exponentiation of Double Precision Values
Hierarchy of Operations and Parentheses
Subscripts of Integer and Real Arrays v, $k$
Subscripts of Integer and Real Arrays $\mathrm{v}+\mathrm{k}, \mathrm{v}-\mathrm{k}$
Subscripts of Integer and Real Arrays $c+v$, $c^{*} v+k, c^{*} v-k$
Subscripts of Double Precision Arrays $v, k, c^{*} k$, $c^{*} v+k, \quad c^{*} v-k, \quad v+k, v-k$

Input 6 cards
Output Print 14 pages
*Produce Output

Unit \#5
Unit \#6

Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 4 are not part of the test, but are included for user output documentation only.

## VERSION 3 PART 5 MAIN PROGRAM (Intrinsic Function Tests)

Segment\# and Name
000
001 SPECS
1.

007 IODEF
054 SIMIF*
055 IFABS*
056 IFFLT*
057 IFFIX*
058 IFSGN*
059 IFDAB*
060 IFTRN*
061 IFMOD*
062 IFMAX*

063 IFMIN*

064 IFDSG*

Test
Special Documentation
Specification needed for Part 5
I/O Unit Assignment Statements Arithmetic IF, logical IF followed by GO TO ABS, IABS (Absolute Value Functions) FLOAT (Convert from Integer to Real) IFIX (Convert from Real to Integer)
SIGN, ISIGN (Transfer of Sign)
DABS (Absolute Value)
AINT, INT, IDINT (Truncation)
AMOD, MOD (Remaindering)
AMAX0, AMAX1, MAX0, MAX1, DMAX1 (Choose
Largest Value)
AMINO, AMIN1, MINO, MIN1, DMIN1 (Choose Smallest Value)
DSIGN (Transfer of Sign)

Input 6 cards Unit \#5
Output Print 14 pages
Unit \#6
*Produce Output
Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 5 are not part of the test, but are included for user output documentation only.

## VERSION 3 PART 6 MAIN PROGRAM (Intrinsic Functions)

Segment \# and Name
000
001 SPECS

1. 007 IODEF

065 IFDIM*
066 IFSGL*
067 IFREL*
068 IFIMG*
069 IFDBL*
070 IFCPX*
071 IFCJG*
072 IFBMS*
073 IFFMS*

## Test

Special Documentation
Specifications needed for Part 6
I/O Unit Assignment Statements
DIM, IDIM (Positive Differences)
SNGL (Obtain most Significant part)
REAL (Obtain Real Part of Complex Argument)
AIMAG (Obtain Imaginary Part of Complex Number)
DBLE (Express Real Argument in D.P. Form)
CMPLX (Express Two Real Arg. in Complex Form)
CONJG (Obtain Conjugate of a Complex Number)
All Intrinsic Functions-Real and Integer
All Intrinsic Functions-Real, Integer and D.P

Input 6 cards
Unit \#5
Output Print 11 pages
*Produces Output

Note 1 Prepare replacement cards for cards 1,3 , and 5 as described in Data Preparation Section II-A-2. These cards in Part 6 are not part of the test, but are included for user output documentation only.

Segment \# and Name

## 000

001 SPECS
1.

007 IODEF
080 EXPON*
081 DEXPO*
082 CEXPO*
083 LOGTM*
084 DPLOG*
085 CXLOG*
086 COLOG*
087 DCLOG*
088 SINUS*
089 DPSIN*
090 CSICO*
091 COSNS*
092 DPCOS*

Test
Special Documentation
Specifications needed for Part 7
I/O Unit Assignment Statements
Basic External Function - EXP
Basic External Function - DEXP
Basic External Function - CEXP
Basic External Function - ALOG
Basic External Function - DLOG
Basic External Function - CLOG
Basic External Function - ALOG10
Basic External Function - DLOG10
Basic External Function - SIN
Basic External Function - DSIN
Basic External Function - CSIN and CCSIN
Basic External Function - COS
Basic External Function - DCOS

Input 6 cards Unit \#5
Output Print 18 pages Unit \#6
*Produces Output
Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 7 are not part of the test, but are included for user output documentation only.

|  | Segment \# and Name |  | Test |
| :---: | :---: | :---: | :---: |
|  | 000 |  | Special Documentation |
|  | 001 | SPECS | Specifications needed for Part 8 |
|  | 005 | BSFDF | Statement Function Definitions for Segment 110 |
|  | 006 | FSFDF | Statement Function Definitions for Segment 111 |
| 1. | 007 | IODEF | I/O Unit Assignment Statements |
|  | 094 | TANGH* | Basic External Function - TANH |
|  | 095 | SQROT* | Basic External Function - SQRT |
|  | 096 | DSQRO* | Basic External Function - DSQRT |
|  | 097 | CSQRO* | Basic External Function - CSQRT |
|  | 098 | ARCTG* | Basic External Function - ATAN |
|  | 099 | DACTG* | Basic External Function - DATAN |
|  | 100 | ACTG2* | Basic External Function - ATAN2 |
|  | 101 | DATN2* | Basic External Function - DATAN2 |
|  | 102 | DMODA* | Basic External Function - DMOD |
|  | 103 | CABSA* | Basic External Function - CABS |
|  |  | BSFTS* | Statement Functions (Real and Integer) |
|  |  | FSFTS* | Statement Functions (D.P., Logical and Complex) |
|  | Input 6 cards <br> Output Print 13 pages <br> *Produce Output |  | Unit \#5 |
|  |  |  | Unit \#6 |
|  |  |  |  |
| Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 8 are not part of the test, but are included for user output documentation only. |  |  |  |

Segment \# and Name
000
001 SPECS

1. 007 IODEF

140 CPXAD*
141 CPXMU*
142 CPXDV*
143 CPXEX*
144 CPXOP*
145 CREAD*
146 CREMU*
147 CREDV*
148 CREOP*
149 MISC3*
150 MISC4*

Input 6 cards
Output Print 12 pages
*Produce Output

Test
Special Documentation
Specifications needed by Part 9
I/O Unit Assignment Statements
Addition and Subtraction of Complex Numbers
Multiplication of Complex Numbers
Division of Complex Numbers
Exponentiation of Complex Numbers
Arithmetic Operations on Complex Numbers
Addition, Subtraction of Complex, Real Numbers
Multiplication of Complex by Real Numbers
Division of Real, Complex by Complex, Real Numbers
Combined Operations on Complex and Real Numbers Blanks in, and Continuation of Statements to Maximum Lines
Special Characters for Continuation Lines

Unit \#5
Unit \#6

Note 1 Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 9 are not part of the test, but are included for user output documentation only.

Segment \# and Name
000
001 SPECS

1. 007 IODEF

160 BRFCP*
161 BIFCP*
162 FRFCP*
163 FIFCP*
164 CFCCP*
Subprograms
400 AFS
420 BFS
430 CFS
440 DFS
450 EFS
460 FFS
Subprograms
401 IAFI
421 IBFI
431 ICFI
441 IDFI
451 IEFI
461 IFFI
Subprograms
402 GFS
422 HFS
432 IRFS
442 JRFS
452 RFS
Subprograms
403 IFI
423 JFI
433 KFI
443 LFI
453 MFI
Subprograms
404 AFC
414 BFC
424 CFC
434 DFC
444 EFC
454 FFC
464 HFC

## Test

Special Documentation
Specifications needed for Part 10
I/O Unit Assignment Statements
External Function Test - Real
External Function Test - Integer
External Function Test - Real - All Argument Types
External Function Test - Integer - All Argument Types
External Function Test - Complex
Used with Segment 160 - Real Function
Real Argument
Real Arguments
Integer Argument
Integer Arguments
Array Name
Integer and Real Arguments
Used with Segment 161 - Integer Function
Real Argument
Real Arguments
Integer Argument
Integer Arguments
Array Name
Integer and Real Arguments
Used with Segment 162 - Real Function
Double Precision Arguments
Complex Arguments
Logical Argument
Argument - External Procedure
Different Types of Arguments
Used with Segment 163 - Integer Function
Double Precision Arguments
Complex Arguments
Logical Arguments
Argument - External Procedure
Different Types of Arguments
Used with Segment 164 - Complex Function
Real Argument
Integer Argument
Array Name
Double Precision Argument
Complex Argument
Logical Arguments
Different Types of Arguments

Note 1 Prepare replacement cards for cards 1,3 , and 5 as described in Data Preparation Section II-A-2. These cards in Part 10 are not part of the test, but are included for user output documentation only.

Test

000
001 SPECS

1. 007 IODEF

165 DPFCP*
166 BFCCP*
167 SBRTN *
168 FSBRT*
169 BLKDT*
Subprograms
405 AFD
415 BFD
425 CFD
435 DFD
445 EFD
455 FFD
465 GFD
475 HFD
Subprograms
406 AFB
416 BFB
426 CFB
436 DFB
446 EFB
456 FFB
466 GFB
476 HFB

Subprograms
407 AAQ
417 ABQ
427 ACQ
Subprograms
408 ADQ
418 AEQ
428 AFQ

Subprogram
409 BLOKD

Input 6 cards
Output Print 6 pages
*Produces Output

Special Documentation
Specifications needed for Part 11
I/O Unit Assignment Statements
External Function Test - Double Precision
External Function Test - Logical
Subroutine Subprogram Test
Subroutine Subprogram Test
Block Data Subprogram Test
Used with Segment 165 - D.P. Function
Real Argument
Integer Argument
Double Precision Argument
Complex Argument
Logical Argument
Argument - External Procedure
Array Name
Different Types of Arguments
Used with Segment 166 - Logical Function
Real Arguments
Integer Arguments
Double Precision Argument
Logical Argument
Complex Argument
Array Name
Argument.- External Procedure
Different Types of Arguments
Used with Segment 167 - Subroutine Subprogram Integer and Real variables and Array Elements Array Elements
No Argument List - Arguments passed thru Common
Used with Segment 168 - Subroutine Subprogram Different Types of Arguments
Array Names and Integer Arguments
No Argument List - Arguments Passed through Common

Used with Segment 169 - Block Data Test Block Data Subprogram

Note 1 Prepare replacement cards for cards 1,3 , and 5 as described in Data Preparation II-A-2. These cards in Part 11 are not part of the list, but are included for user output documentation only.

Segment \# and Name
000
001 SPECS
005 BSFDF

1. 007 IODEF

179 BLKDA*
180 UNFRW*
182 BACUP*
190 DOTRM*
191 DOLMT*
192 DONSC*
193 DONSI*
194 DONSX*
195 DONML*
196 DONIO*
197 MORDO*
200 SUBR1*
Subprogram
410 SUBRQ
Subprogram
412 MDQ
Subprograms
419 BLAKD
429 BLBKD
439 BLCKD

Test
Special Documentation
Specifications needed for Part 12
Statement Function Definitions used with Segment 197
I/O Unit Assignment Statements
Block Data Test
Unformatted Read and Write
Backspace Tape
DoLoops - Terminal Statements
DoLoops - Parameters integer variable names
DoLoops - Completely Nested Nest
DoLoops - Incomplete Looping
DoLoops - Extended Range
DoLoops - Nested Nests
DoLoops - I/O Terminal Statements
DoLoops - I/O, Intrinsic Functions, CALL includer
Subroutine Called
Used with Segment 200

Used with Segment 197
Subroutine Subprogram
Used with Segment 179 - Block Data Test
Block Data Subprogram
Block Data Subprogram
Block Data Subprogram

Input 6 cards
Output Print 13 pages
Intermediate tape *Produces Output

Unit \#5 Unit \#6 Unit \#9

Note 1 Prepare replacement cards for cards 1,3 , and 5 as described in Data Preparation Section II-A-2. These cards in Part 12 are not part of the test, but are included for user documentation only.

Segment \# and Name
000
001 SPECS

1. 007 IODEF

300 LOGIF*
301 BARIF*
302 FARIF*
2. 310 IOFMT*
3. 312 RDFMT*

Subprogram
411 SMCQ
Subprogram
462 FMTQ

Test
Special Documentation
Specifications needed for Part 13
I/O Unit Assignment Statements
Logical If Statements
Arithmetic If Statements (Integer and Real Expressions)
Arithmetic If Statements
Formatted Read and Write, additional properties of
Formats in Arrays
Used with Segment 300
Subroutine
Used with Segment 312
Subroutine
Input 57 cards - prepare 3 cards (cards 1, 3, and 5) Unit \#5 Output Print 10 pages Unit \#6 *Produce Output
Note 1 The first 6 input cards in Part 13 (user prepared cards 1, 3, and 5) are not part of the test, but are included for output documentation only. See Data Preparation Section II-A-2.
Note 238 input cards - for test of seg. 310
Note 313 input cards - for test of seg. 312

|  | Segm | nt \# and Name | Test |
| :---: | :---: | :---: | :---: |
|  | 000 |  | Special Documentation |
|  | 001 | SPECS | Specifications needed for Part 14 |
| 1. | 007 | IODEF | I/O Unit Assignment Statements |
|  | 350 | MISC5* | Specifications for Program Form (Test) |
|  | 351 | FUNMX* | Basic External Functions using Trig Formula |
| 2. | 352 | NAMES* | Names resembling FORTRAN Verbs and Function |
|  | 360 | SPEC2* | Common, Dimension and Equivalence |
| Subprogram |  |  | Used with Segment 352 |
|  | 413 | MAQQ | Subroutine Called from NAMES |
|  |  | MBQQ | Subroutine Called from NAMES |
|  | 473 | AMQQ | Subroutine Called from NAMES |
|  | 483 | BMQQ | Subroutine Called from NAMES |
| Input 6 cards |  |  | Unit \#5 |
| Output Print 5 pages *Produce Output |  |  | Unit \#6 |
|  |  |  |  |
| Note 1 P |  | Prepare replacement cards for cards 1, 3, and 5 as described in Data Preparation Section II-A-2. These cards in Part 14 are not part of the test, but are included for user output documentation only. |  |
|  |  |  |  |
|  |  |  |  |
| Note 2 |  | is test may cau run independen | ies in some compilers and may have to tests. |

## B. PROCEDURES FOR ISOLATING TEST UNIT FAILURES FROM VERSION 3

The following procedures assume the NBS FORTRAN Test Programs, Version 3, are being used with the programs on interpreted punch cards rather than from magnetic tape.

B1. Deleting a Test Unit
If any part fails to complete the execution of all the test units within the part, the printed results will probably contain at least the heading of the segment which failed and no test unit beyond this point will have been completed. If the test which failed is not the last one in a part, remove the cards which define the particular test and proceed with the test with this test unit deleted. Parts 10-14 contain subprograms which may have to be removed if a test failure occurs in these parts.

B2. Creating a Single Test from a Deleted Unit
Each test unit may be run independently by either of the following two methods.
a) Append the FORTRAN specification statements which appear at the beginning of the appropriate part to the beginning of the test unit to be retested. Include the one (or two) Input-Output assignment statements appearing as a segment 007 card within the first test unit of the part. This statement should be inserted into the test unit to be retested as the first executable statement, which can be located by the corresponding statement appearing in the test unit as a comment card with $C=$ in the first two locations. Supply a STOP statement and an END card at the end of the test unit main program. Although specifications not used within this test unit may cause diagnostics to appear as warning messages to non referenced data names, the program test unit is still a standard conforming FORTRAN program.
b) Isolate the test unit. Check the initial comment lines in the listing related to the part containing the test unit. If any additional segments are required to run this test unit, they are identified. For every card in the isolated test unit containing a "C =" in columns 1 and 2, duplicate the cards with the "C $=$ " changed to blanks and omit punching columns $73-80$ of the card. Return the comment cards to their original locations in the deck with the corresponding FORTRAN created statements immediately below the comment card. These "C = " comment indicators have been appended to what otherwise would be FORTRAN specification statements, I/O assignment, STOP statements and END lines. Omitting the duplication of columns 73-80 will make it easier to remove these cards when the test unit is returned to its original state for reinsertion into its appropriate location in the test part.

Test units numbered $008,009,310$ and 312 are the only units which require input data cards to perform the test.
C. SAMPLE TEST RESULTS

C1. Interpreting the Test Results
An attempt was made in the design of these tests to produce test results which were as much as possible self explanatory. Wherever a value of zero could not be created by the addition or subtraction of a constant from the calculated result, a Hollerith equivalent precedes the test result for comparison purposes.

The effects of conversion, precision, and exponent range are minimized by the use of values which are integer and fractional powers of 2 where the choice of values affected the test results. Other results are truncated to minimize the effects of differences in systems precision.

The ASA FORTRAN Standard does not prescribe the external output form for a Real or Double Precision zero. Systems implementors have used a wide variety of forms with and without + or - signs. Some implementations employ a + or - sign with the Fw.d format field descriptor when the printed value is zero to denote a truncated value whose sign corresponds. to the sign of the original value. Expect variations in the form of zero.

The ASA FORTRAN Standard permits the implementor a choice of form for output.
A positive sign is not required.
A leading zero before the decimal point for $E$ and $D$ conversion is not required.

The following exponent forms are equivalent and correct for E conversion:
E+02
E 02
+002
The following exponent forms are equivalent and correct for $D$ conversion:
D+13
D 13
E+13
E 13
+013
In the test program results where D conversion is used on output and the expected output value is stipulated to be zero, any value containing a negative exponent of $\mathrm{D}-13$ or mathematically less is considered to be zero。 The test units containing the Basic External Functions do not attempt to test either the range or the precision of these functions. A selected set of arguments to these functions is presented for the purpose of determining only whether the function name referenced is actually the function delivered.

The following limits have been set for constants in this test program set:

```
Integer 5 digits
Real }7\mathrm{ digits
Double Precision 14 digits
Complex 7 digits (each half)
Hollerith 2 characters except in segment 009 which tests A-conversion
                                    for l to 4 characters and 26 characters for the
                                    truncation test.
```

Where the precision of a FORTRAN processor for a REAL datum approaches the limit established for a Double Precision datum (14 digits) it will be necessary to increase the number of digits printed out for the test of the intrinsic function SNGL (test unit 066) to obtain meaningful test results.

C2. Test Results
The following test results were obtained from actual execution of Version 1 or Version 3 of the NBS FORTRAN Test Program set. These results are a composite set of output pages derived from five FORTRAN processors showing various forms for zero and differences in exponent form.


END OF VERTICAL SPACING TEST
$\begin{array}{lll}\text { BEGIN I CONVERS } \\ \text { EACH PAIR OF LI } \\ \text { LINE } 1 \text { OF } & \text { EACH } \\ & & \\ 999 & & \\ 999 & & \\ 5555 & 4444 & \\ 5555 & 4444 & \\ & & \\ 666 & 777777 & 8 \\ 666 & 777777 & 8\end{array}$
333333111112222222255555444444444444 333333111112222222255555444444444444

BEGIN F CONVERSION TEST EACH PAIR OF LINES SHOULD BE IDENTICAL
7.7123456 .7
7.7123456 .7
8.889.9997.123456
8.889.9997.123456
5.44446 .5555533 .133 .133 .133 .1444 .1
5.44446 .5555533 .133 .133 .133 .1444 .1

```
5555.15555.1 66666.166666.1 44.22
5555.15555.1 66665.166666.1 44.22
```

2.12.12.12.12.1666.3334.3334.3334.333
2.12 .12 .12 .12 .1666 .3334 .3334 .3334 .333

## BEGIN E CONVERSION TEST

EACH PAIR OF LINES SHOULD BE IDENTICAL

```
-0.1E+01 0.22E-01
    -.1E+01 -22E-01
    0.333E+02 0.4444E+03
    .333E+02 .4444E+03
-0.55555E-03 0.666666E+00
    -.55555E-03 .655666E+00
    0.9876543E+12
        .9876543E+12
```

BEGIN COMPLEX CONVERSION TEST EACH GROUP SHOULD BE IDENTICAL
$1.0 \quad 5.5$
1.05 .5
22.066 .6
22.066 .6
$33.1234 \quad 55.0789$
33.1234 55.0789
123.00456 .88
$123.00 \quad 456.88$

```
    0.123E+01 0.987E+01
        .123E+01 -987E+01
-0.2345E+02-0.6879E+02
    -.2345E+02 -.6879E+02
    0.7E+03 0.4E+03
        .7E+03 -4E+03
    0.9876543E-04 0.1357913E-04
        .9876543E-04 .1357913E-04
```

$19.34 \quad 0.2468 E+02$
$19.34 \quad .2468 \mathrm{E}+02$
$0.755 E+02 \quad 87.5$
$.755 \mathrm{E}+0287.6$

```
43.96 0.5407E+02
43.95 . 5407E+02
43.95 . 5407E+02
43.95 . 5467E+02
```

BEGIN D CONVERSION TEST EACH GROUP SHOULD BE IDENTICAL

$$
\begin{array}{r}
0.10+06 \\
.10+06
\end{array}
$$

-0.3340-04
-.3340-04
-. $3340-04$
$0.75576540+00$
$.76576540+00$

## $0.123456789010+10$ $.123456789010+10$ <br> 0.987654321098760-01 <br> $.987654321098760-01$ <br> $.987654321098760-11$ <br> . $987654321098760-31$

$-0.5555555420+03$
$-.5555555420+03$
$-.5555555420+03$

BEGIN L CONVERSION TEST LINES BELOW SHOULD BE IDENTICAL

| $T$ | $F$ | $F$ | $T$ | $T$ | $F T F$ |
| :--- | :--- | :--- | :--- | :--- | :--- |
| T | $F$ | $F$ | $T$ | $T$ | $F T F$ |

TEST UNSUBSCRIPTED ARRAY NAMES IN I/ O LISTS. EACH GROUP OF LINES SHOULD BE IOENTICAL.
$9 \cdot 91 \cdot 19 \cdot 92 \cdot 29 \cdot 93 \cdot 39 \cdot 94 \cdot 4$
$9.91 .19 \cdot 92.29 .93 \cdot 39 \cdot 94 \cdot 4$
$9.91 .19 .92 .29 .93 .39 \cdot 94.4$
-9.9-9.9-9.9-9.9
$-9.9-9 \cdot 9-9.9-9.9$
$-0.990+01-0.990+01-0.990+01-0.990+01$
$=.99 D+01-.990+01-.990+01=.990+01$
$-990+01-990+01=.990+01=.990+01$
9999999999
9999999999

| $0.990+01$ | $0.990+01$ | $0.990+01$ | $0.990+01$ |  |  |  |
| ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| $.990+01$ | $.990+01$ | $.990+01$ | $.990+01$ |  |  |  |
| 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 |
| .9 .9 | 0.9 |  |  |  |  |  |
| .9 | .9 | .9 | .9 | .9 | .9 | .9 |
| .9 | .9 | .9 | .9 | .9 | .9 | .9 |
| .9 | .9 | .9 | .9 | .9 | .9 | .9 |
| .9 | .9 | .9 |  |  |  |  |

```
TF
TF
tFTFTFTF
TFTFTFTF
99999999
99999999
    0.990+01
        .99+001
        .99+001
        .99+001
        -99+001
    .99+001
9.95.59.96.69.97.79.98.8
9.95.59.96.69.97.79.98.8
9.95.59.96.69.97.79.98.8
9.95.59.96.69.97.79.98.8
9999999999999999
9999999999999999
    TFFT
    TFFT
9.99.99.99.99.9
9.99.99.99.99.9
LEADING BLANK INSERTION TEST
EACH PAIR OF LINES SHOULD BE IDENTICAL
8
8
22
22
    22
    22
        22
        22
        22
        22
7.7
7.7
    8.88
    8.88
```

9.999
9.999
5.4444
5.4444
5. 55555
6.55555
7.123456
7.123456

```
0.21E+01
    -21E+01
    0.331E+02
        .331E+02
    0.4441E+03
        .4441E+03
        0. 55551E+04
            - 55551E+04
            0.666651E+05
            . 666651E+05
                    0.1234567E+06
                        .1234567E+06
```

$0.10+00$
$.10+00$
$0.10+00$
$.10+00$
$0.10+00$
$.10+00$
$0.10+00$
- $10+00$
$1.0 \quad 5.5$
$1.0 \quad 5.5$

$$
\begin{array}{ll}
9.9 & 5.5 \\
9.9 & 5.5
\end{array}
$$

$9.9 \quad 5.5$
9.95 .5
$\begin{array}{ll}1.0 & 5.5 \\ 1.0 & 5.5\end{array}$

TEST LOGICAL FIELOS WITH BLANKS LINES BELOW SHOULO BE IDENTICAL

```
    T F F F
TEST D = O, W=O+1 (PAIRS OF LINES
BELOW SHOULD BE IDENTICAL)
```

4444. 

4444 。
. 55555
. 55555

BEGIN G CONVERSION
EACH PAIR OF LINES SHOULD BE IDENTIC

| $.1235 E+05$ | 1235. | 123.5 |
| :--- | :--- | :--- |
| $.1235 E+05$ | 1235. | 123.5 |
|  |  |  |
|  |  |  |
| 12.35 | 1.235 | .1235 |
| 12.35 | 1.235 | .1235 |

SCALE FACTOR ON READ
IN ORDER OF FORMAT OCCURRENCE

| CARD | 9876.54 | $98.7654 E 2$ | 9876.54 |
| :--- | ---: | ---: | ---: |
| DESC | $2 P F B .3$ | $-2 P E 9.4$ | $F 9.4$ |
| TO BE | 98.7654 | $.9877 E+04$ | 987654.00 |
| IS | 98.7654 | $.9877 E+04$ | 987654.00 |
|  |  |  |  |
| CARD | 987.654 | $8647860-4$ | $86.4786 E 2$ |
| DESC | $0 P G 9.4$ | 09.4 | $-2 P E 9.4$ |
| TO BE | 987.654 | $.8648 D=02$ | $.8648 E+04$ |
| IS | 987.654 | $.85480-02$ | $.8648 E+04$ |
|  |  |  |  |
| CARD | 86.4786 | 8557.8700 | 9876.54 |
| DESC | $F 9.4$ | 09.4 | $2 P G 9.4$ |
| TO BE | 8647.860 | $.86580+04$ | 98.77 |
| IS | 8647.860 | $.8658 D+04$ | 98.77 |

SCALE FACTOR ON WRITE IN ORDER OF FORMAT OCCURRENCE

| CARD | 9.87655 | $98.7654 E 2$ | 9876.54 |
| :--- | ---: | ---: | ---: |
| DESC | $2 P F 12.2$ | $-2 P E 12.4$ | $F 12.4$ |
| TO BE | 987.65 | $.0099 E+06$ | 98.7654 |
| IS | 987.66 | $.0099 E+06$ | 98.7654 |
|  |  |  |  |
| CARD | 987.654 | $8647860-3$ | $86.4786 E 2$ |
| DESC | $1 P G 12.2$ | 012.4 | $-2 P E 12.4$ |
| TO BE | $9.88 E+02$ | $8.64790+02$ | $.0086 E+06$ |
| IS | $9.88 E+02$ | $8.6479 D+02$ | $.0086 E+06$ |
|  |  |  |  |
| CARD | 86.4786 | 8657.8600 | 9876.54 |
| DESC | $2 P F 12.2$ | $1 P 012.4$ | $2 P G 16.4$ |
| TO BE | 8647.86 | $8.6579 D+03$ | 9877. |
| IS | 8647.86 | $8.6579 D+03$ | 987. |

THE LAST THO LINES OF EACH
SET SHOULD BE THE SAME
FORMAT RESCAN - THE SECOND GROUP OF EACH SET SHOULD AGREE WITH THE FIRST

| 1 | 22 | 333 |  |  |
| :--- | :--- | :--- | :--- | :--- |
| 4 | 55 | 666 |  |  |
| 7 | 88 | 999 |  |  |
| 1 | 22 | 333 |  |  |
| 4 | 55 | 666 |  |  |
| 7 | 88 | 999 |  |  |
| 2 | FF | 4 | $\$ 8$ | 6 | 11

```
AFRMT - (OO9) AOCONVERSION
```

ASA REF 7.2.3.8
EACH PAIR OF LINES SHOULD BE IDENTICAL
FOR COMPUTERS STORING FOUR
OR MORE CHARACTERS PER WORD

ABCDEFGHIJKLMNOPQRSTUVWX YZ
ABCDEFGHIJKLMNOPQRSTUVWX YZ

```
mon/()中,05
E=*/()+1.S
```

$0123456789+A B 25(C)$
$0123456789+A B 25(C)$

TEST A CONVERSION ADDING BLANKS EAGH PAIR OF LINES SHOULD BE IDENTICAL

A


TEST A FIELD TRUNCATIUN
2ND LINE SHOULD PARTIALLY MATCH IST

ABCDEFGHIJKLMNOPQRSTUVWXYZ
VWXYZ



```
AASGN - (011) SIMPLE REAL AND INTEGER
        ARITHMETIC ASSIGNMENT STATEMENTS
ASA REF. - 7.1.1
LINE 1 OF EACH PAIR IS HOLLERITH INFORMATION
INTEGER RESULTS
```

| 1 | 12345 | 0 |
| ---: | ---: | ---: |
| 1 | 12345 | 0 |
| 2 | -3 | -8765 |
| 2 | -3 | -8765 |
| 6912 | 0 | -23 |
| 6912 | 0 | -23 |
| 4444 | 4321 | 45 |
| 4444 | 4321 | 45 |
| 2468 | -3123 | 0 |
| 2468 | -3123 | 0 |
| REAL RESULTS |  |  |


| 1.0 | 358.6724 | -2.0 |
| :---: | :---: | :---: |
| 1.0 | 358.6724 | -2.0 |
| 3.0 | -2714.250 | 29.30542 |
| 3.0 | -2714.250 | 29.30542 |
| 86.27 | 1034.2 | 0.0 |
| 86.27 | 1034.2 | .0 |
| 0.0 | 345.678 | -2.5 |
| .0 | 345.678 | -2.5 |
| -5.66 | 1.111111 | 1.0 |
| -5.66 | 1.111111 | 1.0 |
| -2.0 | 3.0 | 4.0 |
| -2.0 | 3.0 | 4.0 |
| 5.0 | -6.0 | 0.0 |
| 5.0 | -6.0 | .0 |
| 0.23 | -0.716 | -0.7 |
| .23 | -.716 | -.7 |
| 0.81 | 0.9 |  |


| $\begin{array}{r} 0.105 E+03 \\ .105 E+03 \end{array}$ | $\begin{array}{r} -0.76 E+02 \\ -.76 E+02 \end{array}$ | $\begin{array}{r} 0.3324 E+03 \\ .3324 E+03 \end{array}$ |
| :---: | :---: | :---: |
| $0.5132 E+01$ | $0.534 \mathrm{E}-02$ | $-0.1419 E+00$ |
| . $5132 \mathrm{E}+01$ | . $534 \mathrm{E}-02$ | -. 1419E+00 |
| $-0.99 E+03$ | $0.105210 \mathrm{E}+05$ | $0.456 E+02$ |
| -. $99 E+03$ | -105210E+05 | - $456 \mathrm{E}+02$ |
| $0.6652 E+03$ | $-0.529 E+03$ | $0.78564 E+04$ |
| - $6652 \mathrm{E}+03$ | -. $529 \mathrm{E}+03$ | . $78564 \mathrm{E}+04$ |
| -0.34567E+04 | $0.6162 E+04$ | $0.23 E+00$ |
| $=.34567 \mathrm{E}+04$ | . $6162 E+04$ | - $23 \mathrm{E}+00$ |
| $0.94333 E+01$ | 0.3524E-02 | $-0.7432 E+00$ |
| - $94333 E+01$ | . $3524 \mathrm{E}-02$ | -. $7432 \mathrm{E}+00$ |
| $0 \cdot 1 E+01$ | $0.123 E+05$ | $-0.11 E+05$ |
| -1E+01 | -123E+05 | -. 11E+05 |
| $0.144 E+02$ | $-0.12 E+00$ | $0.3645 \mathrm{E}+01$ |
| . $144 \mathrm{E}+02$ | $-.12 E+00$ | - $3645 \mathrm{E}+01$ |
| -0.200E+04 | $0.99 E+04$ | $0 \cdot 0 E+00$ |
| -. $200 \mathrm{E}+04$ | - $99 E+04$ | 0 - |
| $-0.1512 E+06$ | $0.214 \mathrm{E}+06$ | $0.34 E+01$ |
| -. 1512E+06 | . $214 \mathrm{E}+06$ | - $34 \mathrm{E}+01$ |
| -0.4E-01 | $0.53214 \mathrm{E}+01$ | $0.6 E+104$ |
| -. $4 \mathrm{E}-01$ | . $53214 \mathrm{E}+01$ | - $6 E+04$ |
| $0.72 E+06$ | -0.813E+04 | $0.234 \mathrm{E}+00$ |
| - $72 \mathrm{E}+06$ | -. $813 \mathrm{E}+04$ | - $234 \mathrm{E}+00$ |
| $-0.3 E+02$ | $0.44 E+01$ | $0.1 E+05$ |
| -. $3 E+02$ | - $44 \mathrm{E}+01$ | -1E+05 |
| $0.36 E-03$ | 0.9E-04 | -0.10E-02 |
| -36E-03 | -9E-04 | -. 10E-02 |
| $0.777 \mathrm{E}+01$ | $-0.29 E+03$ | $0.4 E+01$ |
| - $777 \mathrm{E}+01$ | $-.29 E+03$ | - $4 \mathrm{E}+01$ |
| $0.90 \mathrm{E}+01$ | $0.810 \mathrm{E}+00$ | $-0.7 E+03$ |
| - $90 \mathrm{E}+01$ | - $810 \mathrm{E}+00$ | -. $7 E+03$ |
| 0.62E+03 | $0.5310 E+01$ | $-0.442 E+02$ |
| -62E+03 | - $5310 \mathrm{E}+01$ | -. $442 \mathrm{E}+02$ |
| 0.3E-04 | 0.25E-03 | -0.163E-02 |
| -3E-04 | -25E-03 | -. 163E-02 |


| $0.709 E+06$ | $0.81842 E+05$ | $-0.9 E+06$ |
| :---: | :---: | :---: |
| $.709 E+06$ | $.81842 E+05$ | $-.9 E+06$ |
| $0.627 E+05$ | $0.53 E+05$ | $-0.4 E+05$ |
| $.627 E+05$ | $.53 E+05$ | $-.4 E+05$ |
| $0.1463 E+02$ | $0.2 E-02$ | $-0.355 E+02$ |
| $.1463 E+02$ | $.2 E-02$ | $-.355 E+02$ |
|  | $0.4072 E+07$ | $-0.61835 E+07$ |
| $0.29 E+07$ | $.4072 E+07$ | $-.61835 E+07$ |
| $.29 E+07$ | $0.3 E+03$ | $-0.1 E+04$ |
| $0.829 E+04$ | $.3 E+03$ | $-.1 E+04$ |
| $.829 E+04$ |  |  |
| $0.3404 E+00$ | $0.55 E-03$ | $-0.761 E+02$ |
| $.3404 E+00$ | $.55 E=03$ | $-.761 E+02$ |

FORTRANTESTPROGRAMS PREPARED BY NATIONAL BUREAU OF STANDARDS FOR USE ON LARGE FORTRAN PROCESSORS IN ACCORDANCE WITH ASA FORTRAN $\times 3.9-1966$ VERSION 3 PART 2

SAMPLE COMPUTER, FORTRAN COMPILER LFVEL OFERATJNG SYSTEM VERSIOP:

DATE, INSTALLATICN NAME





CASGN - (015) COMPLEX ASSIGNMENT STATEMENTS
ASA REFS. $=5.1 .1 .4$ 7.1.1.1

RESULTS
LINE 1 OF EACH GROUP IS
HOLLERITH INFORMATION
values in a group should be the same

| $0.222 \mathrm{E}+02$ | $0.3333 E+02$ |
| :---: | :---: |
| . $222 \mathrm{E}+02$ | - $3333 \mathrm{E}+02$ |
| - $222 \mathrm{E}+02$ | - $3333 E+02$ |
| - $222 \mathrm{E}+02$ | - $3333 \mathrm{E}+02$ |
| - $222 \mathrm{E}+02$ | . $3333 \mathrm{E}+02$ |
| $0.3956 E+03$ | $0.41067 E+04$ |
| - $3956 \mathrm{E}+03$ | . $41067 E+04$ |
| - $3956 \mathrm{E}+03$ | . $41067 \mathrm{E}+04$ |
| - $3956 \mathrm{E}+03$ | . $41067 E+04$ |
| - $3956 \mathrm{E}+03$ | - $41067 E+04$ |
| -0.1234567E+05 | -0.1234567E +04 |
| -. $1234567 E+05$ | -. $1234567 E+04$ |
| -. $1234567 \mathrm{E}+05$ | -. $1234567 E+04$ |
| -. $1234567 E+05$ | -. $1234567 E+04$ |
| -. $1234567 \mathrm{E}+05$ | -. 1234567E+04 |
| $0.89 E+01$ | $-0.91 E+01$ |
| - $89 \mathrm{E}+01$ | -. $91 E+01$ |
| - $89 E+01$ | -. $91 E+01$ |
| - $89 \mathrm{E}+01$ | -. $91 E+01$ |
| - $89 \mathrm{E}+01$ | -. $91 E+01$ |
| -0.263512E+04 | $0.4621 E+02$ |
| -. $263512 E+04$ | . $4621 E+02$ |
| -. $263512 E+04$ | . $4621 E+02$ |
| -. $263512 E+04$ | . $4621 E+02$ |
| -. $263512 \mathrm{E}+04$ | - $4621 E+02$ |
| $0.1 E+02$ | $0 \cdot 2 E+02$ |
| -1E+02 | - $2 E+02$ |
| -1E+02 | - $2 \mathrm{E}+02$ |
| $0 \cdot 3 E+03$ | $0.4 E+04$ |
| - $3 \mathrm{E}+03$ | . $4 \mathrm{E}+04$ |
| - $3 E+03$ | . $4 E+04$ |
| $-0.5 E+02$ | $-0.6 E+03$ |
| -. $5 E+02$ | -. $6 E+03$ |
| -. $5 E+02$ | $-.6 E+03$ |
| $0.71 E+02$ | -0.92E+02 |
| -71E+02 | -. 92E+02 |
| -71E+02 | -. 92E+02 |


| $-0.883 E+03$ | $0.1414 E+04$ |
| :---: | :---: |
| -. $883 \mathrm{E}+03$ | . $1414 \mathrm{E}+04$ |
| -. $883 \mathrm{E}+03$ | . $1414 \mathrm{E}+04$ |
| $0 \cdot 1 E+02$ | $0.562 E+03$ |
| - $1 E+02$ | - $562 \mathrm{E}+03$ |
| - $1 E+02$ | . $562 \mathrm{E}+03$ |
| $0.2002 E+04$ | $-0.983 E+03$ |
| - $2002 \mathrm{~L}+04$ | -.983E+ 03 |
| - $2002 \mathrm{E}+04$ | -. $983 \mathrm{E}+03$ |
| $0.461 E+03$ | $-0.165 E+03$ |
| - $451 \mathrm{E}+03$ | -. $165 \mathrm{E}+03$ |
| . $461 \mathrm{E}+03$ | -. 165E+03 |
| $-0.21 E+02$ | $0.122 E+03$ |
| -. $21 \mathrm{E}+02$ | . $122 \mathrm{E}+03$ |
| -. $21 \mathrm{E}+02$ | -122E+03 |
| 0.1E-02 | 0. $2 E-02$ |
| -1E-02 | - 2E-02 |
| -1E-02 | -2E-02 |
| $0.562 E+00$ | $0.562 E+00$ |
| - $552 \mathrm{E}+00$ | - $562 \mathrm{E}+00$ |
| . $562 \mathrm{E}+00$ | . $562 \mathrm{E}+00$ |
| $-0 \cdot 3 E+00$ | -0.3333333E+00 |
| -. $3 \mathrm{E}+00$ | -. $3333333 \mathrm{E}+00$ |
| -. $3 \mathrm{E}+00$ | -. $3333333 \mathrm{E}+00$ |
| $0.4 E+00$ | $-0.445 E+00$ |
| - $4 E+00$ | -. $445 \mathrm{E}+00$ |
| - $4 \mathrm{E}+00$ | -. $445 \mathrm{E}+00$ |
| -0.95E+00 | $0.95 E+00$ |
| -. $95 E+00$ | -95E+00 |
| -. 95E+00 | -95E+00 |
| 0.164239E-01 | $0.36 E+00$ |
| . $164239 \mathrm{E}-01$ | - $36 \mathrm{E}+00$ |
| . $164239 \mathrm{E}-01$ | - $36 \mathrm{E}+00$ |
| $0.21 E+00$ | $-0.3963 E+00$ |
| - $21 E+00$ | -. $3963 \mathrm{E}+00$ |
| - $21 \mathrm{E}+00$ | -. $3963 E+00$ |
| $0.3398 \mathrm{E}+00$ | $0.3398 \mathrm{E}+00$ |
| . $3398 \mathrm{E}+00$ | . $3398 \mathrm{E}+00$ |
| - $3398 \mathrm{E}+00$ | - $3398 \mathrm{E}+00$ |
| $-0.6 E+00$ | $0.6 E+00$ |
| -. $6 \mathrm{E}+00$ | - $6 E+00$ |
| -. $6 E+00$ | . $6 E+00$ |

$-0.883 E+03$
-. $883 \mathrm{E}+03$ . $1414 E+04$
$.1414 E+04$
$0.562 E+03$
-562E+03
$.562 E+03$
$-0.983 E+03$
-. $983 E+03$
-. $983 \mathrm{E}+03$
$-0.165 E+03$
$-.165 E+03$
$-.165 E+03$
$0.122 E+03$
-122E+03
$0.2 E-02$
-2E-02
$0.562 E+00$

- $562 E+00$
$.562 E+00$
$-0.3333333 E+00$
-. $3333333 E+00$
$-0.445 E+00$
-. $445 \mathrm{E}+00$
$-.445 E+00$
$0.95 E+00$
- $95 \mathrm{E}+00$
- $95 E+00$
$0.36 E+00$
- $36 E+00$
- $36 E+00$
$-0.3963 E+00$
-. $3963 \mathrm{E}+00$
$-.3963 E+00$
$0.3398 \mathrm{E}+00$
- $3398 \mathrm{E}+00$
-3398E + 00
$0.6 E+00$
. $6 E+00$

| $\begin{aligned} & 0.0 E+00 \\ & 0 . \end{aligned}$ | $\begin{array}{r} 0.1 E+01 \\ .1 E+01 \end{array}$ |
| :---: | :---: |
| $0.4562311 E+07$ | $0.789453 \mathrm{E}+06$ |
| . $4562311 \mathrm{E}+07$ | . $789453 \mathrm{E}+06$ |
| $0.449 E+06$ | $0.25 E+04$ |
| . $449 \mathrm{E}+06$ | - $25 E+04$ |
| $0.22223 E+07$ | $0.3332 E+05$ |
| . $22223 \mathrm{E}+07$ | - $3332 \mathrm{E}+05$ |
| $0 \cdot 3 E+01$ | 0. $3 E+01$ |
| - $3 E+01$ | - $3 \mathrm{E}+01$ |
| - $3 E+01$ | - $3 \mathrm{E}+01$ |
| $0.9376543 E+05$ | $0.8765432 \mathrm{E}+04$ |
| . $9876543 \mathrm{E}+05$ | - $8765432 \mathrm{E}+04$ |
| . $9876543 E+05$ | - $8765432 \mathrm{E}+04$ |
| $0.4444 \mathrm{E}+04$ | 0.55555E-02 |
| . $4444 E+04$ | - 55555E-02 |
| . $4444 \mathrm{E}+04$ | - 55555E-02 |
| 0.6E-04 | $0.77 E+07$ |
| - 6E-04 | - $77 \mathrm{E}+07$ |
| -6E-04 | - $77 \mathrm{E}+07$ |
| $0.142 E+03$ | $0.2667 E+02$ |
| . $142 \mathrm{E}+03$ | - $2667 \mathrm{E}+02$ |
| . $142 \mathrm{E}+03$ | - $2667 \mathrm{E}+02$ |
| -0.36923E+06 | -0.234E+03 |
| -. $36923 \mathrm{E}+06$ | -. $234 \mathrm{E}+03$ |
| -. $36923 \mathrm{E}+06$ | -. $234 \mathrm{E}+03$ |
| $0.21 E+03$ | -0.21E+03 |
| - $21 E+03$ | -. 21E+03 |
| - 21E+03 | -. $21 E+03$ |
| -0.5959E+03 | 0.4967E+03 |
| -. $5959 \mathrm{E}+03$ | . $4967 E+03$ |
| -. $5959 E+03$ | . $4967 E+03$ |
| $0.1 E+01$ | $0.1 E+01$ |
| -1E+01 | -1E+01 |
| - $1 E+01$ | -1E+01 |
| $-0.2 E+01$ | $-0.2 E+01$ |
| -. $2 E+01$ | -. $2 \mathrm{E}+01$ |
| -. $2 E+01$ | -. $2 \mathrm{E}+01$ |
| $0.492 E+01$ | -0.6527E+04 |
| - $492 E+01$ | -. 6527E +04 |
| . $492 \mathrm{E}+01$ | -. $6527 E+04$ |


| $-0.7371 E+06$ | 0.998E-01 |
| :---: | :---: |
| -. $7371 \mathrm{E}+06$ | -998E-01 |
| -. $7371 \mathrm{E}+06$ | - $998 \mathrm{E}-01$ |
| $0.477447 E+07$ | -0.93624E+00 |
| . $477447 \mathrm{E}+07$ | -. $93624 E+00$ |
| . $477447 \mathrm{E}+07$ | -. $93624 E+00$ |
| -0.846200E-02 | $0.133305+03$ |
| -.846200E-02 | -13330E+03 |
| -. $846200 \mathrm{E}-02$ | . $13330 \mathrm{E}+03$ |
| $0.770000 E+09$ | 0.81625E+08 |
| . $770000 \mathrm{E}+09$ | - $81625 E+08$ |
| . $770000 \mathrm{E}+09$ | - $81625 \mathrm{E}+08$ |
| $0.133400 \mathrm{E}+05$ | $0.37900 \mathrm{E}+06$ |
| . $133400 \mathrm{E}+05$ | - $37900 \mathrm{E}+06$ |
| . $133400 \mathrm{E}+05$ | - $37900 \mathrm{E}+06$ |
| $0.303000 E+06$ | $0.30000 \mathrm{E}+06$ |
| - $300000 \mathrm{E}+06$ | -30000E+06 |
| - $300000 \mathrm{E}+06$ | . $30000 \mathrm{E}+06$ |
| 0.299E-01 | 0.299E+02 |
| -299E-01 | - $299 \mathrm{E}+02$ |
| -299E-01 | - $299 \mathrm{E}+02$ |
| $0.1419 E+06$ | $0.1419 \mathrm{E}+02$ |
| . $1419 E+06$ | . $1419 \mathrm{E}+02$ |
| -1419E+06 | -1419E+02 |
| 0.76E-01 | $0.987 E+03$ |
| - $76 \mathrm{E}-01$ | . $987 \mathrm{E}+03$ |
| . $76 E-01$ | - $987 \mathrm{E}+03$ |
| $0.31 E+02$ | $0.4659 E+05$ |
| - $31 E+02$ | . $4659 \mathrm{E}+05$ |
| - $31 \mathrm{E}+02$ | . $4659 E+05$ |
| -0.728E+05 | $-0.93296 E+08$ |
| -. $728 \mathrm{E}+05$ | -. $93296 E+08$ |
| -. $728 \mathrm{E}+05$ | -. $93296 \mathrm{E}+08$ |
| $0 \cdot 6 E+07$ | $-0.6 E+07$ |
| - $6 E+07$ | -. $6 E+07$ |
| - $6 E+07$ | -. $6 E+07$ |
| $-0.7914 E+07$ | $0.16 E+07$ |
| -. $7914 \mathrm{E}+07$ | -16E+07 |
| -. $7914 \mathrm{E}+07$ | -16E+07 |
| $0.1 E+02$ | $0.1 E+02$ |
| -1E+02 | -1E+02 |
| -1E+02 | -1E+02 |


| -0.2E-01 | -0.2E-01 |
| :---: | :---: |
| -. $2 E-01$ | -. 2E-01 |
| -. 2E-01 | -. 2E-01 |
| 0.3E-02 | -0.3E+04 |
| - 3E-02 | -. $3 E+04$ |
| -3E-02 | -. $3 E+04$ |
| $-0.4 E+05$ | 0.4E-03 |
| -. $4 E+05$ | - 4E-03 |
| $-.4 E+05$ | -4E-03 |
| $0 \cdot 5 E+06$ | -0.5E-04 |
| - $5 E+06$ | -. $5 E-04$ |
| - $5 E+06$ | -. $5 E-04$ |
| -0.6E-05 | $0 \cdot 6 E+07$ |
| -. 6E-05 | - $6 \mathrm{E}+07$ |
| -. $6 \mathrm{E}-05$ | - $6 E+07$ |
| $0.39393 E+01$ | $0.62 E+04$ |
| - $39393 \mathrm{E}+01$ | - $62 E+04$ |
| - $39393 \mathrm{E}+01$ | - $62 \mathrm{E}+04$ |
| $0.9 E+00$ | $0.765765 \mathrm{E}+03$ |
| -9E+ 00 | - $765765 \mathrm{E}+03$ |
| - $9 E+00$ | - $765765 \mathrm{E}+03$ |
| $0.352 E+09$ | $0.35 E+03$ |
| - $352 \mathrm{E}+09$ | - $35 \mathrm{E}+03$ |
| - $352 \mathrm{E}+09$ | - $35 \mathrm{E}+03$ |
| $0.147626 E+00$ | 0.891E-14 |
| . $147626 \mathrm{E}+00$ | -891E-14 |
| -147626E + 00 | -891E-14 |
| C. 9E-07 | $0.9999 E+08$ |
| -9E-07 | - $9999 \mathrm{E}+08$ |
| -9E-07 | - $9999 E+08$ |
| 0.13E-04 | 0.13E-04 |
| -13E-04 | -13E-04 |
| -13E-04 | -13E-04 |
| $0.77 E+00$ | $0.77 \mathrm{E}+00$ |
| -77E+00 | - $77 \mathrm{E}+00$ |
| - $77 \mathrm{E}+00$ | - $77 \mathrm{E}+00$ |
| $0.878 \mathrm{E}+01$ | $-0.878 \mathrm{E}+01$ |
| - $873 \mathrm{E}+01$ | -. $878 \mathrm{E}+01$ |
| - $878 \mathrm{E}+01$ | -.878E+01 |
| $-0.9797 E+02$ | $0.9797 E+02$ |
| -.9797E+02 | -9797E+02 |
| -.9797E+02 | - $9797 \mathrm{E}+02$ |


| -0.10101E+15 | -0.10101E+15 |
| :---: | :---: |
| -. $10101 \mathrm{E}+15$ | -. $10101 \mathrm{E}+15$ |
| -. $10101 \mathrm{E}+15$ | -. $10101 \mathrm{E}+15$ |
| $0.68 \mathrm{E}+12$ | $0.357628 E+00$ |
| - $68 \mathrm{E}+12$ | - $357628 \mathrm{E}+00$ |
| - $68 \mathrm{E}+12$ | . $357628 \mathrm{E}+00$ |
| $0.798 \mathrm{E}-03$ | $0.76444 \mathrm{E}+00$ |
| . $798 \mathrm{E}-03$ | . $76444 \mathrm{E}+00$ |
| . $798 \mathrm{E}-03$ | . $76444 \mathrm{E}+00$ |
| -0.3247E+20 | $-0.2594 E+05$ |
| -. $3247 \mathrm{E}+20$ | -. $2594 E+05$ |
| -. $3247 \mathrm{E}+20$ | -. $2594 \mathrm{E}+05$ |
| -0.43599E-19 | -0.12E-04 |
| -. 435 99E-19 | -. 12E-04 |
| -. $43599 \mathrm{E}-19$ | -. 12E-04 |
| -0.6E-09 | $-0.6 E+09$ |
| -. $6 E-09$ | -. $6 E+09$ |
| -. 6E-09 | -. $6 \mathrm{E}+09$ |
| $-0.9119 E+06$ | 0.9119E-06 |
| -. $9119 E+06$ | - 9119E-06 |
| -. $9119 \mathrm{E}+06$ | . $9119 \mathrm{E}-06$ |
| $0.39426 E+02$ | -0.39426E-02 |
| - $39426 \mathrm{E}+02$ | -.39426E-02 |
| - $39426 \mathrm{E}+02$ | -. 39426 E-02 |
| 0.45E-12 | $0.45 E+12$ |
| . $45 E-12$ | - $45 \mathrm{E}+12$ |
| -45E-12 | - $45 \mathrm{E}+12$ |
| $0.4793 E+06$ | $0.3479 E+05$ |
| . $4793 \mathrm{E}+06$ | - $3479 \mathrm{E}+06$ |
| -4793E+06 | - $3479 \mathrm{E}+06$ |
| . $4793 \mathrm{E}+06$ | - $3479 \mathrm{E}+06$ |
| 0. $3682 \mathrm{E}+01$ | $0.8236 E+02$ |
| - $3682 \mathrm{E}+01$ | - $8236 \mathrm{E}+02$ |
| - $3682 \mathrm{E}+01$ | - $8236 \mathrm{E}+02$ |
| - $3682 \mathrm{E}+01$ | - $8236 \mathrm{E}+02$ |
| -0.2571E+09 | 0.1752E+09 |
| -. $2571 E+09$ | -1752E+09 |
| -. $2571 E+09$ | -1752E+09 |
| -. $2571 E+09$ | -1752E+09 |
| $0.1460 E+00$ | -0.1064E +05 |
| -1460E+00 | -. $1064 E+05$ |
| -1460E+00 | -. $1064 E+05$ |
| -1460E+00 | -. $1064 E+05$ |

```
0.1642390E-01
    -1642390E-01
    .1642390E-01
    .1642390E-01
    .1642390E-01
    .1642390E-01
0.4552311E+07
    .4562311E+07
    .4562311E+07
    .4562311E+07
    .4552311E+07
    .4552311E+07
```

$-0.6000000 E=05$
-. $6000000 \mathrm{E}-05$
-. $6000000 \mathrm{E}-05$
-. $6000000 \mathrm{E}=05$
-. $6000000 \mathrm{E}-05$
-. $6000000 \mathrm{E}-05$
$-0.9119000 E+06$
-. $9119000 \mathrm{E}+06$
-. $9119000 \mathrm{E}+06$
-. $9119000 \mathrm{E}+06$
-. $9119000 \mathrm{E}+06$
$-.9119000 E+06$
$0.3600000 E+00$

- $3600000 \mathrm{E}+00$
- $3600000 \mathrm{E}+00$
. $3600000 E+00$
- $3600000 \mathrm{E}+00$
-3600000E+00
$0.7894530 \mathrm{E}+06$
$.7894530 E+06$
$.7894530 E+06$
$.7894530 E+06$
$.7894530 E+06$
$.7894530 E+06$
$0.6000000 E+07$
. $6000000 \mathrm{E}+07$
- $6000000 \mathrm{E}+07$
. $5000000 \mathrm{E}+07$
- $6000000 \mathrm{E}+07$
-6000000E+07
$0.9119000 \mathrm{E}-06$
.9119000E-06
-9119000E-06
. $9119000 \mathrm{E}-06$
$.9119000 E-06$
.9119000E-06

EACH GROUP SHOULD BE IDENTICAL EXCEPT FOR THE SIGN OF THE FIRST TWO LINES
$0.3000000 \mathrm{E}+03$
$.300000 \mathrm{E}+03$
$-.3000000 \mathrm{E}+03$
$-.3000000 \mathrm{E}+03$
$-.3000000 \mathrm{E}+03$
$-.3000000 \mathrm{E}+03$
$-0.5000000 \mathrm{E}+02$
$-.500000 \mathrm{E}+02$
$.5000000 \mathrm{E}+02$
$.5000000 \mathrm{E}+02$
$.5000000 \mathrm{E}+02$
$.5000000 \mathrm{E}+02$
$0.7700000 \mathrm{E}+00$ $.7700000 \mathrm{E}+00$
-.7700000E+00
$-.7700000 E+00$
$-.7700000 E+00$
$-.7700000 \mathrm{E}+00$
$0.5000000 \mathrm{E}+06$ . $5000000 \mathrm{E}+06$
-. $5000000 \mathrm{E}+06$
-. $5000000 \mathrm{E}+06$
-. $5000000 \mathrm{E}+06$
$-.5000000 \mathrm{E}+06$
$0.4920000 E+01$ $.4920000 E+01$
-. $4920000 \mathrm{E}+01$
-. $4920000 \mathrm{E}+01$
$-.4320000 \mathrm{E}+01$
-. $4920000 \mathrm{E}+01$
-0.6000000E-05
-. $6000000 \mathrm{E}-05$
-6000000E-05
-6000000E-05
. $6000000 \mathrm{E}-05$
. $6000000 \mathrm{E}-05$
$0.4444000 E+04$
$.4444000 E+04$
$-.4444000 E+04$
-. $4444000 E+04$
$-.4444000 E+04$
$-.4444000 E+04$
$0.4000000 E+04$ . $4000000 \mathrm{E}+04$
$-.4000000 \mathrm{E}+04$
$-.4000000 \mathrm{E}+04$
-. $4000000 \mathrm{E}+04$
$=.4000000 \mathrm{C}+04$
$-0.6000000 \mathrm{E}+03$
$-.6000000 E+03$ -600000CE+03 -6001000E+03 -5000000E+03 - $6000000 \mathrm{E}+03$
$0.7700000 E+00$ $.7700000 E+00$
$-.7700000 \mathrm{C}+00$
$-.7700000 \mathrm{E}+00$
$-.7700000 \mathrm{E}+00$
$-.7700000 \mathrm{E}+00$
$-0.5000000 \mathrm{E}-04$ -. $5000000 \mathrm{E}-04$
. $5000000 \mathrm{E}-04$
. $5000000 \mathrm{E}-04$
-5000000E-04 -5000000E-04
$-0.6527000 E+04$
-. $6527000 E+04$
-6527000E +04
-6527000E + 04
-6527000E+04

- $5527000 E+04$
$0.6000000 E+07$
$.6000000 E+07$
-. $6000000 \mathrm{E}+07$
-. $6000000 \mathrm{E}+07$
-. $5000000 \mathrm{E}+07$
$-.6000000 E+07$
$0.5555500 \mathrm{E}-02$
. $5555500 \mathrm{E}-02$
-. $5555500 \mathrm{E}-02$
-. 55555 DOE-02
-. $5555500 \mathrm{E}-02$
-. $5555500 \mathrm{E}-02$

$$
\begin{array}{rr}
-0.3692300 E+06 & -0.2340000 \mathrm{E}+03 \\
-.3692300 \mathrm{E}+06 & -.2340000 \mathrm{E}+03 \\
.3692300 \mathrm{E}+06 & .2340000 \mathrm{E}+03 \\
.3692300 \mathrm{E}+06 & .2340000 \mathrm{E}+03 \\
.3692300 \mathrm{E}+06 & .2340000 \mathrm{E}+03 \\
.3692300 \mathrm{E}+06 & .2340000 \mathrm{E}+03
\end{array}
$$

LASGN - (016) ASSIGNMENT OF LOGIGAL VARIABLES ASA REFS. -7.1 .1 .2

RESULTS

ALL ANSWERS BELOW MUST BE TRUE
$T$
$T$
T
T
$T$
T
T
$T$

T
$T$
T
$T$
$T$
T
T
$T$
$T$
T
T
T
$T$

ALL ANSWERS BELOW MUST BE FALSE
F
F
F
F
F
F
F
F
F
F
F
F
F
F
INTRL - (017) ASSIGN INTEGER, REAL, AND DOUBLE PRECISION VALUES
ASA REFS. - 7.1.1.1. 5.1.1.2 RESULTS

## ASSIGN INTEGER VARIABLES

1 - TO REAL VARIABLES

$$
111.0 *
$$

$$
111.0
$$

$$
-1111.0
$$

$$
-1111.0
$$

$$
-1111100
$$

$$
-11111.0
$$

$$
1.0 *
$$

$$
1 \cdot 0
$$

2 - TO DOUBLE PRECISION VARIABLES

$$
\begin{array}{r}
-0.11111005 \\
-111110+05
\end{array}
$$

$$
0.1001 *
$$

$$
\cdot 10+01
$$

$$
0.111003 *
$$

$$
\cdot 1110+03
$$

$$
-0.1111004 *
$$

$$
-.11110+04
$$

ASSIGN INTEGER CONSTANTS
1 - TO REAL VARIABLES

$$
\begin{aligned}
& -2222.0 \\
& -2222.0
\end{aligned}
$$

$$
222.0 *
$$

$$
222.0
$$

$$
-22222.0 *
$$

$$
-22222.0
$$

2.0*
2.13

2 - TO DOUBLE PRECISION VARIABLES

$$
\begin{gathered}
0.2001 \\
\cdot 20+01 \\
-0.22222005 \\
-.222220+05 \\
-0.2222004 \\
-.22220+04 \\
0.222003 \\
.2220+03
\end{gathered}
$$

ASSIGN BASIC REAL CONSTANTS
1 - TO INTEGER VARIABLES

$$
\begin{array}{r}
3 \\
3 \\
3 \\
3 \\
-3 \\
-3
\end{array}
$$

2 - TO DOUBLE PRECISION VARIABLES

$$
\begin{array}{r}
0.33333001 \\
.333330+01 \\
\\
0.3333333001 \\
.33333330+01 \\
-0.3333333001 \\
-.3333330+01^{*} \\
-0.333333001 \\
-.3333330+01
\end{array}
$$

ASSIGN REAL VARIABLES
1 - TO INTEGER VARIABLES


2 - TO DOUBLE PRECISION VARIABLES
$-0.444002 *$
-. $444 \mathrm{D}+02$
$-0.4444002 *$
$-.44440+02$
$0.44444002 *$

- $444440+02$
$0.44444002 *$
- $444440+02$

ASSIGN double precision variables
1 - TO INTEGER VARIABLES
5555 *
5555
5 *
5
-5 *
$-5$
$-5$
2 - TO REAL VARIABLES
$-0.5555556 E 01 *$
-. $5555556 E+01$
$-0.5555556 E 01 *$ -. $5555556 \mathrm{E}+01$
$0.5555556 E 01$ * $.5555556 E+01$
0.555555E $04^{*}$

- $555555 \mathrm{E}+04$

ASSIGN DOUBLE PRECISION CONSTANTS

```
1 - TO INTEGER VARIABLES
            6*
            6
            -6*
            -6
            -6
            6666
            666
2 - TO REAL VARIABLES
    0.6666667E 14 *
        .6666667E+14
            0.66666E 01 *
            . 66666E +01
-0.6666666E 01*
    -. 6666666E + 01
-0.6666667E 01 *
    -.6666667E+01
```

ALL TEST OUTPUT SHOULD BE CHECKED AGAINST THE ASTERISKED (*) FIGURE HHICH PRECEDES IT

UGOTO - (020) UNCONDITIONAL GO TO statement

ASA REFS. -7.1 .2 .1 .1
RESULTS
1
2
3

4

5
6
7
8

THIS TEST IS SUCCESSFUL ONLY IF THE NUMBERS LISTED ABOVE ARE SEQUENTIALLY IN ORDER FROM 1 TO 8

AGOTO - (021) ASSIGN AND ASSIGNED Go To

ASA REFS. - 7.1.1.3 AND 7.1.2.1 RESULTS

1

2

3
4
5
6

7

8
9
10
11
12
13
14
15
16
17
18
19
20
THIS TEST IS SUCCESSFUL ONLY IF THE NUMBERS LISTED ABOVE ARE SEQUENTIALL'Y IN ORDER FROM 1 TO 20

```
CGOTO - (022) COMPUTED GO TO
ASA REF. - 7.1.2.1.3
```

RESULTS
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

THIS TEST IS SUCCESSFUL ONLY IF THE NUMBERS LISTED ABOVE ARE SEQUENTIALLiY IN ORDER FROM 1 TO 20

```
ARBAO - 1030J BASIC ADDITION
ASA REF. - 6.1
RESULTS
INTEGER ADOITION
TEST 1 0
TEST 2 0
TEST 3 0
TEST 4 0
TEST 5 0
TEST 6 0
REAL ADOITION
TEST }70.
TEST 8 0.0
TEST 9 0.0
TEST 10 0.0
TEST 11 0.0
TEST 12 0.0
ALL ABOVE ANSWERS SHOULD BE O FOR
THIS SEGMENT TO BE SUCCESSFUL
```

ARFAD $=10311$ D.P. ADDITION
ASA REF. -6.1
RESULTS
0.
0.
0.
0.
0.

THE 5 ANSWERS ABOVE SHOULD BE 0 PLUS OR MINUS AN ERROR FACTOR OF 0.1D-13

ARBSB $-(032)$ BASIC SUBTRACTION ASA REFS. - 6.1

RESULTS

TESTL INTEGER SUBTRACTION
0
0
0
0
0

TESTZ REAL SUBTRACTION

$$
\begin{array}{ll}
0.0 \\
0.0 \\
0 . & 0 \\
0 . & 0
\end{array}
$$

ALL ABOVE ANSWERS SHOULD BE O FOR THIS SEGMENT TO BE SUCCESSFUL

ARFSB (O33) D.P. SUBTRACTION
ASA REF. 6.1
RESULTS

$$
0,00000000000+00
$$

0.00000000000600
$0.00000000000+00$
$0,0000000000 D \$ 00$
$0,00000000000 \div 00$
$0,00000000000+00$
$0.00000000000+00$
$0.00000000000 \$ 00$
$0.00000000000 \$ 00$

THE ANSWERS ABOVE SHOULD BE 0 PLUS OR MINUS AN ERROR FACTOR OF O.1Dm 13
$\begin{aligned} &\text { ARBAS ( } 034) \text { BASIC ADDITION AND } \\ & \text { SUBTRACTION }\end{aligned}$
ASA REF. 6. 4
RESULTS

TESTI INTEGER ADO AND SUBT

| $\ldots$ | 0 |
| :--- | :--- | :--- |
|  | 0 |
|  | 0 |
|  | 0 |
|  |  |

TESTE REAL ADD AND GUBTR

> 0.0 0.0 0.0 0.0

ALL ABOVE ANSWERS SHOULO BE O FOR
THIS SEGMENT TO BE SUCCESSFUL
FORTRANTEST PR OG R A M S
PREPARED BY NATIONAL BUREAU OF STANDAROS
FOR USE ON LARGE FORTRAN PRDCESSORS
IN ACCORDANCE WITH ASA FORTRAN X3.9-1966
VERSION 3 PART 4
SAMPLE COMPUTER, FORTRAN COMPILER LEVEL
OPERATING SYSTEM VERSION
DATE, INSTALLATION NAME

ARFAS - (035) D.P. ADD AND SUBTR
ASA REF. - 6.1
RESULTS

```
0.
0.
0.
0.
    -.20679515310-24
```

The answers above should be o for THIS SEGMENT TO BE SUCCESSFUL. values hith exponents less than 10**(-14) ARE CONSIDERED ZERO

```
ARBMI - (036) INTEGER MULTIPLICATION
ASA REF. - b.1
RESULTS
```

0
0
0
0
0
0
0

```
ALL ABOVE ANSWERS SHOULD EE O FOR
this segment to be successful
```

ARBMR - (037) REAL MULTIPLICATION
ASA REF. - 6.1

RESULTS

$$
\begin{aligned}
& 0.0 \\
& 0.0 \\
& 0.0 \\
& 0.0 \\
& 0.0 \\
& 0.0 \\
& 0.0
\end{aligned}
$$

ALL AbOVE ANSWERS SHOULD BE D FOR THIS SEGMENT TO BE SUCCESSFUL

ARFMD - (038) D,P. MULTIPLICATION
ASA REF. 6.1

## RESULTS

$0.0000000000 D \$ 00$
$0.00000000000+00$
$0.00000000000+00$
$0.00000000000 \$ 00$
$0.00000000000+00$
$0.00000000000 \$ 00$
$0.0000000000 D+00$
$0.0000000000 D \$ 00$

THE ANSWERS ABOVE SHOULD EE O FOR
THIS SEGMENT TO BE SULCESSFUL



```
SBB67 - (050) SUBSCRIPTS FOR INTEGER AND REAL ARRAYS, \(V, K\)
```

ASA REF. 5.1.3
RESULTS

0
0.0

0
0
0.0
0.0

THE ANSHERS ABOVE SHOULD BE O FOR THIS SEGMENT TO BE SUCGESSFUL

SBB45 - (051) SUBSCRIPTS FOR INTEGER AND REAL ARRAYS, $V+K, V=K$

ASA REF. 5.1.3.3
RESULTS

$$
\begin{aligned}
& 0 \\
& 0 \\
& 0 \\
& 0.0 \\
& 0.0 \\
& 0.0
\end{aligned}
$$

THE ANSHERS ABOVE SHOULD BE O FOR THIS SEGMENT TO BE SUCCESSFUL
$\begin{aligned} \text { SB813 - (052) } & \text { SUBSCRIPTS INTEGER AND } \\ & \text { REAL, CキV, } C \neq V-K, C \neq V+K\end{aligned}$
ASA REF. 5.1.3.3
RESULTS

0
0
0.0
0.0

0
0
0.0
0.0

0
0.0

THE ANSHERS ABOVE SHOULD BE 0 FOR THIS SEGMENT TO BE SUCCESSFUL

SEFI7 - (053) SUBSCRIPTS FOR D.P. ARRAYS, ALL FORMS

ASA REF, 5,1,3,3
RESULTS

$$
\begin{aligned}
& 0.000000 \$ 00 \\
& 0.00000 D \$ 00 \\
& 0.00000 D \$ 00 \\
& 0.00000 D \$ 00
\end{aligned}
$$

THE ANSWERS ABOVE SHOULD BE O FOR THIS SEGMENT TO BE SUCCESSFUL

```
    FORTRRANTESTMPROGRAMS
    PKEPARED BY NATIONAL GUREAU OF STANDARDS
    FOR USE ON LARGE FORTRAN PROCESSORS
    IN ACCORDANCE WITH ASA FORTRAN X3.9-1966
    VERSION 3 PART 5
```

    SAMPLE COMPUTER, FORTRAN COMPILER LEVEL
    OPERATING SYSTEM VERSION
    DATE, installation name
    SIMIF - 1054 SIMFLE AFITH. IF
                                    AND LOGICAL IF
    ASA REF. \(=7.1 .2 .2\)
        7.1.2.3
    RESULTS
    T
    1
    1
    T
    T
    \(\uparrow\)
    \(T\)
    1
    1
    1
    the ten ansmers above must be true
    


```
FMAX - (062) INTRINSIC FUNCTIONS=-
                AMAXO,AMAXI,MAXO, MAXI,DMAX!
SA REF. - 8.2
ESULTS
EST OF AMAXO--
    -0
    .0
    .0
    .0
    .0
        END OF 2-ARGUMENT TEST:
        .0
        -0
    -0
        END OF 3-ARGUMENT TEST.
        .0
    -0
        END OF 4- OR S-ARGUMENT TEST.
EST OF AMAXI=0
        -0
        .0
        .0
        END OF 2-ARGUMENT TEST.
        -0
        .0
        -0
        END OF 3-ARGUMENT TEST.
        -0
        -0
        END OF 4- OR 5-ARGUMENT TEST.
```

```
TEST OF MAXO=-
    O
    O
    0
    0
        END OF 2-ARGUMENT TEST.
        O
        0
            END OF 3-ARGUMENT TEST.
        0
    O
        END OF 4- OR 5-ARGUMENT TEST.
TEST OF MAXI=-
        O
        0
        O
            END OF 2-ARGUMENT TEST.
        O
        0
            END OF 3-ARGUMENT TEST.
        0
            0
                END OF 4-OR 5-ARGUMENT TEST.
```

TEST OF DMAX1--
.0000000000
.0000000000
.0000000000
.0000000000
END OF z-ARGUMENT TEST.
.0000000000
.0000000000
.0000000000
END OF 3-ARGUMENT TEST.
.0000000000
.0000000000
END OF 4-OR 5-ARGUMENT TEST.
THE ABOVE ANSWERS SHOULD ALL BE O FOR
THIS TEST SEGMENT TO BE SUCCESSFUL.

$F O R T R A N T E S T P G O R A M S$ PREPARED BY NATIONAL BUREAU OF STANDARDS FOR USE ON LARGE FORTRAN PROCESSORS

IN ACCORDANCE WITH ASA FORTRAN $\times 3.9-1966$ VERSION 3 PART 6

SAMPLE COMPUTER, FORTRAN COMPTLER LEVEL OPERATING SYSTEM VERSION

DATE, INSTALLATION NAME

IFDIM - (065) INTRINSIC FUNCTIONS - OIM AND IOIM (POSITIVE OIFFERENCE)
ASA REF. - 8.2

## RESULTS

$$
\begin{aligned}
& 0.00 \\
& 0.00 \\
& 0.00 \\
& 0.00
\end{aligned}
$$

0
0
0
0
0
ALL ABOVE ANSWERS SHOULO BE O FOR THIS TEST SEGMENT TO BE SUCCESSFUL.

IFSGL - 10661 INTRINSIC FUNCTION SNGL $=-$ OBTAIN MOST SIGNIFICANT PT OF D.P. ARGUMENT.

ASA REFS. - 8.2
RESULTS
LINE A . $48748749377973+003$
LINE B . $48748749160767+003$
LINE A - $39689549238764+003$
LINE B $\quad .39689540100098+003$
LINE A . $33333962558434+0$ OD
LINE B $\quad 33333962410488+000$
LINE A . $79379080477578+003$
LIME B . 7937908020 C195+003
LINEA $\quad .44445133956719+0$ OO
LINE B $\quad .44445133954287+000$
LINE A $\quad-.66667193798867+000$
LINE B $\quad=.66667183488607+000$
LINE A $\quad=.39689539609539+093$
LINE B $\quad-39689539337158+003$
LINE A . $48748749377973+003$
LINE 8 . $88748749160767+003$
LINE B SHOULD AGREE WITH LINE A ONLY TO THE PRECISION OF A REAL DATUM. REMAINING OIGITS RESULT FROM OUTPUT CONVERSION WHEN A REAL VALUE IS ASSIGNED TO D.P. FOR PRINTING.

IFREL - (067) INTRINSIC FUNCTION-REAL
ASA REF. - 8.2

RESULTS

$$
\begin{aligned}
& 0.00000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& 0.0000 \\
& \\
& 0 .
\end{aligned}
$$

ALL ABOVE ANSWERS SHOULD BE 0 FOR THIS TEST SEGMENT TO BE SUCCESSFUL.

```
IFIMG - (068) INTRINSIC FUNCTION - AIMAG
                                    OBTAIN IMAGINARY PT
                                    OF COMPLEX ARGUMENT
ASA PEF.- 8.2
```

RESULTS

```
0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.30000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
    0.00000
```

ALL ABOVE ANSWERS SHOUL? BE O FOR THIS TEST SEGMENT TO BE SUCCESSFUL.


## RESULTS

| LINE A | $0.9765625 E=03$ |
| :--- | :--- |
| LINE B | $0.976562500000000=03$ |
| LINE A | $0.1953825 E=02$ |
| LINE B | 0.19538250000000002 |
| LINE A | $0.5859375 E 002$ |
| LINE B | $0.585937500000000=02$ |
| LINE A | $0.1048576 E+07$ |
| LINE B | $0.104857600000000 \$ 07$ |
| LINE A | $0.1146880 E \$ 06$ |
| LINE B | $0.114688000000000+06$ |

## A COMPARISON OF LINE A AGAINST LINE B IS NEEDED TO CHECK THE VALIDITY OF TEST

IFCPX - (070) INTRINSIC FUNCTION - CMPLX EXPRESS TWO REAL ARGUMENTS IN COMPLEX FORM
ASA REF. - 8.2
RESULTS

| 0.0000000 | 0.0000000 |
| :--- | :--- |
| 0.0000000 | 0.0000003 |
| 0.000000 | 0.0000000 |
| 0.000000 | 0.0000000 |
| 0.0000000 | 0.0000000 |
| 0.0000000 | 0.0000000 |

THE ABOVE ANSWERS SHOULD ALL BE $O$ FOR THIS TEST SEGMENT TO BE SUCCESSFUL.

IFCJG - (071) INTRINSIC FUNCTION - CONJG OBTAIN CONJUGATE OF a COMPLEX NUMBER
ASA REFS. - 8.2
RESULTS

$$
\begin{aligned}
& 0.00000000 .0000000 \\
& 0.00000000 .0000000 \\
& 0.0000000 \quad 0.0000000 \\
& 0.00000000 .0000000 \\
& 0.0000000 \quad 0.00000100 \\
& 0.00000000 .0000000 \\
& 0.00000000 .0000000 \\
& 0.00000000 .0000000
\end{aligned}
$$

ALL ABOVE ANSWERS MUST BE $O$ FOR THIS TEST SEGMENT TO BE SUCCESSFUL.
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$
$\qquad$

```
IFBMS - (072) BASIC FORTRAN INTRINSIC
    FUNCTIONS ACCEPT EXPRESSIONS
    OF TYPE SPECIFIED IN I.F.TABLE
```

ASA REF.- 8.2
RESULTS
TEST OF ABS IN EXPRESSIONS -
0.0
0.0
0.0
0.0
TEST OF IABS IN EXPRESSIONS -
0
0
0
0
TEST OF FLOAT IN EXPRESSIONS -
0.0
0.0
0.0
0.0
TEST OF IFIX IN EXPRESSIONS -
0
0
0
0
TEST OF SIGN IN EXPRESSIONS -
0.0
0.0
0.0
0.0
TEST OF ISIGN IN EXPRESSIONS -
0
0
0
0

COMBINATION OF ALL INTRINSIC FUNCTIONS 0.0 0.0
0
0
0.0
0.0
0
0

ALL ABOVE ANSWERS SHOULD BE 0 FOR THIS TEST SEGMENT TO BE SUCCESSFUL．

```
IFFMS - (073) FORTRAN INTRINSIC FUNCTION
    ACCEPT EXPRESSIONS OF TYPE
    SPECIFIED IN I.F.TABLE
ASA REF.- 8.2/TABLE 3
```

RESULTS
TEST OF DABS IN EXPRESSIONS
3.
0 。
0 。
0 .
TEST OF AINT IN EXPRESSIONS
0 。
0 .
0 .
0 。
TEST OF INT IN EXPRESSIONS
0
0
0
0
TEST OF IOINT IN EXPRESSIONS
0
0
0
0
TEST OF AMOD, MOD IN EXPRESSIONS
0.
0 .
0
TEST OF AMAXO, AMAX1, MAXO,MAXI AND DMAX
0.
0 .
0
0.

ST OF AMINO,AMIN1,MINO,MIN1 AND DMIN 0.

0
0
0 .
ST OF DSIGN AND DBLE IN EXPRESSIONS 0.
0.
0.

ST OF DIM AND IDIM IN EXPRESSIONS 0 .
0.

0
0
ST OF SNGL,REAL,AIMAG,CMPLX AND NJG IN EXPRESSIONS

> 0.
> 0.
> 0.

ST OF SOME COMBINATIONS OF ABOVE TRINSIC FUNCTIONS
0.

0 .
$L$ ABOVE ANSWERS SHOULD BE $O$ FOR THIS GMENT TO BE SUCCESSFUL.

FORTRANTESTPROGRAMS PREPAREO BY NATIONAL BUREAU OF STANDARDS FOR USE ON LARGE FORTRAN PROCESSORS IN ACCORDANCE WITH ASA FORTRAN X3.9-1966 VERSION 3 PART 7

SAMPLE COMPUTER, FORTRAN COMPILER LEVEL
OPERATING SYSTEM VERSION
DATE, INSTALLATION NAME

EXPON - (080)
BASIC EXTERNAL FUNCTION -EXP-
(EXPONENTIAL -TYPE REAL)
ASA REF. - 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS
$x=-16.0 \quad 0.1125351747192591145 E-06$ .1125352E-06
$x=-8.0 \quad 0.3354625279025118388 E-03$ . 3354626 E-0 3
$x=-4.0 \quad 0.1831563889873418029 E-01$
. 1831564 E-01
$X=0.0 \quad 0.100000000000000000 E+01$ $.1000000 E+01$
$x=4.0 \quad 0.5459815003314423908 E+02$ . $5459815 E+02$
$x=8.0 \quad 0.2980957987041728275 E+04$ . $2980958 \mathrm{E}+04$
$x=16.0 \quad 0.8886110520507872637 E+07$ . $8886111 E+07$

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTED TO 7 DIGITS

DEXPO - (081)
BASIC EXTERNAL FUNCTION -DEXP-
(EXPONENTIAL -TYPE DOUBLE PRECISION)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS

$$
\begin{aligned}
& x=-16.0 \\
& \text { 0.11253517471925911450-06 } \\
& \text {. 112535174719260-06 } \\
& x=-8.0 \quad 0.33546262790251193880-03 \\
& \text {. } 335462627902510-03 \\
& x=-4.0 \quad 0.18315638888734180290-01 \\
& .183156388887340-01 \\
& x=0.0 \quad 0.10000000000000000000+01 \\
& .10000000000000+01 \\
& x=4.0 \quad 0.54599150033144239080+02 \\
& .545981500331440+02 \\
& x=8.0 \quad 0.29809579870417282750+04 \\
& .298095798704170+04 \\
& x=16.0 \\
& 0.88861105205078726370+07 \\
& .888611052050790+07
\end{aligned}
$$

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTED TO 14 DIGITS

CEXPO - (OB2)
BASIC EXTERNAL FUNCTION -CEXP-
(EXPONENTIAL TYPE COMPLEX)
ASA REF, $=8,3,3$ (TABLE 4)
(COMPLEE ARGUMENT)

## EXPECTED RESULT

FUNETION RESULT
$(m 0.1611810 E+02,-0,7330383 E+01)$ $0.5000000 E=07 \quad-0.8660254 E=07$ $0.5000000 \mathrm{E}=07 \quad 0.8660254 \mathrm{E}-07$
( $\quad$ ( $0.1450866 \mathrm{E}+02,-0,7330383 \mathrm{E}+01$ ) $0.2500000 \mathrm{E}=06 \quad-0.4330127 E=06$ $0.2500000 E=06-0.4330127 E=06$
(न0.138155IE+0ट, $0.6283985 E 01)$ $0.1000000 E=05 \quad 0.0000000 E+00$ $0.1000000 E=05 \quad 0,0000000 E+00$
$(-0.1220607 E+02,0,6283185 E+01)$ $0.5000000 E=05 \quad 0.0000000 E+00$ $0,5000000 E=05 \quad 0,0000000 E+00$
$(-0,1151293 E+02,-0,5235988 E+01)$ $0,5000000 E=05 \quad 0,8660254 E=05$ $0.5000000 \mathrm{E}=05 \quad 0.8660254 \mathrm{E}=05$
( $00.9903488 \mathrm{E}+01,-00,5235988 \mathrm{E}+01$ ) $0.2500000 E=04 \quad 0.4330127 E-04$ $0,2500000 \mathrm{E}=04 \quad 0.4330127 \mathrm{E}=04$
$(\sim 0.9210340 E+01,00,4188790 E+01)$

$$
-0.5000000 E=04 \quad 0.8660254 E=04
$$

- $0.5000000 \mathrm{E}=04 \mathrm{O}$-8660254EनU
$(=0.7600902 E+01,-0.4188790 E+01)$
$-0.2500000 E=03 \quad 0.4330127 E=03$
$-0.2500000 E=03 \quad 0.4330127 E=03$
( $=0.6907755 E+01,00.3145593 E W 01)$
- $0.1000000 E=02 \quad 0.0000000 E+00$
$=0.1000000 E=02 \quad 0.0000000 E+00$
$(-0.5298317 E+01,-0.3141593 E+01)$
, $0.5000000 E=02 \quad 0.0000000 E+00$
-0.5000000E-02 O.0000000E 000


```
LOGTM - (083)
BASIC EXTERNAL FUNCTION -ALOG-
(NATURAL LOG -TYPE REAL)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS
x=0.125 -2.0794415416798359
    -2.479442
x=0.25 -1.3862943611198906
    -1.38623.4
x=0.5 -0.6931471805599453
x=1.0 0.0000000000000000
    0.0000000
X=1.5 0.4054651081081644
                                .40540551
x=2.0
    0.6931471805599453
                        .5931472
LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTED TO 7 OIGITS
```

```
OPLOS - (084)
BASIO EXTERNAL FUNCTION -DLOG-
(NATURAL LOG -TYPE DOUBLE PRECISION)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS
x=0.125 -2.07944154167983590+00
    -2.07944154167980+00
x=0.25 -1.38623436111989360+00
    -1.38623436111990+00
x=0.5 -0.69314718055994530+00
    -.693147180559950+00
X=1.0 0.0003300000000000
    0.
X=1.5 0.40545510810816440+00
                                .405455108108160+30
X=2.0 0.69314718055994530+00
    .693147180559950+30
```

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTEO TO 14 DIGITS





```
DPSIN - (089)
BASIC EXTERNAL FUNCTION -DSIN-
(TRIGONOMETRIC SINE -TYPE D.P.)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS
x=0.0 0.00000000000000000000000
x=1.0 +0.841470984807896506652500+00
    .84147098480790D+00
x=2.0 +0.909297426825681695396020+00
                        .909297426825680 +00
x=3.0+0.141120008059867222100740+00
                        .141120008059870+00
X=(PI) 0.00000000000000000000000
                        .543383291778630-21
x=4.0 -0.756802495307928251372640+00
    -.756802435307930+00
X= 5.0 - 0.9589242745631384688993150+00
    -.958924274663140+00
x=6.0-0.279415498198925872811560+00
    -.279415498198930+00
X=(2PI) 0.00000000000000000000000
    -.128676658355730-20
```

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTEO TO 14 OIGITS

CSICO - (090)

BASIC EXTERNAL FUNCTIONS -CSIN, CCOS(TRIG. SINE AND COSINE -TYPE COMPLEX)

```
ASA REF 8.3.3 (TABLE 4)
```

FUNCTION
RESULTS

| TABLE VALUE | 1.2984576 | 0.6349639 |
| :--- | ---: | ---: |
| CSIN(1.,1.) $=$ | 1.2984576 | .6349639 |
| TABLE VALUE | 0.8337300 | -0.9888977 |
| CCOS $(1 ., 1)=$. | .8337390 | -.9888977 |

$\operatorname{CSIN}(x) * * 2+\operatorname{CCOS}(x) * * 2=1.0,0.0$

ARGUMENT

| $(1,1 / 1)$ | 1.0000000 | -.0000000 |
| :--- | :--- | :--- |
| $(2,1 / 2)$ | 1.0000000 | 0.0000000 |
| $(3,1 / 3)$ | 1.0000000 | .0000000 |
| $(4,1 / 4)$ | 1.0000000 | 0.0000000 |
| $(5,1 / 5)$ | 1.0000000 | -.0000000 |
| $(6,1 / 6)$ | 1.0000000 | -.0000000 |
| $(7,1 / 7)$ | 1.0000003 | .0000000 |
| $(8,1 / 8)$ | 1.0000000 | -.0000000 |
| $(10,1 / 10)$ | 1.0000000 | -.0000000 |


| COSNS - (091) | OPCOS - (092) |
| :---: | :---: |
| BASIC EXTERNAL FUNCTION ECOSO | BASIC EXTERNAL FUNCTION DDCOS\% |
| TYRIGONOMETRIC COSINE TYPE REAL) | '(TRIGONOMETRIC COSINE TYPE D,P.) |
| TSA REF. 8,3.3 (TABLE 4) | ASA REF. 6.3.3 (TABLE 4) |
| TINE I OF EACH PAIR IS HOLLERITH INFORMATION | LINE I OF EACH PAIR IS HOLLERITH INFORMATION |
| RE8ULIS | RESULTS |
| $x=0.0 \quad \$ 1.000000000000$ | $x=0.0 \quad+0.100000000000000000000000+01$ |
| 1.0000000 | $0.100000000000000 \$ 01$ |
| XE $1.0 \quad 0.540302305668$  <br>  0.5403023 | $\begin{gathered} x=1.0 \quad 0.540302305868139717400940+00 \\ 0.540302305868140+00 \end{gathered}$ |
|  | $\begin{array}{cc} x=2.0 \quad 0.416146836547142386997570+00 \\ 0.456146836547140+00 \end{array}$ |
| $\begin{array}{ll} x=3.0 & 0.989992496600 \\ & 0.9899925 \\ \hline \end{array}$ | $\begin{array}{rl} X=3.0 & 0.98999249660044545727157 D+00 \\ & =0.989992496600450+00 \end{array}$ |
| $x=(P I) \quad-1,000000000000$ <br> 1.000000 | $\frac{x=(P I)-0.100000000000000000000000+01}{0.100000000000000+01}$ |
| $\begin{array}{ll} x=4.0 & 0,653643620864 \\ & 0,6536436 \\ \hline \end{array}$ | $\begin{aligned} x=4.0 \quad & 0,653643620863611914639170+00 \\ & 00,653643620863610 \$ 00 \end{aligned}$ |
| $\frac{X=5.0}{0.283662185463}$ | $\frac{x=5.0 \quad+0,283662185463226264466640+00}{0,283662185463230600}$ |
| $X=6,0 \quad$ 0.960170286650 <br>  0.9601703 | $\begin{gathered} x=6.0 \quad 0.960170286650366020545650+00 \\ 0.960170286650370 中 00 \end{gathered}$ |
| $\frac{X=(2 P I)}{1,0000000}$ | $\frac{X=(2 P I) \quad 0.100000000000000000000000+0 \&}{0.10000000000000+01}$ |
| LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTED TO 7 DIGITS | LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTED TO IA DIGITS |

FORTRANTESTPROGRAMS PREPARED BY NATIONAL BUREAU OF 5 TANDARDS FOR USE ON LARGE FORTRAN PROCESSORS IN ACCORDANCE WITH ASA FORTRAN $\times 3.9-1966$ VERSION 3 PART 8

SAMPLE COMPUTER, FORTRAN COMPILER LEVEL
OPERATING SYSTEM VERSION
DATE, INSTALLATION NAME

TANGH - (094)
BASIC EXTERNAL FUNCTION TANHO
(HYPERBOLIC TANGENT TYPE REAL)
ASA REF $8,3,3$ (TABLE 4)
LINE I OF EACH PAIR IS
HOLGERITH INFORMATION
RESULTS
$\frac{x \equiv 0.0 \quad 0,0000000000}{0,0000000}$
$x=2,0 \quad 0.9640275801$
0.9640276
$x=2,5 \quad 0,9866142982$
0.9866143
$X=4,0 \quad 0.9993292997$
0,9993293
$x=0,0 \quad 0.9999877117$
0,9999877
$x=8.0 \quad 0.9999997749$
0.9999998

LINE 2 OF EACH PAIR IS TRE FUNCTION CALCULATION PRINTED TO 7 DIGITS

```
SQROT - (095)
```

BASIC EXTERNAL FUNGTION - SQRT-
(SQUARE ROOT -TYPE REAL)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS

```
x=2.0 1.41421356237310
    1.4142136
X=3.0 1.73205090756888
        1.7320508
x=17.0 4.12310562561756
    4.1231056.
x=31.0 5.56776436283002
    5.5677644
X=89.0 9.43398113205660
    9.4339811
```

LINE 2 OF EACH PAIR IS THE FUNCTION
CALCULATION PRINTEO TO 7 OIGITS

DSQRO - (096)
BASIC EXTERNAL FUNCTION -DSQRT-
(SQUARE ROOT -TYPE D.P.)
ASA REF.- 8.3 .3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION

RESULTS
$x=2.0 \quad 1.41421356237309504880+00$ $1.4142135623731+000$
$x=3.0 \quad 1.73205080756897729350+00$ 1.7320508075689+000
$x=17.0 \quad 4.12310562561766054980+00$ 4.1231056256177+000
$x=31.0 \quad 5.5677643628300219221 D+00$ $5.5677643628300+000$
$x=89.0 \quad 9.4339811320566038113 D+00$ $9.4339811320566+000$

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTEO TO 14 DIGITS

```
CSQRO - (097)
```

BASIC EXTERNAL FUNCTIION -CSQRT-
(SQUARE ROOT -TYPE COMPLEX)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
THE EXPECTED VALUE

RESULT

```
.9950342E-02
.9950042E-02 .998i3340E-03
.9800666E-01 . 1986693E-01
.9800666E-01 . 1986693E-01
.9553365E+00 . 2955202E+00
.9553365E+00 . 2955202E+00
.9210610E+01 . 3884183E+01
.9210610E+01 . 3894183E+01
.8775826E+02 . 4794255E+02
.8775826E+02 . 4794255E+02
.8253356E-02 . 5646425E-02
.8253356E-02 . 564,6425E-02
.7648422E-01 . 644:2177E-01
.7648422E-01 . 6442177E-01
.6967067E+00 . 7173561E+00
.6967067E+00 . 7173561E+00
.5403023E+01 . 8414710E+01
.5403023E+01 . 841:4710E+01
.4161468E+02 =.9092974E+02
.4161468E+02 =.9092974E+02
```

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION


ACTG2 - (100)
BASICI EXTERNAL FUNCTION - ATAN2-
(ARCTANGENT, 2 ARGUWENT -TYPE REAL)
ASA REF.- 8.3.3 (TABLE 4)
LINE 1 OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS

| $x=0.125$ | $\begin{aligned} & 0.124354994547 \\ & .1243550 \end{aligned}$ |
| :---: | :---: |
| $x=0.250$ | $\begin{gathered} 0.244978663127 \\ .2448787 \end{gathered}$ |
| $x=0.375$ | $\begin{aligned} & 0.3587170670271 \\ & .3587707 \end{aligned}$ |
| $x=0.500$ | $\begin{aligned} & 0.4636 .47609001 \\ & .4635476 \end{aligned}$ |
| $x=-0.750$ | $\begin{aligned} & -0.643501108793 \\ & -.6435011 \end{aligned}$ |
| $x=1.000$ | $\begin{aligned} & 0.785398163397 \\ & .7853982 \end{aligned}$ |

LINE 2 OF EACH PAIR IS THE FUNCTION CALCULATION PRINTED TO 7 DIGITS

```
DATN2 - (101)
BASIC EXTERNAL FUNCTION -DATAN2=
(ARCTANGENT, 2 ARGUMENT -TYPE D.P.)
ASA REF.- 8.3.3 (TABLE 4)
LINE I OF EACH PAIR IS
HOLLERITH INFORMATION
RESULTS
x=0.125 0.1243549945470+00
                                .124354994547+000
x=0.250 0.2449786631270+00
    .244978663127+000
x=0.375 0.3587706702710+00
    .358770670271+000
x=0.500 0.4636476090010+00
    .463647609001+000
x=-0.750 -0.6435011087930+00
                                    -.643501108793+000
x=1.000 0.7853981633970+00
                                .735398163397+000
LINE 2 OF EACH PAIR IS THE FUNCTION
CALCULATION PRINTED TO 12 DIGITS
```



CABSA - (103)

BASIC EXTERNAL FUNCTION -CABS-
(MODULUS OF A COMPLEX NUMBER)
ASA REF.- 8.3 .3 (TABLE 4)
RESULTS

| SET 1 | SET 2 |
| :---: | :---: |
| . $100000 \mathrm{E}-96$ | . $500000 \mathrm{E}-06$ |
| . $100000 \mathrm{E}-05$ | . $500000 \mathrm{E}-05$ |
| . $100000 \mathrm{E}-04$ | . $500000 \mathrm{E}-04$ |
| . $100000 \mathrm{E}-03$ | . $500000 \mathrm{E}-03$ |
| -100000E-02 | . 50000 OE-02 |
| . $100000 \mathrm{E}-01$ | . $500000 \mathrm{E}-01$ |
| . $100050 \mathrm{E}+00$ | $.500000 \mathrm{E}+00$ |
| . $100000 \mathrm{E}+01$ | .500000 C 01 |
| . $100000 \mathrm{E}+02$ | . $500000 \mathrm{E}+02$ |
| -100000E+03 | . $500000 \mathrm{E}+03$ |
| . $100000 \mathrm{E}+04$ | . $500000 \mathrm{E}+04$ |
| . $100000 \mathrm{E}+05$ | . $500000 \mathrm{E}+05$ |
| . $100000 \mathrm{E}+06$ | . $500000 \mathrm{E}+05$ |
| . $100000 \mathrm{C}+07$ | . $503000 \mathrm{E}+07$ |
| . $100000 \mathrm{E}+08$ | - $500000 \mathrm{C}+08$ |

VALUES IN EACH SET SHOULD be positive . 1 FOR SET 11.5 FOR SET 21, EXPONENT RANGE FROM -06 TO +08 IN SEQUENGE

| BSFTS (110) STATEMENT FUNCTION TEST INTEGER AND REAL | FSFTS - (ILD) STAYEMENT FUNCIION TEST |
| :---: | :---: |
| ASA REF = B.1.2 | DOUELE PRECISION, COMPLEX AND LOGICAL |
| RESULTS | ASA REF. 8, 1,2 |
| 0.0000000000 | RESULTS |
| $\begin{array}{r} 0.0000000000 \\ 0 \end{array}$ | $\begin{aligned} & 0.0000000000000000000+00 \\ & 0.0000000000000000000+00 \end{aligned}$ |
| 0 | $\begin{aligned} & 0,0000000000000000000+00 \\ & 0.000000000000000000 D+00 \end{aligned}$ |
| 0.0000000000 |  |
| $\begin{array}{r} 0,0000000000 \\ 0 \end{array}$ | $\begin{aligned} & 0.0000000000000000000+00 \\ & 0.0000000000000000000+00 \end{aligned}$ |
| 0 | $\begin{aligned} & 0.0000000000000000000+00 \\ & 0.000000000000000000+00 \end{aligned}$ |
| 0,0000000000 |  |
| 0.0000000000 | 0.00000000 .0000000 |
| - 0 | 0.00000000 .000000 |
| 0 | 0.00000000 .0000000 |
|  | 0.0000000 .0000000 |
| 0.0000000000 |  |
| 0,0000000000 | ALL ABOVE ANSWERS SHOULD BE O FOR THIS |
|  | TEST SEGMENT TO BE SUCCESSFUL. VALUES |
| 0 | WITH EXPONENTS LESS THAN IO**(नIA) ARE CONSIDERED ZERD |
| ALL ABOVE ANSWERS SHOULD BE O FOR |  |
| THIS TEST SEGRENT TO BE SUCCESSFUL. | T-T T T |
|  | THE FDUR ABOVE ANSWERS SHOULD BE TRUE FOR THIS SEGMENT TO BE SUCCESSFUL |

 SUBTRACTION

ASA REF. - 6.1
RESULTS

$$
\begin{array}{ll}
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000 \\
.0000 & .0000
\end{array}
$$

TEST IS POSIPIVE IF NUMBERS PRINTED ABOVE ARE 0.0 .0 .0

CPXMU - (141) COMPLEX MULTIPLICATION
TSA REF. 6.1
RESULTS

| 1.000 | 0.000 |
| :---: | :---: |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1,000 | 0,000 |
| 1.000 | 0,000 |
| 1.000 | -0,000 |
| 1.000 | 0.000 |
| 1.000 | 0,000 |
| 1.000 | -0,000 |
| 1.000 | -0,000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0.000 |
| 1.000 | 0,000 |
| 1,000 | 0,000 |

TEST IS POSITIVE LF NUMBERS PRINTED ABOVE ARE $100,0,0$

ERROR SHOULD NOT EXCEED + OR . .00T

```
CPXDV - (142) DIVISION OF
                                    COMPLEX NUMBERS
```

ASA REF.-6.1

RESULTS

| 1.0000 | 1.0000 |
| :--- | :--- |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |
| 1.0000 | 1.0000 |

TEST IS POSITIVE IF NUMBERS PRINTED ABOVE ARE 1.0 .1 .0

ERROR SHOULD NOT EXCEED + OR - . OOOI

```
CPXEX - (143) COMPLEX EXPONENTIATION
ASA.REF.6.1
RESULTS BASED ON THE FUNCTION
1.0=S1N**2(x)+\operatorname{cos**2(x)}
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 -.0000
    1.0000 =.0000
    1.0000 -.0000
    1.0000 -.0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 .0000
    1.0000 -.0000
    1.0000 -.0000
    1.0000 -.0000
    1.0000 =.0000
TEST IS POSITIVE IF NUMBERS PRINTFD
ABOVE ARE CLOSE TO 1.0,0.0
ERROR SHOULD NOT EXCEED + OR - .OOOI
```


$\begin{aligned} \text { CREOP - (148) } & \text { OPERATIONS ON REAL AND } \\ & \text { COMPLEX NUMBERS }\end{aligned}$

```
ASA REF. 6.1
```

RESULTS
$2.0000-1.0000$
TEST IS POSITIVE IF NUMBERS PRINTED ABOVE ARE 2.0,-1.0
1.0000 .0000

TEST IS POSITIVE IF NUMBERS PRINTED ABOVE ARE 1.0,0.0

ERROR SHOULO NOT EXCEED + OR - . 0001

MISC3-(149) EFFECT OF BLANKS WITHIN STMNT AND CONTINUAYION OF STMNT TO 20 LINES

ASA REFS. 3.1 .4 .13 .2 .4 .3 .33 .2 .4
RESULYS

0

- 0

TEST IS POSITIVE IF NUMBERS PRINTED ABOVE ARE D

```
MISC4 - (150) EFFECT OF BLANKS WITHIN
            STMNT AND CONTINUATION
            OF STMNT TO 20 LINES
```

ASA REFS. - 3.1.4.1 3.2.4.3.3 3.2.4
RESULTS
$0.0000 \quad 0.0000$
.0000 .0000
TEST IS POSITIVE IF NUMBERS PRINTED ABOVE ARE $0.0,0.0$

BRFCP - (160) REAL EXTERNAL FUNCTIONS
ASA REF. $=8.3 .1$
RESULTS SHOULD BE POSITIVE

TEST 1 IS POSITIVE
test 2 Is positive
test 3 is positive

TEST 4 IS POSITIVE

TEST 5 is POSitive

TEST 6 IS POSITIVE

BIFCP - (16I) INTEGER EXTERNAL FUNCTIONS WITH INTEGER AND REAL ARGS

## ASA REF. $=8.3 .1$

RESULTS SHOULD BE POSITIVE

TEST 1 IS POSITIVE

TEST 2 Is POSITIVE

TEST 3 IS POSITIVE

TEST 4 IS POSItive
test 5 is positive

TEST 6 IS POSITIVE

```
FIFCP - (IG3) INTEGER FUNCTION IN
                                    FULL FORTRAN
ASA REF. 8.3.1
RESULTS SHOULD BE POSITIVE
TEST & IS POSITIVE
TEST 2 IS POSITIVE
TEST 3 IS POSITIVE
TEST 4 IS POSITIVE
TEST 5 lS POSITIVE
TEST 6 IS POSITIVE
TEST 7 IS POSITIVE
```

$\begin{aligned} & \text { FRFCP - } 11621 \text { REAL FUNCTIONS WITH } \\ & \text { LOGICAL. D.P. AND COMPLEX ARGS }\end{aligned}$ ASA REF. 8.3.1

RESULTS SHOULD BE POSITIVE

TEST I IS POSITIVE.

TEST 2 IS POSITIVE.

TEST 3 IS POSITIVE.

TEST 4 IS POSITiVE.

TEST 5 IS POSITIVE.

TEST 6 IS POSITIVE.

TEST 7 IS POSITIVE.

CFCCP - (164) COMPLEX FUNCTIONS

ASA REFS. 8.3.1.8.3.2
RESULTS

| . 0 | - 0 |  | TEST | 1 | POSITIVE |  | 0.0,0.0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| . 0 | - 0 | - - | TEST | 2 | POSITIVE | IF | $0.0,0.0$ |
| . 0 | . 0 | - - | TEST | 3 | POSITIVE | IF | $0.0,0.0$ |
| . 0 | - 0 | - | TEST | 4 | POSITIVE | IF | $0.0,0.0$ |
| - 0 | - 0 | =- | TEST | 5 | positive | IF | 0,0,0.0 |
| - 0 | - 0 | - | TEST | 6 | POSITIVE | $1 F$ | 0.0.0.0 |
| - 0 | - 0 |  | TEST | 7 | POSITIVE | 15 | 0.0 .0 .0 |
| . 0 | - 0 | - | TEST | 8 | POSITIVE | 15 | $0.0,0.0$ |
|  |  |  | TEST | 9 | IS POSITI |  |  |
|  |  |  | TEST | 10 | IS POSITI |  |  |

FORTRANTESTPROGRAMS PREPARED BY NATIONAL BUREAU OF STANDARDS FOR USE ON LARGE FORTRAN PROCESSORS IN ACCORDANCE WITH ASA FORTRAN $\times 3.9-1965$ VERSION 3 PART 11

SAMPLE COMPUTER, FORTRAN COMPILER LEVEL OPERATING SYSTEM VERSION DATE, INSTALLATION NAME

## DPFCP - (165) DOUBLE PRECISION FUNCTIONS

ASA REFS. 8.3.1,8.3.2

## RESULTS

TEST 1 IS POSITIVE
TEST 2 IS POSITIVE
TEST 3 IS POSITIVE
TEST 4 IS POSITIVE
TEST 5 IS POSITIVE
TEST 6 IS POSITIVE
TEST 7 IS POSITIVE
TEST 8 IS POSITIVE
TEST 9 IS POSITIVE
TEST 10 IS POSITIVE
$0.0 \quad 0.0$
TEST 11 IS POSITIVE IF NUMBERS PRINTED ABOVE ARE $0.0,0.0$

TEST 12 IS POSITIVE

```
    BFCCP - (166) LOGICAL FUNCTIONS
    ASA REF 8.3.1
    RESULTS
    TEST 1 IS POSITIVE
    TEST 2 IS POSITIVE
    TEST 3 IS POSITIVE
    TEST 4 IS POSITIVE
    TEST 5 IS POSITIVE
    TEST 6 IS POSITIVE
    TEST }7\mathrm{ IS POSITIVE
    TEST 8 IS POSITIVE
    TEST }9\mathrm{ IS POSITIVE
    TEST 10 IS POSITIVE
    0.0000 0.0000
    TEST 11 IS POSITIVE IF NUMBERS PRINTED
    ABOVE ARE 0.0,0.0
    END OF (166)
```

    SERTN - (16T) SUBROUT INE SUBPROGRAM
    ASA REF : - 8.4.1
    RESULTS
    

II-C-57

FSBRT - (158) SUBROUTINE SUBPROGRAMS

## ASA REF. - 8.4.1

## RESULTS

TEST IS SUCCESSFUL IF EACH group contains same values

1
1
1
1
1
1
1
1
2.0
2.0
2.0
2.0
2.0
2.0
2.0
2.0
2.0
2.0
$4.00+00$
4.00403
$4.00+00$
$4.00+00$
$4.00+00$
$4.00+00$
$4.00+00$
4. $00+00$

| 6.0 | 6.0 |
| :--- | :--- |
| 6.0 | 6.0 |
| 6.0 | 6.0 |
| 6.0 | 6.0 |
| 6.0 | 6.0 |
| 6.0 | 6.0 |
| 6.0 | 6.0 |
| 6.0 | 6.0 |

$T$
$T$
$T$
$T$
$T$
$T$
$T$
$T$

BLKDT - (169) BLOCK DATA SUBPROGRAM

```
ASA REF. 8.5
```

RESULTS
TEST IS SUCCESSFUL IF EACH
GROUP CONTAINS SAME VALUES


AB
$A B$
$A B$
FORTRANTEST PROGRAMS
PREPARED BY NATIONAL BUREAU OF SIANDAROS
FOR USE ON LARGE FORTRAN PROCESSORS
IN ACCORDANCE WITH ASA FORTRAN X $3.9-1966$
VERSION 3 PART 12

SAMPLE COMPUTER, FORTRAN COMPILER LEVEL
OPERATING SYSTEM VERSION
date, installayion name

BLKKDA - (179) SEVERAL BLOCK DATA SUBPROGRAMS
ASA REF. 8.5
RESULTS

TEST IS SUCCESSFUL IF EACH GROUP CONTAINS SAME VALUES

2.0
2.0
2.0
2.0
$4.00+00$
4.00 000
$4,00+00$
$4.0 D+00$

| 3.0 | 4.0 |
| :--- | :--- |
| 3.0 | 4.0 |
| 3.0 | 4.0 |
| 3.0 | 4.0 |

$F$
$F$

UNFRW - |1801 UNFORMATTED READ AND WRITE STATEMENTS

ASA REFS - 7.1.3.2.4 AND 7.1.3.2.5
RESULTS

> .0000000000 .0000000000
> 0
> 0
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000 .0000000000

0
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
.0000000000
0
0

ALL ABOVE ANSWERS SHOULD BE ZERO IF THE READ AND WRITE RECORDS COMPARE.



MORDO $=(197)$ A MORE COMPLICATED SEG. OF DO STATEMENTS

ASA REFS $=7.1 .2 .8$ AND 7.1 .2 .8 .1
RESULTS

THIS SEGMENT SUCCESSFULLY TESTED IF NO ERROR MESSAGES

SUBRI $=(200)$ SUBROUTINE SIIBPROGRAM WITHOUT AN ARGUMENT LIST

ASA REF. - 8.4.1
RESULTS

THIS SEGMENT SUCCESSFULLY TESTED IF NO ERROR MESSAGES.

```
    FORT R A N T E S T P R O G F A M S
    PREPARED BY NATIONAL BUREAU OF STANDARDS
    FOR USE ON LARGE FORTRAN PROCESSORS
    IN ACCORDANCE WITH ASA FORTRAN X3.901966
    VERSION 3 PART 13
SAMPLE COMPUTER, FORTRAN COMPILER LEVEL
OPERATING SYSTEM VERSION
DATE, INSTALLATIGN NAME
```

```
LOGIF - (300) LOGICAL IF STATEMENT
```

ASA REF. - 7.1.2.3
RESULTS
TEST EXPLICITLY WRITTEN-SIGNEO-ZERO
+0 EQUALS -0
+0.0 EQUALS -0.0
$+0.0 D 0$ EQUALS - 0.000
TEST COMPUTATIONAL SIGN OF- ZERO -
40 EQUALS -0
+0.0 EQUALS -0.0
$+0.0 D 0$ EQUALS $=0.000$
TEST -LOGICAL IF - FOLLOWEO BY
OIFFERENT KINDS OF STATEMENTS
0
0
0
0
0
0
0
0
0
0

THERE SHOULD $3 E 10$ VALUES ABOVE, IF ONLY 9, TEST 9 HAS FAILED.

0
0
0
0
0
0
0
0

ALL VALUES SHOULD BE ZERO. a value other than zero will be fhe NUMBER OF THE TEST WHICH FAILED.

BARIF $=(301)$ BASIC FORTRAN ARI THMETIC IF STATEMENT

## ASA REF. - 7.1.2.2

## RESULTS

TEST FOR SIGN OF ZERO - TYPE INTEGER


TEST FOR SIGN OF ZERO - TYPE REAL


ALL ENTRIES SHOULD BE O EXCEPT
THE ZERO PATH, WHICH SHOULO BE 11 IN EACH COLUMN. OTHER TESTS MAY FAIL IF THESE RESULTS DIFFER.

TEST EXPRESSIONS IN IF STATEMENTS tests successful

FARIF - (302) FULL FORTRAN ARITHMETIC IF STATEMENTS
ASA REF. -7.1 .2 .2
RESULTS

SEGMENT 302 TESTED SUCCESSFULLY。

IOFMT (310) AODITIONAL FORMATTED $1 / 0$
ASA REFS 7, 1,3,2,2 7,1,3,2,3 7,2,3
RESULTS
TEST ELANK INPUT
EACH ANSWER SHOULD BE ZERO 0

0
0.0
0.0
0.0
0.0
$0,0 E \neq 00$
$0,0 E \neq 00$
$0.0 E \neq 00$
$0.0 E+00$
$0.00+00$
$0.00 \neq 00$
$0,00+00$
$0.00 \$ 00$

TEST DEC, PT, SPECIFIED BY INPUT 3 LINES IN EACH GROUP SHOULO MATCH

* LINE IS HOLLERITH DATA
-1.23456
1.23456
1.23456
+ 987654.0
987654.0
- 0.1234E+01
$0.1234 E+01$
$0.1234 E+01$
- . $\quad$. $0.987654 E \square 02$
$-0.987654 E+02$
-0.987654E 0.02
- $0.2345678910110+06$
$0.2345678910110+06$
0.2345678910110706
- $0.1098760-04$
$-0.1098760=04$
$-0.1098760-04$



FORTRANTESTPROGRAMS PREPARED BY NATIONAL BUREAU OF STANOARDS FOR USE ON LARGE FORTRAN PROCESSORS IN ACCORDANCE WITH ASA FORTRAN $\times 3.9-1965$ VERSION 3 PART 14

SAMPLE COMPUTER, FORTRAN COMPILER LEVEL OPERATING SYSTEM VERSION
date, installation name

```
MISC5 - (350) SPECIFICATIONS FOR
    PROGRAM FORM
ASA REFS. - 3.2 3.2.1 3.4 3.5
TEST THAT COMMENTS ARE NOT EXECUTED
TEST SUCCESSFUL IF NO ERROR MESSAGE
```


## TEST 72 CHARACTER LINE

$$
12345678910111213141516171819
$$

$$
12345678910111213141516171819
$$

TEST SUCCESSFUL IF 2 LINES AbOVE are OIGITS 1 THROUGH 19

```
TEST 1:2.3.4.5 CHARACTER STMNT. LABEL
```

1 CHARACTER LABEL ACCEPTED
2 CHARACTER LABEL ACCEPTED
3 CHARACTER LABEL ACCEPTED
4 CHARACTER LABEL ACCEPTED
5 CHARACTER LABEL ACCEPTED
TEST $1,2,3,4,5,6$ CHARACTER VARIABLES
AND ARRAY NAMES
**TEST SUCCESSFUL-ALL NAMES ACCEPTED**
TEST PLACEMENT OF STATEMENT LABELS
AND LABELS WITH LEADING ZEROS
1
2
3
4
5
6
7
8
9

TEST SUCCESSFUL IF 9 NUMBERS IN SEQUENTIAL ORDER FROM 1 TO 9 ARE WRITTEN ABOVE

END OF SEGMENT 350

FUNMX - (351)
THIS SEGMENT FURTHER TESTS
SOME BASIC EXTERNAL FUNCTIONS BY USING TRIGONOMETRIC FORMULAE

ASA REFS. - 8.3.3
RESULTS

> 0.00000
> -.00000
> -.00000
> .00000
-.00000
-.00000
.00000
0.00000
.00000
C. 00000
-. 00000

ALL ABOVE ANSWERS SHOULD BE O PLUS OR MINUS AN ERROR FACTOR OF NOT MORE THAN $10 * *(-4)$

NAMES - (352)
TEST OF THE COMPILERS CAPABILITY OF IDENTIFYING DATA NAMES THAT RESEMBLE FORTRAN VERBS AND/OR PREDEFINEO FUNCTION NAMES

ASA REFS. - $10.1 .7 / 4$
RESULTS

SPEC2 (360) COMMON AND EGUIVALENCE
ISA REFS . 7.2.1.2 7.2.1.3 7.2.1.4 RESULTS

LINE 1 BELOW IS HOLLERITH
$\begin{array}{ll}2 & 2,0 \\ 2 & 2,0\end{array}$

ANSWERS BELOW SHOULO BE O OR O,O
0
0
0
0,0
0
0
0
0,0

ARITHMETIC IF SUCCESSFUL

ANSWER BELOW SHOULD BE 13,0
13.0

COMPUTED GO TO SUCCESSFUL
TEST EQUIVALENCE EXTENDS COMMON
TEST SUCCESSFUL

```
0.00000
0.00000
0.00000
0.0000B
0.00000
0.00000
0.00000
0.00000
0.00000
    0.00000
        0
            0
    0.00000
    0.00000
ALL ABOVE ANSWERS SHOULD BE O FOR THIS TEST SEGMENT TO BE SUCCESSFUL

\section*{GENERAL DESCRIPTION}

This section of the document describes the organization of the NBS FORTRAN Test Programs and data as recorded on magnetic tape for distribution. When the programs have been retrieved and stored in a form more appropriate to utilization, this section of the manual is of no significance.

The distribution tape containing both Version 1 (116 executable test units) and Version 3 ( 14 executable programs containing the 116 test units) is available in 800 cpi recording density in the following forms:

7 track, even parity, BCD recorded from FORTRAN H set punch card code (See Appendix D X3.9-1966)
9 track, odd parity, EBCDIC recorded from the American National Standard punch card code
9 track, odd parity, ASCII recorded from the American National Standard punch card code

The distribution tape is an unlabeled, fixed block size recorded tape, terminating with two tape mark records.

Version 1 Programs and its data precede Version 3 with its data. Each block contains 720 characters comprised of nine \(80-\) character card image records. Partial blocks at the end of both Version 1 and Version 3 are filled with blank card images, so that Version 1 begins in Block 1 record 1 and Version 3 begins with Block 1597 record 1 .

The differences between the punch card code for the FORTRAN H Set and the American National Standard are reflected in the following four characters:
\begin{tabular}{lrl} 
& H-Set & Standard \\
() left parenthesis & \(0-4-8\) & \(12-5-8\) \\
) right parenthesis & \(12-4-8\) & \(11-5-8\) \\
\(=\) equal & \(3-8\) & \(6-8\) \\
+ plus & 12 & \(12-6-8\)
\end{tabular}

The programs and the data are in the same code.
For FORTRAN processors which contain an option on the coded character set for conversion of the FORTRAN programs, but not for the data, or perform a logical conversion only, causing the program listing to print a different character representation for the four characters listed above should perform a character conversion to the test programs and data before performing the tests, because the program listing is considered part of the documentation.

The following tables identify each main program unit, subprogram and data in two different forms:

The Block and Record number identifies the block number and the record within the block of the start of each element of information.

The card image number is the record number for the location of the start of each element of information.

For Version 3, one table lists the elements in relation to their position on the tape with Version 1 preceding it, and the other table assumes that the tape has been forward spaced over Version 1 (1596 blocks).

Each element of information in the tables is identified by the letter:
\(M=\) main program unit
\(F=\) function subprogram
S = subroutine subprogram
\(B=\) Block Data subprogram
WARNING - Version 1 and Version 3 each contain the same subprograms. If Version 1 and Version 3 are to be retained as a single file for use, one copy of the subprograms (63 functions and subroutines) must be deleted otherwise duplicate external procedure names will occur.

In Version 1, the Directory (segment 000) recorded as a set of comment lines is included as part of the first test unit, segment 008. This causes this test unit to contain 871 card images. The Directory of 342 card images may be removed and by appending a STOP statement and an END line may be compiled to obtain a program listing.


VERSION 1 DISTRIBUTION TAPE ORGANIZATION - continuation
\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|c|c|}
\hline Block & \& Record & \# Segment \# & Name & \begin{tabular}{l}
Card \\
Image
\end{tabular} & \# Block & \& Record & \# & Segment \# & Name & & \begin{tabular}{l}
Card \\
Image
\end{tabular} \\
\hline 1053 & 1 & 161 & BIFCP M & 9469 & 1196 & 7 & & 167 & SBRTN & M & 10762 \\
\hline 1062 & 7 & 401 & IAFI F & 9556 & 1208 & 2 & & 407 & AAQ & S & 10865 \\
\hline 1063 & 8 & 421 & IBFI F & 9566 & 1210 & 7 & & 417 & ABQ & S & 10888 \\
\hline 1064 & 9 & 431 & ICFI F & 9576 & 1212 & 2 & & 427 & ACQ & S & 10901 \\
\hline 1066 & 1 & 441 & IDFI F & 9586 & 1214 & 5 & & 168 & FSBRT & M & 10922 \\
\hline 1067 & 5 & 451 & IEFI F & 9599 & 1231 & 5 & & 408 & ADQ & S & 11075 \\
\hline 1068 & 7 & 461 & IFFI F & 9610 & 1235 & 8 & & 418 & AEQ & S & 11114 \\
\hline 1070 & 4 & 162 & FRFCP M & 9625 & 1238 & 4 & & 428 & AFQ & S & 11137 \\
\hline 1085 & 1 & 402 & GFS F & 9757 & 1242 & 9 & & 169 & BLKDT & M & 11178 \\
\hline 1086 & 3 & 422 & HFS F & 9768 & 1250 & 8 & & 409 & BLOKD & D & 11249 \\
\hline 1087 & 6 & 432 & IRFS F & 9780 & 1254 & 8 & & 179 & BLKDA & M & 11285 \\
\hline 1089 & 4 & 442 & JRFS F & 9796 & 1262 & 6 & & 419 & BLAKD & B & 11355 \\
\hline 1090 & 6 & 452 & RFS F & 9807 & 1265 & & & 429 & BLBKD & B & 11379 \\
\hline 1093 & & 163 & FIFCP M & 9836 & 1267 & 2 & & 439 & BLCKD & B & 11396 \\
\hline 1107 & 5 & 403 & IFI F & 9959 & 1269 & & & 180 & UNFRW & M & 11416 \\
\hline 1108 & 7 & 423 & JFI F & 9970 & 1284 & 2 & & 182 & BACUP & M & 11549 \\
\hline 1110 & 1 & 433 & KFI F & 9982 & 1292 & 4 & & 190 & DOTRM & M & 11623 \\
\hline 1111 & 8 & 443 & LFI F & 9998 & 1307 & 4 & & 191 & DOLMT & M & 11758 \\
\hline 1113 & 1 & 453 & MFI F & 10009 & 1314 & 3 & & 192 & DONSC & M & 11820 \\
\hline 1116 & 3 & 164 & CFCCP M & 10038 & 1332 & 7 & & 193 & DONSI & M & 11986 \\
\hline 1130 & 9 & 404 & AFC F & 10170 & 1339 & & & 194 & DONSX & M & 12046 \\
\hline 1132 & 1 & 414 & BFC F & 10180 & 1353 & 8 & & 195 & DONML & M & 12176 \\
\hline 1133 & 2 & 424 & CFC F & 10190 & 1361 & 1 & & 196 & DONIO & M & 12241 \\
\hline 1134 & 4 & 434 & DFC F & 10201 & 1371 & 2 & & 197 & MORDO & M & 12332 \\
\hline 1135 & 7 & 444 & EFC F & 10213 & 1390 & & & 412 & MDQ & S & 12510 \\
\hline 1136 & 9 & 454 & FFC F & 10224 & 1392 & 4 & & 200 & SUBR1 & M & 12523 \\
\hline 1138 & 6 & 464 & HFC F & 10239 & 1398 & 2 & & 410 & SUBRQ & S & 12575 \\
\hline 1141 & 7 & 165 & DPFCP M & 10267 & 1409 & 4 & & 300 & LOG IF & M & 12676 \\
\hline 1156 & 7 & 405 & AFD F & 10402 & 1439 & 9 & & 411 & SMCQ & S & 12951 \\
\hline 1157 & 8 & 415 & BFD F & 10412 & 1441 & 3 & & 301 & BARIF & M & 12963 \\
\hline 1158 & 9 & 425 & CFD F & 10422 & 1460 & 7 & & 302 & FARIF & M & 13138 \\
\hline 1160 & 2 & 435 & DFD F & 10433 & 1471 & 7 & & 310 & IOFMT & M & 13237 \\
\hline 1161 & 5 & 445 & EFD F & 10445 & 1506 & 2 & & & data cards & & 13547 \\
\hline 1163 & 3 & 455 & FFD F & 10461 & 1510 & 4 & & 312 & RDFMT & M & 13585 \\
\hline 1164 & 5 & 465 & GFD F & 10472 & 1532 & 7 & & 462 & FMTQ & S & 13786 \\
\hline 1165 & 8 & 475 & HFD F & 10484 & 1536 & 4 & & 13 d & data cards & & 13819 \\
\hline 1169 & 4 & 166 & BFCCP M & 10516 & 1537 & & & 350 & MISC5 & M & 13832 \\
\hline 1185 & 4 & 406 & AFB F & 10660 & 1555 & & & 351 & FUNMX & M & 13988 \\
\hline 1186 & 5 & 416 & BFB F & 10670 & 1561 & 6 & & 352 & NAMES & M & 14046 \\
\hline 1187 & & 426 & CFB F & 10680 & 1570 & 4 & & 413 & MAQQ & S & 14125 \\
\hline 1188 & 8 & 436 & DFB F & 10691 & 1572 & 1 & & 463 & MBQQ & S & 14140 \\
\hline 1190 & 1 & 446 & EFB F & 10702 & 1573 & 7 & & 473 & AMQQ & S & 14155 \\
\hline 1191 & 4 & 456 & FFB F & 10714 & 1576 & 1 & & 483 & BMQQ & S & 14176 \\
\hline 1192 & 7 & 466 & GFB F & 10726 & 1577 & 8 & & 360 & SPEC2 & M & 14192 \\
\hline 1193 & 9 & 476 & HFB F & 10737 & 1596 & 6 & & blan & nk card & & 14361 \\
\hline & & & & & 1596 & 9 & & last blan & nk card & & 14364 \\
\hline
\end{tabular}

A2.
VERSION 3 DISTRIBUTION TAPE ORGANIZATION


VERSION 3 DISTRIBUTION TAPE ORGANIZATION
(Listed as if Version 1 ( 1596 blocks) had been deleted or forward spaced.)
\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|c|c|c|}
\hline Block & \& Record & \# & Segment & \# Name & & \[
\begin{aligned}
& \text { Card } \\
& \text { Image \# }
\end{aligned}
\] & Block \& & Record \# & Segment & Name & & \[
\begin{array}{r}
\text { Card } \\
\text { Image }
\end{array}
\] \\
\hline 1 & 1 & & 008-011 & PART 1 & M & 1 & 1142 & 1 & 165-169 & PART 11 & M & 10270 \\
\hline 120 & 4 & & & 49 data cards & & 1075 & 1216 & 7 & 405 & AFD & F & 10942 \\
\hline 125 & 8 & & 013-015 & PART 2 & M & 1124 & 1217 & 8 & 415 & BFD & F & 10952 \\
\hline 228 & 7 & & & 6 data cards & & 2050 & 1218 & 9 & 425 & CFD & F & 10962 \\
\hline 229 & 4 & & 016-034 & PART 3 & M & 2056 & 1220 & 2 & 435 & DFD & F & 10973 \\
\hline 348 & 3 & & & 6 data cards & & 3126 & 1221 & 5 & 445 & EFD & F & 10985 \\
\hline 348 & 9 & & 035-053 & PART 4 & M & 3132 & 1223 & 3 & 455 & FFD & F & 11001 \\
\hline 473 & 1 & & & 6 data cards & & 4249 & 1224 & 5 & 465 & GFD & F & 11012 \\
\hline 473 & 7 & & 054-064 & PART 5 & M & 4255 & 1225 & 8 & 475 & HFD & F & 11024 \\
\hline 601 & 2 & & & 6 data cards & & 5402 & 1229 & 4 & 406 & AFB & F & 11056 \\
\hline 601 & 8 & & 065-073 & PART 6 & M & 5408 & 1230 & 5 & 416 & BFB & F & 11066 \\
\hline 702 & 5 & & & 6 data cards & & 6314 & 1231 & 6 & 426 & CFB & F & 11076 \\
\hline 703 & 2 & & 080-092 & PART 7 & M & 6320 & 1232 & 8 & 436 & DFB & F & 11087 \\
\hline 813 & 3 & & & 6 data cards & & 7311 & 1234 & 1 & 446 & EFB & F & 11098 \\
\hline 813 & 9 & & 094-111 & PART 8 & M & 7317 & 1235 & 4 & 456 & FFB & F & 11110 \\
\hline 918 & 9 & & & 6 data cards & & 8262 & 1236 & 7 & 466 & GFB & F & 11122 \\
\hline 919 & 6 & & 140-150 & PART 9 & M & 8268 & 1237 & 9 & 476 & HFB & F & 11133 \\
\hline 1026 & 8 & & & 6 data cards & & 9233 & 1240 & 7 & 407 & AAQ & & 11158 \\
\hline 1027 & 5 & & 160-164 & PART 10 & M & 9239 & 1243 & 3 & 417 & ABQ & S & 11181 \\
\hline 1097 & 9 & & 400 & AFS & F & 9873 & 1244 & 7 & 427 & ACQ & S & 11194 \\
\hline 1099 & 1 & & 420 & BFS & F & 9883 & 1247 & 1 & 408 & ADQ & S & 11215 \\
\hline 1100 & 2 & & 430 & CFS & F & 9893 & 1251 & 4 & 418 & AEQ & S & 11254 \\
\hline 1101 & 3 & & 440 & DFS & F & 9903 & 1253 & 9 & 428 & AFQ & S & 11277 \\
\hline 1102 & 5 & & 450 & EFS & F & 9914 & 1258 & 5 & 409 & BLOKD & B & 11318 \\
\hline 1103 & 7 & & 460 & FFS & F & 9925 & 1262 & 5 & & 6 data cards & & 11354 \\
\hline 1105 & 4 & & 401 & IAFI & F & 9940 & 1263 & 2 & 179-200 & PART 12 & M & 11360 \\
\hline 1106 & 5 & & 421 & IBFI & F & 9950 & 1402 & 3 & 410 & SUBRQ & S & 12612 \\
\hline 1107 & 6 & & 431 & ICFI & F & 9960 & 1413 & 5 & 412 & MDQ & S & 12713 \\
\hline 1108 & 7 & & 441 & IDFI & F & 9970 & 1414 & 9 & 419 & BLAKD & B & 12726 \\
\hline 1110 & 2 & & 451 & IEFI & F & 9983 & 1417 & 6 & 429 & BLBKD & B & 12750 \\
\hline 1111 & 4 & & 461 & IFFI & F & 9994 & 1419 & 5 & 439 & BLCKD & B & 12767 \\
\hline 1113 & 1 & & 402 & GFS & F & 10009 & 1421 & 7 & & 6 data cards & & 12787 \\
\hline 1114 & 3 & & 422 & HFS & F & 10020 & 1422 & 4 & 300-312 & PART 13 & M & 12793 \\
\hline 1115 & 6 & & 432 & IRFS & F & 10032 & 1543 & 3 & 411 & SMCQ & S & 13881 \\
\hline 1117 & 4 & & 442 & JRFS & F & 10048 & 1544 & 6 & 462 & FMTQ & S & 13893 \\
\hline 1118 & 6 & & 452 & RFS & F & 10059 & 1548 & 3 & & 57 data cards & & 13926 \\
\hline 1121 & 8 & & 403 & IFI & F & 10088 & 1554 & 6 & 350-360 & PART 14 & M & 13983 \\
\hline 1123 & 1 & & 423 & JFI & F & 10099 & 1610 & 8 & 413 & MAQQ & S & 14489 \\
\hline 1124 & 4 & & 433 & KFI & F & 10111 & 1612 & 5 & 463 & MBQQ & S & 14504 \\
\hline 1126 & 2 & & 443 & LFI & F & 10127 & 1614 & 2 & 473 & AMQQ & S & 14519 \\
\hline 1127 & 4 & & 453 & MFI & F & 10138 & 1616 & 5 & 483 & BMOQ & S & 14540 \\
\hline 1130 & 6 & & 404 & AFC & F & 10167 & 1618 & 3 & & 6 data cards & & 14556 \\
\hline 1131 & 7 & & 414 & BFC & F & 10177 & 1618 & 8 & & ast data card & & 14561 \\
\hline 1132 & 8 & & 424 & CFC & F & 10187 & 1618 & 9 & (blank & filler card) & & 14562 \\
\hline 1134 & 1 & & 434 & DFC & F & 10198 & & & & & & \\
\hline 1135 & 4 & & 444 & EFC & F & 10210 & \multicolumn{6}{|l|}{\(\mathrm{M}=\) Main Program} \\
\hline 1136 & 6 & & 454 & FFC & F & 10221 & \(\mathrm{F}=\mathrm{Func}\) & ction Subp & & & & \\
\hline 1138 & 3 & & 464 & HFC & F & 10236 & \multicolumn{6}{|l|}{S = Subroutine Subprogram} \\
\hline 1141 & 4 & \multicolumn{3}{|r|}{6 data cards} & & 10264 & \(B=B L O C\) & ck data Sub & program & & & \\
\hline
\end{tabular}
U.S. DEPT. OF COMM.

BIBLIOGRAPHIC DATA SHEET
4. TITLE AND SUBTITLE
2. Gov't Accession No.
NBS-SP399 Vol. 1

NBS FORTRAN TEST PROGRAMS

\section*{7. AUTHOR(S)}

Frances E. Holberton Elizabeth G. Parker
9. PERFORMING ORGANIZATION NAME AND ADDRESS

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234
2. Sponsoring Organization Name and Complete Address (Street, City, State, ZIP)

Same as 9
Library of Congress Catalog Card Number: 74-12314
3. Recipient's Accession No.
5. Publication Date

Ocłober 1974
6. Performing Organization Code
8. Performing Organ. Report No.
10. Project/Task/Work Unit No. 6401123
11. Contract/Grant No.
13. Type of Report \& Period Covered

\section*{Final}
14. Sponsoring Agency Code
5. SUPPLEMENTARY NOTES Volumes 1 , 2 , and 3 contain the documentation, Version 1 program listing, Version 3 program listing, respectively. The magnetic tape containing the NBS FORTRAN Test Programs is available in 7-track BCD, 9-track ASCII or EBCDIC recording.
6. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)
The NBS FORTRAN test programs, written in Standard FORTRAN, are designed to test whether a FORTRAN compiler accepts the forms and interpretations of the FORTRAN language as described in the American National Standard FORTRAN document X3.9-1966. The test programs, comprised of 116 test units, are structured into two versions, each containing approximately 14,500 punch card images. The test units may be used as separate executable FORTRAN programs, or may be linked end to end with other test units, with a minimum of user effort, to improve operating efficiency. Version 1 is structured into ll6 executable FORTRAN programs, and Version 3, containing the same 116 test units, is structured into 14 executable FORTRAN programs for use on large FORTRAN processors.

The test-program design criteria was to:
- Constrain all test programs to the FORTRAN Standard X3.9-1966.
- Reduce the effect of those areas in which the FORTRAN Standard does not prescribe a method or solution, e.g., range, precision, size of computer, etc.
- Simplify the use of the FORTRAN test programs.
- Test FORTRAN language elements before they are used in support of other tests.
. Maintain an open ended system so that tests may be changed or added.
The test programs require the use of a card reader, printer and one intermediate tape unit.
7. KEY WORDS (six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)
Computer programming language; FORTRAN: FORTRAN validation; language validation; standard FORTRAN; test program design.
\begin{tabular}{|c|c|c|}
\hline \begin{tabular}{l}
8. AVAILABILITY \\
X Unlimited
For Official Distribution. Do Not Release to NTIS
\end{tabular} & \begin{tabular}{l}
19. SECURITY CLASS (THIS REPORT) \\
UNCL ASSIFIED
\end{tabular} & 21. NO. OF PAGES
\[
171
\] \\
\hline X. Order From Sup. of Doc., U.S. Government Printing Office Washington, D.C. 20402, SD Cat. No. C13. 10:399/V. 1.
Order From National Technical Information Service (NTIS) Springfield, Virginia 22I5I & \begin{tabular}{l}
20. SECURITY CLASS (THIS PAGE) \\
UNCLASSIFIED
\end{tabular} & 22. Price \(\$ 2.35\) \\
\hline
\end{tabular}

\section*{PERIODICALS}

JOURNAL OF RESEARCH reports National Bureau of Standards research and development in physics, mathematics, and chemistry. Comprehensive scientific papers give complete details of the work, including laboratory data, experimental procedures, and theoretical and mathematical analyses. Illustrated with photographs, drawings, and charts. Includes listings of other NBS papers as issued.

Published in two sections, available separately:
- Physics and Chemistry (Section A)

Papers of interest primarily to scientists working in these fields. This section covers a broad range of physical and chemical research, with major emphasis on standards of physical measurement, fundamental constants, and properties of matter. Issued six times a year. Annual subscription: Domestic, \(\$ 17.00\); Foreign, \$21.25.
- Mathematical Sciences (Section B)

Studies and compilations designed mainly for the mathematician and theoretical physicist. Topics in mathematical statistics, theory of experiment design, numerical analysis, theoretical physics and chemistry, logical design and programming of computers and computer systems. Short numerical tables. Issued quarterly. Annual subscription: Domestic, \(\$ 9.00\); Foreign, \(\$ 11.25\).
DIMENSIONS/NBS (formerly Technical News Bul-letin)-This monthly magazine is published to inform scientists, engineers, businessmen, industry, teachers, students, and consumers of the latest advances in science and technology, with primary emphasis on the work at NBS.
DIMENSIONS/NBS highlights and reviews such issues as energy research, fire protection, building technology, metric conversion, pollution abatement, health and safety, and consumer product performance. In addition, DIMENSIONS/NBS reports the results of Bureau programs in measurement standards and techniques, properties of matter and materials, engineering standards and services, instrumentation, and automatic data processing.
Annual subscription: Domestic, \(\$ 6.50\); Foreign, \(\$ 8.25\)

\section*{NONPERIODICALS}

Monographs-Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.
Handbooks-Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.
Special Publications-Include proceedings of high-level national and international conferences sponsored by NBS, precision measurement and calibration volumes, NBS annual reports, and other special publications appropriate to this grouping such as wall charts and bibliographies.
Applied Mathematics Series-Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series-Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a world-wide program coordinated by NBS. Program under authority of National Standard Data Act (Public Law 90-396). See also Section 1.2.3.

Building Science Series-Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.
Technical Notes-Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards-Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The purpose of the standards is to establish nationally recognized requirements for products, and to provide all concerned interests with a basis for common understanding of the characteristics of the products. The National Bureau of Standards administers the Voluntary Product Standards program as a supplement to the activities of the private sector standardizing organizations.
Federal Information Processing Standards Publications (FIPS PUBS)-Publications in this series collectively constitute the Federal Information Processing Standards Register. The purpose of the Register is to serve as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 ( 38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations). FIPS PUBS will include approved Federal information processing standards information of general interest, and a complete index of relevant standards publications.
Consumer Information Series-Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.
NBS Interagency Reports-A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service (Springfield, Va. 22151) in paper copy or microfiche form

Order NBS publications (except Bibliographic Subscription Services) from: Superintendent of Documents, Government Printing Office, Washington, D.C. 20402.

\section*{BIBLIOGRAPHIC SUBSCRIPTION SERVICES}

The following current-awareness and literature-survey bibliographies are issued periodically by the Bureau:
Cryogenic Data Center Current Awareness Service (Publications and Reports of Interest in Cryogenics). A literature survey issued weekly. Annual subscription: Domestic, \(\$ 20.00\); foreign, \(\$ 25.00\).
Liquefied Natural Gas. A literature survey issued quarterly. Annual subscription: \(\$ 20.00\).
Superconducting Devices and Materials. A literature survey issued quarterly. Annual subscription: \(\$ 20.00\). Send subscription orders and remittances for the pre-
ceding bibliographic services to the U.S. Department of Commerce, National Technical Information Service, Springfield, Va. 22151.

Electromagnetic Metrology Current Awareness Service (Abstracts of Selected Articles on Measurement Techniques and Standards of Electromagnetic Quantities from D-C to Millimeter-Wave Frequencies). Issued monthly. Annual subscription: \(\$ 100.00\) (Special rates for multi-subscriptions). Send subscription order and remittance to the Electromagnetic Metrology Information Center, Electromagnetics Division, National Bureau of Standards, Boulder, Colo. 80302.

\section*{U.S. DEPARTMENT OF COMMERCE}

\section*{National Bureau of Standards}

Washington, D.C. 20234

Penalty for Private Use, \(\$ 300\)```


[^0]:    ${ }^{1}$ Headquarters and Laboratories at Gaithersburg, Maryland, unless otherwise noted; mailing address Washington, D.C. 20234.
    ${ }^{2}$ Part of the Center for Radiation Research.
    ${ }^{3}$ Located at Boulder, Colorado 80302.
    \& Part of the Center for Building Technology.

[^1]:    *These statement numbers appear in a main program and a subprogram.

