Ref.

ED STATES
TMENT OF
MERCE
ICATION

NBS TECHNICAL NOTE 590

A Preliminary Design of a Data Retrieval Language to Handle a Generalized Data Base: DRL

U.S.
RTMENT
OF
MMERCE
National
Bureau
of
andards

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 Center 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—Heat—Mechanics—Optical Physics—Linac Radiation[2]—Nuclear Radiation[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:

> Analytical Chemistry—Polymers—Metallurgy—Inorganic Materials—Reactor Radiation—Physical Chemistry.

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 also monitors NBS engineering standards activities and provides liaison between NBS and national and international engineering standards bodies. The Institute consists of the following technical divisions and offices:

> Engineering Standards Services—Weights and Measures—Flammable Fabrics—Invention and Innovation—Vehicle Systems Research—Product Evaluation Technology—Building Research—Electronic Technology—Technical Analysis—Measurement Engineering.

THE CENTER 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 Center consists of the following offices and divisions:

> Information Processing Standards—Computer Information—Computer Services—Systems Development—Information Processing 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, and directs the public information activities of the Bureau. The Office consists of the following organizational units:

> Office of Standard Reference Data—Office of Technical Information and Publications—Library—Office of Public Information—Office of International Relations.

---

[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.

UNITED STATES DEPARTMENT OF COMMERCE
Maurice H. Stans, Secretary
NATIONAL BUREAU OF STANDARDS • Lewis M. Branscomb, Director

NBS TECHNICAL NOTE 590

ISSUED JULY 1971
Nat. Bur. Stand. (U.S.), Tech. Note 590, 26 pages (July 1971)
CODEN: NBTNA

A Preliminary Design of a Data Retrieval Language to Handle a Generalized Data Base: DRL

Elizabeth Fong

Systems Development Division
Center for Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

NBS Technical Notes are designed to supplement the Bureau's regular publications program. They provide a means for making available scientific data that are of transient or limited interest. Technical Notes may be listed or referred to in the open literature.

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C., 20402.
(Order by SD Catalog No. C 13.46:590). Price 35 cents.
Stock Number 0303-0886

TABLE OF CONTENTS

I. INTRODUCTION 2

II. BRIEF DESCRIPTION OF LANGUAGE. 3

II.1 Input/Output Commands. 4

II.2 Data Description Commands. 7

II.3 Data Maintenance and Manipulation Commands 9

II.4 Data Retrieval Commands. 11

II.5 Higher Order Functions 13

II.6 Built-In String Manipulation Functions 14

III. METHOD OF IMPLEMENTATION 14

IV. OPERATION. 18

V. REFERENCES 18

Appendix A
A Sample Run of DRL A-1

A PRELIMINARY DESIGN OF A DATA RETRIEVAL LANGUAGE TO HANDLE A GENERALIZED DATA BASE: DRL

ELIZABETH FONG

DRL (Data Retrieval Language) is a high-level programming language for information retrieval. The language includes a data description language which can describe fixed-length hierarchical data structures, and DRL includes a data retrieval statement whereby a user can retrieve data by specifying conditions on to the data value. DRL also has an environment declaration statement in which the user can indicate specific peripheral devices by unit number for files. The rest of the language consists of an operation repertory of input-output functions and other data manipulations.

DRL is implemented as a preprocessor to FORTRAN V on the UNIVAC 1108, under EXEC II Operating system. Keywords act as triggers and are replaced by blocks of FORTRAN code.

The purpose of this project is to investigate the design of an information retrieval language to handle a generalized data base. The DRL system consists of a set of primitives utilizing both compile-time macros and run-time subroutines. These primitives are embedded in a high-level procedure-oriented programming language--the "host language" -- FORTRAN in this case. These primitives form a base upon which a class of languages can be defined.

Key words: Data base; data retrieval; data structure; information storage and retrieval; language extension; preprocessor; programming language.

I. Introduction

In implementing an information storage and retrieval system, one is faced with the problem of choosing a suitable programming language. Requirements of that programming language are:

- ability to define data structures
- ability to describe an environment
- ability to manipulate data structures
- ability to address data by content
- ability to offer computational power comparable to FORTRAN.

It is possible to implement information storage and retrieval system in procedure-oriented language such as ALGOL, FORTRAN, COBOL, or PL/1, but it is unnatural and requires indirectness in the use of primitives of the language. For example, let us consider writing a program to do the following simple task:

> Fine the first element of the array A, of length N, whose value is equal to 3 and set I equal to the value of the index. If none, set I equal to zero.

In a procedural-oriented language such as FORTRAN, one would say:

```
      DO 10 I = 1,N
      IF (A(I) .EQ. 3) GO TO 20
10    CONTINUE
      I = 0
20    . . .
```

Using languages such as LISP, SNOBOL, L6, etc. , in doing information retrieval seems even more awkward. COBOL has file manipulation capability but lacks the ability of addressing data by content.[5] This leads to the notion of addressing data by content, which is more natural to a user, who is not a professional programmer. Hence the design of an information retrieval language should be convenient and natural for the user, yet the language should be powerful enough to handle complicated types of data structure. Such a language must have primitives at the level of <u>what</u> is to be done rather than <u>how</u> it is to be done.

An information retrieval language called DRL (<u>D</u>ata <u>R</u>etrieval <u>L</u>anguage) has been designed and is partially implemented at the National Bureau of Standards to meet these objectives on an experimental basis. The DRL language is designed as an extension to FORTRAN explicitely to include the primitives necessary for an information retrieval language. The language will enable us to investigate the benefits of this approach to retrieval problems.

II. Brief Description of the Language

The DRL language is embedded in the FORTRAN V language system on the UNIVAC 1108 computer. It will allow all the usual FORTRAN capabilities plus the following four classes:

1. Input/Output statements
2. Data description statements
3. Data maintenance and manipulation statements
4. Data retrieval statements

II.1 Input/Output Commands

The input/output commands include the peripheral device declaration and also record accessing commands. The input/output devices permitted at present are card reader, card punch, printer, magnetic tape and drum. The logical unit number is the same as FORTRAN V standard table as set up at National Bureau of Standards, Gaithersburg.

a. ENVIRO

The ENVIRONMENT declaration provides information about the physical location of the data set associated with a file. This information allows the preprocessor to determine the method of accessing the data set, and causes the tape or drum to position at the beginning.

FORMAT

ENVIRO (<filename >, <logical unit no>)

<filename>::= A FORTRAN variable.

<logical unit no> ::= An integer variable/integer.

Allowable values for the logical unit numbers and their assignments are listed as follows:

LOGICAL UNIT	ASSIGNMENT
0	Reread
1	Card reader
2	Printer
3	Card punch
4	Console

5	Card reader
6	Printer
7 - 32	Tapes A - Z
33	Tape (
34	Tape -
35 and 36	Entire Drum 1300000 to 3777777 (octal)
37 and 38	Lower Half 1300000 to 2537777 (octal)
39 and 40	Upper Half 2540000 to 3777777 (octal)
41 and 42	Lower third 1300000 to 2177777 (octal)
43 and 44	Middle third 2200000 to 3077777 (octal)
45 and 46	Upper third 3100000 to 3777777 (octal)

EXAMPLE

ENVIRO (PAYROL, 35)

The above means declaring the entire drum as a mass storage device for a file called PAYROL.

b. PUTOUT

PUTOUT will write a record currently set up in core onto the indicated peripheral device. Unless a FORTRAN format label is supplied, the output is assumed to be binary.

FORMAT

PUTOUT (<where> , {<label> ,} <filename>)

<where>::= peripheral device unit number. It is assumed that the device is positioned to be written.

<filename>::= A FORTRAN variable.

<label>::= FORTRAN format label. This field is optional

EXAMPLE

PUTOUT (35, PAYROL)

or

PUTOUT (6,100, PAYROL)

100 FORMAT (1X , 22A6)

The first PUTOUT example means write out a record which is in main storage array PAYROL onto the previously positioned drum. The second PUTOUT example means write on the printer the record according to the FORTRAN format statement labeled 100.

c. GETIN

GETIN reads in a record from the indicated peripheral device onto the record image space in core. Unless a FORTRAN format label is supplied, the input is assumed to be binary.

FORMAT

GETIN (<where> , {<label> ,}<filename>)

<where>::= peripheral device unit number. It is assumed that the device is positioned to be read.

<filename>::= A FORTRAN variable.

<label>::= FORTRAN format label. This field is optional

EXAMPLE

GETIN (35, PAYROL)

or

GETIN (5, 100, PAYROL)

100 FORMAT (80A1)

The first GETIN example reads one record from the previously positioned drum into the main storage array PAYROL. The second GETIN example reads from the card reader according to FORTRAN format statement labeled 100 into the array PAYROL.

II.2 Data Description Commands

Facility for declaring hierarchically structured data is provided. The declaration format is patterned after PL/1 where the level of hierarchy is indicated by the level number in front of the variables in the declaration. When the variable occurs without the level number in front, it is assumed that the declaration is merely a single data item or an array. To facilitate character manipulation, the string declaration is added to the FORTRAN type statements.

a. DECLAR CHARAC or BITS

DECLAR with the data type CHARAC means the variable being declared is a string of n six-bit Fieldata code as defined for the UNIVAC 1108. DECLAR with the data type BITS means the variable being declared is of n bits taking on values one or zero.

FORMAT

```
DECLAR ( <variable> CHARAC  ( <n> ))
DECLAR ( <variable>  BITS   ( <n> ))
```

<variable> ::= A FORTRAN Variable.

<n> ::= An integer greater than 0.

EXAMPLES

```
DECLAR ( NAME CHARAC (10))
DECLAR ( MATRIX  BITS (8))
```

b. DECLAR hierarchically

DECLAR with a hierarchical data list declares a data structure consisting of elementary data items (terminal nodes) and composite data items (non-terminal nodes or meta syntactical variable). The composite data items must have one or more subordinates and the elementary data items must have data type and size specifications associated with them.

FORMAT

```
DECLAR ( <hierarchical data list> )

<hierarchical data list>::= <hierarchical data>/

     <hierarchical data list>, <hierarchical data>

<hierarchical data>::= <n> <variable>/

                       <n> <variable> <type> <dimension>

<type>::= all legal FORTRAN data types (only first six char-

          acters) plus CHARAC and BITS

<dimension>::= ( <n> )

<variable>::= A FORTRAN variable

<n>::= An integer greater than 0
```

EXAMPLE

```
DECLAR ( 0 PAYROL,
           1  NAME,
              2  FIRST CHARAC (10),
              2  MIDDLE CHARAC (10),
              2  LAST CHARAC (10),
           1  SALARY,
              2  REGU INTEGE (1),
              2  OVER INTEGE (1),
           1  OCC CHARAC (20) )
```

II.3 Data Maintenance and Manipulation Commands

a. PUT

PUT assumes a data structure into which values are to be stored. If the attribute name happens to be an elementary data item, then the value is simply put in. If the attribute is not an elementary data item, then the lists of values to fill the attributes subordinate to it must be given.

FORMAT

```
PUT ( <attribute>, <value list> )

<attribute>::= elementary data item variable or composite
               data item variable as declared in DECLAR
               statement

<value list> ::= <value>/<value list>, <value>

<value>::=  any expression (The present version can only
               handle constants, literals and variables)
```

EXAMPLE

The following example assumes the declaration which appears in the DECLAR hierarchically example given above.

```
PUT (LAST, 'FONG')

PUT (NAME, 'LIZ', 'NEE', 'FONG')
```

The first PUT expression whose first argument is the elementary data item LAST and therefore the value is immediately assigned. In the second PUT expression the first argument is the composite data item NAME consisting of three subordinates and therefore the three values 'LIZ', 'NEE', and 'FONG' are assigned to FIRST, MIDDLE and LAST respectively.

b. LOCATE

Records within a file have an ordinal number according to their position within the file. LOCATE positions the file with respect to this ordinal number of the record.

FORMAT

LOCATE (<filename>, <index>)

<filename> ::= A FORTRAN variable defined in ENVIRO and DECLAR statements

<index> ::= An integer / An integer variable. If index = 0 the file is position to the beginning of the file.

EXAMPLE

LOCATE (PAYROL, 5)

LOCATE (PAYROL, IXGET)

c. DELETE AND DELIX

DELETE deletes the first encountered record which satisfies the given condition list.

DELIX deletes the Nth record in the file where N is given.

FORMAT

DELETE (<filename> , <condition list>)

DELIX (<filename >, <index>)

<condition list> ::= This is the same as described under the Get command

<filename> ::= A FORTRAN variable defined in ENVIRO and DECLAR statements

<index> ::= An integer / An interger variable

EXAMPLE

The following example assumes the declaration which appears in the DECLAR (hierarchically) example.

DELETE (PAYROL, LAST EQ 'SMITH')

DELIX (PAYROL, 5)

II.4 Data Retrieval Commands

a. GET

GET is a retrieval function which returns the value of the specified attribute in the first encountered record which satisfies the given conditions.

FORMAT

A = GET (< filename > , <attribute> , < condition list >)

< filename > ::= A FORTRAN variable declared in ENVIRO and DECLAR statements. It is assumed that the peripheral device is positioned to start searching.

<attribute> ::= Elementary data item variable or complex data item variable as declared in DECLAR statement.

<condition list > ::= < wff >

< wff > ::= <proposition> / <wff><connectives><wff>

<proposition > ::= < attribute><rel><value>

<connectives > ::= AND / OR

<rel > ::= EQ / NE / GT / GE / LT / LE

<value> ::= Any expression

EXAMPLE

The following example assumes the declaration which appears in the DECLAR (hierarchically) example.

A = GET (PAYROL, NAME, REGU EQ 400 AND OCC EQ 'MATH')

This GET command will search the PAYROL file. If the field REGU equal 400 and the field OCC equals MATH, then the field NAME will be retrieved and stored in A. A must be properly dimensioned.

DEFAULT CONDITIONS

If an error occurs, RUNERR routine is executed. RUNERR is a routine which may be supplied by the user to handle error recovery. If the user does not supply a RUNERR routine, the DRL system will execute the UNIVAC EXEC II error routine which will just stop execution.

REMARK

If there is no second argument of the GET command, i.e., two commas with nothing in between, then it is assumed that the index value is required. In any case an internally defined variable called IXGET will always contain the index value after each GET function. The IXGET value will be destroyed upon initiation of the next GET function.

b. GETALL

GETALL is the same as GET except instead of retrieving a single item, a whole set is retrieved. The variable occurring on the left of the GETALL statement must be pre-declared as one dimensional array with an estimate of maximum size. After retrieving, the first entry of that array will contain the count of the number of items retrieved followed by the values. The user must also DIMENSION the IXGET to be one dimensional array of maximum size. After each execution of GETALL, IXGET (1) contains the count of the number of items retrieved followed by the list of pointer to the retrieved records.

II.5 Higher Order Functions

These functions could be defined by combining appropriate previously defined primitive functions.

a. REPLACE

REPLACE may be defined by combining the following four DRL primitives.

GET - get a whole record that meets given conditions

PUT - change value as desired

LOCATE - position back

PUTOUT - put back the modified record in the file.

b. SUBSET

SUBSET may be defined by the following pseudo statements.

```
   ENVIRO - declared a different unit number for a subfile
10 GET    - get a record that meets the given conditions
   IF end-of-file THEN stop
   PUTOUT - putout on the new unit
   GO TO 10
```

An alternative way of defining assumes the existence of the primitive MOVE. This may be defined as a user's subroutine.

ENVIRO - declare a different unit number for a subfile

GETALL - get all records that meets the given conditions

MOVE - move to individual buffers for a unit record

PUTOUT - putout on the new unit.

c. COUNT

COUNT may be defined with the GETALL command and reading out the first word of the IXGET array or the user-defined array containing the answers.

II.6 Built-in String Manipulation Functions

There exists in the DRL system a group of run-time subroutines which are accessible to the user. The following is a set of string i.e., characters manipulation functions which are patterned after PL/1. The string variables must be declared as characters in a DECLAR CHARAC statement.

CONC (A.B) is the concatenation of string A and B.

LENGTH (A) is the length of string A.

INDEX (A, 'B') returns the starting location for the first occurrence of 'B' in string A or zero if not found.

SUBSTR (A, I, J) extracts the substring starting at position I of length J. If J equals 0 the rest of String A is returned.

MATCH (A,B,N) compares the first N characters of A to B. If they are identical, the value of MATCH is true, otherwise the value of MATCH is false.

III. Method of Implementation

The DRL language translator is a preprocessor to FORTRAN V on the UNIVAC 1108. It consists of two major phases:

Phase 1 - A scanner reads the input stream and traps all the DRL keywords and replaces them with appropriate blocks of FORTRAN code.

The declarative statements generate FORTRAN dimensioning statements and tables containing the data descriptions.

Phase 2 - A collection of predefined run-time subroutines to perform all of the above described tasks.

Both phase 1 and phase 2 work. All of the primitives defined above have been implemented. The higher order functions have not been implemented.

The main routine is the lexical scanner called LSCAN. This routine reads the DRL statements and branches according to the keywords scanned. The DRL syntax conforms to the FORTRAN statement format. If a given statement does not contain a DRL keyword, then it is assumed to be a FORTRAN statement, and the line is carried over to the generated program. The DRL keyword analysis is described individually as follows:

a. DROP - This should be the first statement of the DRL program. The operand of this statement defines a name for the output FORTRAN source.

b. ENVIRO - This statement generates the equivalent of an open file statement by positioning the peripheral device indicated by the unit number to the beginning. The file name is entered into the FILNAM table.

Layout of FILNAM table

FILNAM

FPOINT	
FILENAME	
Unit no.	block size
⋮	

pointer to next empty entry

c. DECLAR - This generates FORTRAN DIMENSION statements. If it is hierarchical data declaration, then it generates DIMENSION statements and EQUIVALENCE statements for all the complex data item names. Also the hierarchical data information is stored in the 8-column matrix J. The meaning of these 8 columns are as follows:

- J(I,1) contains the hierarchical level number
- J(I,2) contains the BCD symbol table index number
- J(I,3) contains the degree or number of subtrees of that node
- J(I,4) contains the index no. of the parent node
- J(I,5) contains the index no. of the sister node
- J(I,6) contains the next occupant of the same name
- J(I,7) contains the size in characters
- J(I,8) contains the starting character position from the beginning.

The details of the J-matrix can be found in Lawson [2].

The variable names are also entered into the SYMBOL table.

Layout of SYMBOL table

SYMBOL

SPOINT		
SYMBOL in BCD		
Type	Size	Index to J
⋮		

Pointer to next available entry

Type = 1 means character

Type = 2 means BITS

Type = 0 means other

SIZE = no. of characters or bits.

d. PUTOUT - This generates an appropriate output statements including calls to NTRAN if the output is to be binary. NTRAN is a FORTRAN callable subroutine which provides buffered input/output routines for tape and drum. Detailed description of NTRAN can be found in section 7.5 of the UNIVAC 1107 FORTRAN manual [3].

e. GETIN - This is the same as PUTOUT except it generates appropriate output statements.

f. GET - This will first generate a NTRAN call to read in a record. A boolean function containing the conditions given is next generated. On the 'true' branch the extraction of the specified attribute code is supplied. On the 'false' branch, a GOTO statement back to the NTRAN call to read in the next record is generated.

g. GETALL - Same as GET except a loop is set up to continuing retrieving until an end of file is reached.

h. PUT - This will generate a DO Loop containing a call to a run-time routine called MOVECH. MOVECH will move a character from the indicated source to the indicated destination.

i. LOCATE - This will generate a NTRAN call to position the peripheral device N records from the beginning. If the device is drum, the drum address is positioned from the beginning by the amount N times the length of the record.

j. DELETE - (This command is to be used primarily with drum) This statement will first call GET to determine which record meets the conditions given. Then it calls LOCATE to position the file to this record. The record is then zeroed. Garbage collection is not yet coded.

k. DELIX - (This command is to be used primarily with drum). This first calls LOCATE to position the file to the Nth record, and the record is zeroed.

IV. Operation

The input to the DRL translator is the program text written in the DRL language. The output is a FORTRAN program automatically residing on the drum, linked-edited, and ready to be executed. This output FORTRAN program, together with the predefined run-time action routines and block data, will be the final executable program capable of manipulating data, accessing the peripheral storage and performing any kind of retrieval tasks. This present version consists of approximately 1900 lines of FORTRAN and approximately 300 lines of UNIVAC assembly code.

The source listing of the entire DRL system is available from the author upon request.

The author is deeply indebted to Mr. Charles T. Meadow for first suggesting the topic and for his interest throughout the implementation.

V. References

[1] UNIVAC 1108 FORTRAN V-Programmer's Reference Manual UP4060, Sperry Rand Corporation, 1966.

[2] Lawson, Harold W., Jr., "The Use of Chain List Matrices for the Analysis of COBOL Data Structures," presented at the ACM National Conference, 1962.

[3] UNIVAC 1107 FORTRAN IV - Programmer's Reference Manual, UP-3569, Sperry Rand Corporation, 1966.

[4] IBM System/360 PL/1 Reference Manual, File No. S 360-29, Form No. C 28-8201-1, IBM 1968.

[5] Hanlon, A.G., "Content - Addressable and Associative Memory Systems - A Survey", IEEE Transaction on Electronic Computers, vol. EC-15, No. 4, August, 1966.

APPENDIX A -- A Sample Run of DRL

In this appendix, the following three outputs are presented:

(1) DRL sample program. These DRL statements are translated into FORTRAN codes which appear indented to the right. The data description table, symbol table, and file name table are generated as FORTRAN assignment statements.

(2) FORTRAN compilation of the DRL generated program.

(3) Execution of the sample program. The sample program reads in from the card reader a file of personnel records. It sets up the data base on drum. Two retrieval commands are executed.

```
        DROP ABC
                                               IMPLICIT INTEGER (A-Z)
        ENVIRO (PAYROL,35)
                                               CALL  NTRAN(    35, 10 )
        DECLAR (O PAYROL,
       X   -       1 NAME,
       X              2 FIRST CHARAC (12),
       X              2 MIDDLE CHARAC (12),
       X              2 LAST CHARAC (12),
       X           1 SALARY,
       X              2 REGULR INTEGE (1),
       X              2 OVER INTEGE (1) )
                                               DIMENSION   PAYROL  (        9)

                                               EQUIVALENCE(NAME  ,     PAYROL)
                                               EQUIVALENCE(FIRST ,     PAYROL)
                                               EQUIVALENCE(MIDDLE,     PAYROL)
                                               EQUIVALENCE(LAST  ,     PAYROL)
                                               EQUIVALENCE(SALARY,     PAYROL)
                                               EQUIVALENCE(REGULR,     PAYROL)
                                               EQUIVALENCE(OVER  ,     PAYROL)
                                               COMMON/MATRIX/INO,ILIM,SP,J(50,8)
                                               COMMON/STABLE/SYMBOL(200),FILNAM(20)
                                               LOGICAL TESTRL
                                               GO TO      1
                                          2    CONTINUE
        DECLAR (ANS CHARAC (36))
                                               DIMENSION   ANS      (        7)

        ANS (7)=''
                                               ANS (7)=''
        DO 600 I=1,7
                                               DO 600 I=1,7
        GETIN (5,500,PAYROL)
                                               READ (5,       500)      PAYROL
    500 FORMAT (3(A6,A4), 2I6,A6)
                                          500  FORMAT (3(A6,A4), 2I6,A6)
        PUTOUT (6,501,PAYROL)
                                               WRITE(6,       501)      PAYROL
    501 FORMAT (1X, 'NAME=', 3(A6,A4), ' REG=', I6, ' OVER=', I6)
                                          501  FORMAT (1X, 'NAME=', 3(A6,A4), ' REG=', I6, ' OVER=', I6)
        PUTOUT (35,PAYROL)
                                               ST=1
                                               CALL NTRAN (    35,1,         9,      PAYROL,ST)
                                               CALL CHECKS (ST, $     3)
                                               GO TO      4
                                             3 WRITE (6,        5)
                                             5 FORMAT(1X,11HEND OF FILE)
                                               CALL  RUNERR
                                             4 CONTINUE
    600 CONTINUE
                                          600  CONTINUE
        LOCATE (PAYROL,0)
                                               CALL  NTRAN(    35, 10)
        ANS=GET (PAYROL.NAME,LAST EQ 'LEAR')
                                               IXGET=0
                                               ST=1
                                               ANS    (1)=0
                                             6 IXGET=IXGET+1
                                               CALL  NTRAN(    35,2,         9,      PAYROL,ST)
                                               CALL CHECKS (ST, $     7)
                                               IF (          TESTRL(1,'LAST   ',LAST   ,'LEAR'))GOTO      8
                                               GO TO      6
                                             8 CALL SUBSTR(PAYROL,    9,     1,   36,      ANS    ,    36)
                                             7 CONTINUE
        PUTOUT (6,502,ANS)
                                               WRITE(6,       502)      ANS
    502 FORMAT (1H0,1X, 'NAME WHERE LAST EQ LEAR =', 7A6)
                                          502  FORMAT (1H0,1X, 'NAME WHERE LAST EQ LEAR =', 7A6)
        LOCATE (PAYROL,0)
                                               CALL  NTRAN(    35, 10)
        ANS=GET(PAYROL,NAME,REGULR GT 700)
                                               IXGET=0
                                               ST=1
                                               ANS    (1)=0
                                             9 IXGET=IXGET+1
                                               CALL  NTRAN(    35,2,         9,      PAYROL,ST)
                                               CALL CHECKS (ST, $    10)
                                               IF (          TESTRL(3,'REGULR',REGULR,700    ))GOTO     11
                                               GO TO      9
                                            11 CALL SUBSTR(PAYROL,    9,     1,   36,      ANS    ,    36)
                                            10 CONTINUE
        PUTOUT (6,503,ANS)
                                               WRITE(6,       503)      ANS
    503 FORMAT (1H0,1X,'NAME WHERE REG GT 700=', 7A6)
                                          503  FORMAT (1H0,1X,'NAME WHERE REG GT 700=', 7A6)
        STOP
                                               STOP
        ANS=GETALL (PAYROL.NAME,OVER GT 500)
                                               I=1
                                               IXGET=0
                                               ST=1
                                               ANS    (1)=0
                                            12 IXGET=IXGET+1
                                               CALL  NTRAN(    35,2,         9,      PAYROL,ST)
                                               CALL CHECKS (ST, $    13)
                                               IF (          TESTRL(3,'OVER   ',OVER   ,500    ))GOTO     14
                                               GO TO     12
                                            14 CALL SUBSTR(PAYROL,    9,     1,   36,      ANS    (I+  6),  36)
                                               ANS    (1)=  ANS    (1)+1
                                            13 CONTINUE
        END
                                               STOP
                                          1    CONTINUE
                                               ILIM =    10
                                               J (         1      ,      1)  =        0
                                               J (         1      ,      2)  =        2
                                               J (         1      ,      3)  =        2
                                               J (         1      ,      4)  =        0
                                               J (         1      ,      5)  =        9
                                               J (         1      ,      6)  =        6H@@@@@@
                                               J (         1      ,      7)  =        48
                                               J (         1      ,      8)  =        0
                                               J (         2      ,      1)  =        1
                                               J (         2      ,      2)  =        4
                                               J (         2      ,      3)  =        3
```

```
J (     2   ,   4)  =   1
J (     2   ,   5)  =   6
J (     2   ,   6)  =   6H@@@@@@
J (     2   ,   7)  =   36
J (     2   ,   8)  =   0
J (     3   ,   1)  =   2
J (     3   ,   2)  =   6
J (     3   ,   3)  =   0
J (     3   ,   4)  =   2
J (     3   ,   5)  =   4
J (     3   ,   6)  =   6H@@@@@@
J (     3   ,   7)  =   12
J (     3   ,   8)  =   0
J (     4   ,   1)  =   2
```

```
BEFORE CALLING DRUM WRITE
DRUM WRITE CALLED
```

```
J (     4   ,   2)  =   8
J (     4   ,   3)  =   0
J (     4   ,   4)  =   2
J (     4   ,   5)  =   5
J (     4   ,   6)  =   6H@@@@@@
J (     4   ,   7)  =   12
J (     4   ,   8)  =   12
J (     5   ,   1)  =   2
J (     5   ,   2)  =   10
J (     5   ,   3)  =   0
J (     5   ,   4)  =   2
J (     5   ,   5)  =   3
J (     5   ,   6)  =   6H@@@@@@
J (     5   ,   7)  =   12
J (     5   ,   8)  =   24
J (     6   ,   1)  =   1
J (     6   ,   2)  =   12
J (     6   ,   3)  =   2
J (     6   ,   4)  =   1
J (     6   ,   5)  =   2
J (     6   ,   6)  =   6H@@@@@@
J (     6   ,   7)  =   12
J (     6   ,   8)  =   36
J (     7   ,   1)  =   2
J (     7   ,   2)  =   14
J (     7   ,   3)  =   0
J (     7   ,   4)  =   6
J (     7   ,   5)  =   8
J (     7   ,   6)  =   6H@@@@@@
J (     7   ,   7)  =   6
J (     7   ,   8)  =   36
J (     8   ,   1)  =   2
J (     8   ,   2)  =   16
J (     8   ,   3)  =   0
J (     8   ,   4)  =   6
J (     8   ,   5)  =   7
J (     8   ,   6)  =   6H@@@@@@
J (     8   ,   7)  =   6
J (     8   ,   8)  =   42
J (     9   ,   1)  =   0
J (     9   ,   2)  =   18
J (     9   ,   3)  =   0
J (     9   ,   4)  =   0
J (     9   ,   5)  =   10
J (     9   ,   6)  =   6H@@@@@@
J (     9   ,   7)  =   36
J (     9   ,   8)  =   0
J (    10   ,   1)  =   0
J (    10   ,   2)  =   20
J (    10   ,   3)  =   0
J (    10   ,   4)  =   0
J (    10   ,   5)  =   0
J (    10   ,   6)  =   6H@@@@@@
J (    10   ,   7)  =   6
J (    10   ,   8)  =   36
SYMBOL(     1)  =   6H@@@@@Q
SYMBOL(     2)  =   6HPAYROL
SYMBOL(     3)  =   6H@@@@@C
SYMBOL(     4)  =   6HNAME
SYMBOL(     5)  =   6H@@@@@]
SYMBOL(     6)  =   6HFIRST
SYMBOL(     7)  =   6H@[@G@*
SYMBOL(     8)  =   6HMIDDLE
SYMBOL(     9)  =   6H@[@G@Δ
SYMBOL(    10)  =   6HLAST
SYMBOL(    11)  =   6H@[@G@
SYMBOL(    12)  =   6HSALARY
SYMBOL(    13)  =   6H@@@A@A
SYMBOL(    14)  =   6HREGULR
SYMBOL(    15)  =   6H@@@A@B
SYMBOL(    16)  =   6HOVER
SYMBOL(    17)  =   6H@@@@@C
SYMBOL(    18)  =   6HANS
SYMBOL(    19)  =   6H@[@=@0
SYMBOL(    20)  =   6HDUMMY
SYMBOL(    21)  =   6H@@@@@E
SYMBOL(    22)  =   6H@@@@@@
FILNAM(     1)  =   6H@@@@@Δ
FILNAM(     2)  =   6HPAYROL
FILNAM(     3)  =   6H@@<@@0
FILNAM(     4)  =   6H@@@@@@
GO TO     2
END
```

```
DRUM WRITE CALLED
```

```
@IT FOR,* ABC,ABC
UNIVAC 1108 FORTRAN V LEVEL   22n6 0018 FS01aP
THIS COMPILATION WAS DONE ON 03 JUN 71 AT 14:1n:19

   MAIN PROGRAM

   STORAGE USED (BLOCK, NAME, LENGTH)

         0001    *CODE    000543
         0000    *DATA    000142
         0002    *BLANK   000000
         0003    MATRIX   000623
         0004    STABLE   000334


   EXTERNAL REFERENCES (BLOCK, NAME)

         0005    NTRAN
         0006    TESTRL
         0007    CHECKS
         0010    RUNERR
         0011    SUBSTR
         0012    NROUS
         0013    NIO1S
         0014    NIO2S
         0015    NWOUS
         0016    NSTOPS


   STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001     000277 1L          0001     000210 10L         0001     000200 11L         0n01     n0n211 12L         0001     00n01n 124G
    0001     000275 13L         0001     000016 130G        0001     000030 137G        0001     000257 14L         0001     000136 175G
    0001     000005 2L          0001     000216 220G        0001     000052 3L          0001     000061 4L          0000     000027 5F
    0000     000013 500F        0000     000016 501F        0000     000037 502F        0000     000050 503F        0001     000072 6L
    0001     000130 7L          0001     000120 8L          0001     000152 9L          0000 I   000000 ANS         0004 I   000310 FILNAM
    0000     000061 FIRST       0000 I   000007 I           0003 I   000001 ILIM        0003 I   000000 IND         0000 I   000012 IXGET
    0000 I   000010 IS          0003 I   000003 J           0000 I   000061 LAST        0000     000061 MIDDLE      0000     000061 NAME
    0000 I   000061 OVER        0000 I   000061 PAYROL      0000 I   000061 REGULR      0000     000061 SALARY      0003 I   000002 SP
    0000 I   000011 ST          0004 I   000000 SYMBOL      0006 L   000000 TESTRL



00101      1*           IMPLICIT INTEGER (A-Z)
00103      2*           CALL  NTRAN(     35, 10 )
00104      3*           DIMENSION   PAYROL  (          9)
00105      4*
00105      5*           EQUIVALENCE(NAME  ,      PAYROL)
00106      6*           EQUIVALENCE(FIRST ,      PAYROL)
00107      7*           EQUIVALENCE(MIDDLE,      PAYROL)
00110      8*           EQUIVALENCE(LAST  ,      PAYROL)
00111      9*           EQUIVALENCE(SALARY,      PAYROL)
00112     10*           EQUIVALENCE(REGULR,      PAYROL)
00113     11*           EQUIVALENCE(OVER  ,      PAYROL)
00114     12*           COMMON/MATRIX/IND,ILIM,SP,J(50,8)
00115     13*           COMMON/STABLE/SYMBOL(200),FILNAM(20)
00116     14*           LOGICAL TESTRL
00117     15*           GO TO      1
00120     16*      2    CONTINUE
00121     17*           DIMENSION   ANS      (          7)
00122     18*
00122     19*           ANS (7)='        '
00123     20*           DO 600 I=1,7
00126     21*           READ (5,        500)      PAYROL
00134     22*    500    FORMAT (3(A6,A4), 2I6,A4)
00135     23*           WRITE(6,        501)      PAYROL
00143     24*    501    FORMAT (1X, 'NAME=', 3(A6,A4), ' REG=', I6, ' OVER=', I6)
00144     25*           ST=1
00145     26*           CALL NTRAN (     35,1,          9,      PAYROL,ST)
00146     27*           CALL CHECKS (ST, 5      3)
00147     28*           GO TO      4
00150     29*         3 WRITE (6,         5)
00152     30*         5 FORMAT(1X,11HEND OF FILE)
00153     31*           CALL  RUNERR
00154     32*         4 CONTINUE
00155     33*    600    CONTINUE
00157     34*           CALL  NTRAN(     35, 10)
00160     35*           IXGET=0
00161     36*           ST=1
00162     37*           ANS   (1)=0
00163     38*         6 IXGET=IXGET+1
00164     39*           CALL  NTRAN(     35,2,          9,      PAYROL,ST)
00165     40*           CALL CHECKS (ST, 5      7)
00166     41*           IF (          TESTRL(1,'LAST  ',LAST  ,'LEAR'))GOTO      8
00170     42*           GO TO      6
00171     43*         8 CALL SUBSTR(PAYROL,     9,    1,   36,      ANS   ,    36)
00172     44*         7 CONTINUE
00173     45*           WRITE(6,        502)      ANS
00201     46*    502    FORMAT (1H0,1X, 'NAME WHERE LAST EQ LEAR =', 7A6)
00202     47*           CALL  NTRAN(     35, 10)
00203     48*           IXGET=0
00204     49*           ST=1
00205     50*           ANS   (1)=0
00206     51*         9 IXGET=IXGET+1
00207     52*           CALL  NTRAN(     35,2,          9,      PAYROL,ST)
00210     53*           CALL CHECKS (ST, 5     10)
00211     54*           IF (          TESTRL(3,'REGULR',REGULR,700    ))GOTO     11
00213     55*           GO TO      9
00214     56*        11 CALL SUBSTR(PAYROL,     9,    1,   36,      ANS   ,    36)
00215     57*        10 CONTINUE
00216     58*           WRITE(6,        503)      ANS
00224     59*    503    FORMAT (1H0,1X,'NAME WHERE REG GT 700=', 7A6)
00225     60*           STOP
00226  *DIAGNOSTIC*     CONTROL CAN NEVER REACH THE NEXT STATEMENT
00226     61*           I=1
00227     62*           IXGET=0
00230     63*           ST=1
00231     64*           ANS   (1)=0
00232     65*        12 IXGET=IXGET+1
00233     66*           CALL  NTRAN(     35,2,          9,      PAYROL,ST)
00234     67*           CALL CHECKS (ST, 5     13)
00235     68*           IF (          TESTRL(3,'OVER  ',OVER  ,500    ))GOTO     14
```

```
      GO TO      12
   14 CALL SUBSTR(PAYROL,    9,    1,   36,     ANS   (1,  61,  361
      ANS   (11=  ANS   (11+1
   13 CONTINUE
      STOP
 1    CONTINUE
      ILIH =     10
      J (      1    ,    (1  =    0
      J (      1    ,    21  =    2
      J (      1    ,    31  =    2
      J (      1    ,    41  =    0
      J (      1    ,    51  =    9
      J (      1    ,    61  =    6H@@@@@@
      J (      1    ,    71  =    48
      J (      1    ,    81  =    0
      J (      2    ,    11  =    1
      J (      2    ,    21  =    4
      J (      2    ,    31  =    3
      J (      2    ,    41  =    1
      J (      2    ,    51  =    6
      J (      2    ,    61  =    6H@@@@@@
      J (      2    ,    71  =    36
      J (      2    ,    81  =    0
      J (      3    ,    11  =    2
      J (      3    ,    21  =    6
      J (      3    ,    31  =    0
      J (      3    ,    41  =    2
      J (      3    ,    51  =    4
      J (      3    ,    61  =    6H@@@@@@
      J (      3    ,    71  =    12
      J (      3    ,    81  =    0
      J (      4    ,    11  =    2
      J (      4    ,    21  =    8
      J (      4    ,    31  =    0
      J (      4    ,    41  =    2
      J (      4    ,    51  =    5
      J (      4    ,    61  =    6H@@@@@@
      J (      4    ,    71  =    12
      J (      4    ,    81  =    12
      J (      5    ,    11  =    2
      J (      5    ,    21  =    10
      J (      5    ,    31  =    0
      J (      5    ,    41  =    2
      J (      5    ,    51  =    3
      J (      5    ,    61  =    6H@@@@@@
      J (      5    ,    71  =    12
      J (      5    ,    81  =    24
      J (      6    ,    11  =    1
      J (      6    ,    21  =    12
      J (      6    ,    31  =    2
      J (      6    ,    41  =    1
      J (      6    ,    51  =    2
      J (      6    ,    61  =    6H@@@@@@
      J (      6    ,    71  =    12
      J (      6    ,    81  =    36
      J (      7    ,    11  =    2
      J (      7    ,    21  =    14
      J (      7    ,    31  =    0
      J (      7    ,    41  =    6
      J (      7    ,    51  =    8
      J (      7    ,    61  =    6H@@@@@@
      J (      7    ,    71  =    6
      J (      7    ,    31  =    36
      J (      8    ,    11  =    2
      J (      8    ,    21  =    16
      J (      8    ,    31  =    0
      J (      8    ,    41  =    6
      J (      8    ,    51  =    7
      J (      8    ,    61  =    6H@@@@@@
      J (      8    ,    71  =    6
      J (      8    ,    81  =    42
      J (      9    ,    11  =    0
      J (      9    ,    21  =    18
      J (      9    ,    31  =    0
      J (      9    ,    41  =    0
      J (      9    ,    51  =    10
      J (      9    ,    61  =    6H@@@@@@
      J (      9    ,    71  =    36
      J (      9    ,    81  =    0
      J (     10    ,    11  =    0
      J (     10    ,    21  =    20
      J (     10    ,    31  =    0
      J (     10    ,    41  =    0
      J (     10    ,    51  =    0
      J (     10    ,    61  =    6H@@@@@@
      J (     10    ,    71  =    6
      J (     10    ,    81  =    36
      SYMBOL(        11  =    6H@@@@@Q
      SYMBOL(        21  =    6HPAYROL
      SYMBOL(        31  =    6H@@@@@E
      SYMBOL(        41  =    6HNAME
      SYMBOL(        51  =    6H@@@@@J
      SYMBOL(        61  =    6HFIRST
      SYMBOL(        71  =    6H@E@G@*
      SYMBOL(        81  =    6HMIDDLE
      SYMBOL(        91  =    6H@E@G@A
      SYMBOL(       101  =    6HLAST
      SYMBOL(       111  =    6H@E@G@
      SYMBOL(       121  =    6HSALARY
      SYMBOL(       131  =    6H@@@A@A
      SYMBOL(       141  =    6HREGULR
      SYMBOL(       151  =    6H@@@A@8
      SYMBOL(       161  =    6HOVER
      SYMBOL(       171  =    6H@@@@@C
      SYMBOL(       181  =    6HANS
      SYMBOL(       191  =    6H@E@=@0
      SYMBOL(       201  =    6HDUMMY
      SYMBOL(       211  =    6H@@@@@E
      SYMBOL(       221  =    6H@@@@@@
      FILNAM(        11  =    6H@@@@@A
      FILNAM(        21  =    6HPAYROL
      FILNAM(        31  =    6H@@<@@0
      FILNAM(        41  =    6H@@@@@@
      GO TO      2
      END
```

```
       END OF UNIVAC 1108 FORTRAN V COMPILATION.      1 *DIAGNOSTIC* MESSAGE(S)
PHASE 1 TIME =    1 SEC.
PHASE 2 TIME =    0 SEC.
PHASE 3 TIME =    0 SEC.
PHASE 4 TIME =    0 SEC.
PHASE 5 TIME =    1 SEC.
PHASE 6 TIME =    0 SEC.

TOTAL COMPILATION TIME =   2  SEC




 NAME=PAUL                KRCS       REG=100000 OVER=120000

 NAME=ZBGIN      NEW      PRZYBYLSKI REG=101000 OVER=100000

 NAME=TRUMAN     E.       TURNIPSEED REG= 60000 OVER=  7000

 NAME=PEACE               HAPPINESS  REG=520000 OVER=880000

 NAME=CRYSTAL    SHANDA   LEAR       REG=350000 OVER=      0

 NAME=PRAISE     GOD      BAREBONES  REG=640000 OVER=960000

 NAME=IMA        A.       HOGG       REG=442500 OVER=800000

  NAME WHERE LAST EQ LEAR =CRYSTAL     SHANDA     LEAR

  NAME WHERE REG GT 700=PAUL                     KRCS



@ EOF                                                             03 JUN 71          14:10:23.794
```

FORM NBS-114A (1-71)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. NBS-TN-590	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE A PRELIMINARY DESIGN OF A DATA RETRIEVAL LANGUAGE TO HANDLE A GENERALIZED DATA BASE: DRL	5. Publication Date July 1971 6. Performing Organization Code
7. AUTHOR(S) Elizabeth Fong	8. Performing Organization
9. PERFORMING ORGANIZATION NAME AND ADDRESS NATIONAL BUREAU OF STANDARDS DEPARTMENT OF COMMERCE WASHINGTON, D.C. 20234	10. Project/Task/Work Unit No. Project 6406111 11. Contract/Grant No.
12. Sponsoring Organization Name and Address	13. Type of Report & Period Covered Final 14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

16. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)

DRL (Data Retrieval Language) is a high-level programming language for information retrieval. The language includes a data description language which can descri fixed-length hierarchical data structures, and DRL includes a data retrieval statement whereby a user can retrieve data by specifying conditions on to the data value. DRL also has an environment declaration statement in which the user can indicate specific peripheral devices by unit number for files. The rest of the language consists of an operation repertory of input-output functions and other data manipulations.

DRL is implemented as a preprocessor to FORTRAN V on the UNIVAC 1108. Keywords act as triggers and are replaced by blocks of FORTRAN code.

The purpose of this project is to investigate the design of an information retrieval language to handle a generalized data base. The DRL system consists of a se of primitives utilizing both compile-time macros and run-time subroutines. These primitives are embedded in a high-level procedure-oriented programming language--the "host language"--FORTRAN in this case. These primitives form a base upon which a class of languages can be defined.

17. KEY WORDS (Alphabetical order, separated by semicolons)

Data base; data retrieval; data structure; information storage and retrieval; language extension; preprocessor; programming language.

18. AVAILABILITY STATEMENT [X] UNLIMITED. [ ] FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NTIS.	19. SECURITY CLASS (THIS REPORT) UNCLASSIFIED 20. SECURITY CLASS (THIS PAGE) UNCLASSIFIED	21. NO. OF PAGES 26 22. Price .35

NBS TECHNICAL PUBLICATIONS

PERIODICALS

JOURNAL OF RESEARCH reports National Bureau of Standards research and development in physics, mathematics, chemistry, and engineering. 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.

Published in three sections, available separately:

- **Physics and Chemistry**

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, $9.50; foreign, $11.75*.

- **Mathematical Sciences**

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, $5.00; foreign, $6.25*.

- **Engineering and Instrumentation**

Reporting results of interest chiefly to the engineer and the applied scientist. This section includes many of the new developments in instrumentation resulting from the Bureau's work in physical measurement, data processing, and development of test methods. It will also cover some of the work in acoustics, applied mechanics, building research, and cryogenic engineering. Issued quarterly. Annual subscription: Domestic, $5.00; foreign, $6.25*.

TECHNICAL NEWS BULLETIN

The best single source of information concerning the Bureau's research, developmental, cooperative and publication activities, this monthly publication is designed for the industry-oriented individual whose daily work involves intimate contact with science and technology—*for engineers, chemists, physicists, research managers, product-development managers, and company executives.* Annual subscription: Domestic, $3.00; foreign, $4.00*.

NONPERIODICALS

Applied Mathematics Series. Mathematical tables, manuals, and studies.

Building Science Series. Research results, test methods, and performance criteria of building materials, components, systems, and structures.

Handbooks. Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications. Proceedings of NBS conferences, bibliographies, annual reports, wall charts, pamphlets, etc.

Monographs. Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

National Standard Reference Data Series. NSRDS provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated.

Product Standards. Provide requirements for sizes, types, quality and methods for testing various industrial products. These standards are developed cooperatively with interested Government and industry groups and provide the basis for common understanding of product characteristics for both buyers and sellers. Their use is voluntary.

Technical Notes. This series consists of communications and reports (covering both other agency and NBS-sponsored work) of limited or transitory interest.

Federal Information Processing Standards Publications. This series is the official publication within the Federal Government for information on standards adopted and promulgated under the Public Law 89–306, and Bureau of the Budget Circular A–86 entitled, Standardization of Data Elements and Codes in Data Systems.

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 Special Publication 305, Supplement 1, Publications of the NBS, 1968-1969. When ordering, include Catalog No. C13.10:305. Price $4.50; foreign, $5.75.

* Difference in price is due to extra cost of foreign mailing.

Order NBS publications from: Superintendent of Documents
Government Printing Office
Washington, D.C. 20402

U.S. DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20230

OFFICIAL BUSINESS

PENALTY FOR PRIVATE USE, $300

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF COMMERCE