AIDS FOR COBOL PROGRAM CONVERSION
(FIPS PUB 21 to FIPS PUB 21-1)

CATEGORY: SOFTWARE
468
1975

CATEGORY: PROGRAMMING LANGUAGE
NO. 43
Foreword

The Federal Information Processing Standards Publication Series of the National Bureau of Standards is the official publication relating to standards adopted and promulgated under the provisions of Public Law 89-306 (Brooks Bill) and under Part 6 of Title 15, Code of Federal Regulations. These legislative and executive mandates have given the Secretary of Commerce important responsibilities for improving the utilization and management of computers and automatic data processing systems in the Federal Government. To carry out the Secretary's responsibilities, the NBS, through its Institute for Computer Sciences and Technology, provides leadership, technical guidance, and coordination of government efforts in the development of guidelines and standards in these areas.

The establishment of COBOL as a Federal Standard in March 1972 marked a milestone in the effort to assist the Federal Government ADP user in stating data processing applications in such a way that the programs and data can be developed and maintained with a minimum of time and effort. In recognition of the need to keep the COBOL standard responsive to the requirements of Federal users, FIPS Task Group 9 was established and was charged with making recommendations to NBS for updating and revising Federal Standard COBOL. The Task Group pursued their charge with full recognition of the activities of American National Standards Institute Committee X3J4 and of CODASYL Programming Languages Committee.

The work of X3J4 resulted in revised American National Standard COBOL, X3.23-1974. FIPS Task Group 9 recommended to NBS that the revised National Standard be adopted as revised Federal Standard COBOL.

Task Group 9, recognizing the need to inform COBOL users of the differences between original Federal Standard COBOL and the revised Federal Standard, has compiled this publication which is a handy reference to all the changes made to Federal Standard COBOL. The National Bureau of Standards is pleased to have the opportunity to make this reference material available for use by Federal agencies.

R. M. Davis, Director
Institute for Computer Sciences and Technology

Abstract

Since COBOL is a "living" language, in the sense that it is under constant development and clarification, the Federal community which relies heavily on COBOL to satisfy programming needs has a large degree of assurance that COBOL will continue to meet their needs as future generation systems are introduced. However, along with the advantage of having more sophisticated and better COBOL tools to meet new systems requirements, there is a short term disadvantage. As clarifications and new facilities are added, they interact with the language specifications already standardized, and this interaction sometimes requires changes in source programs. An analysis, in the form of narrative descriptions and syntax comparisons, is provided to aid the transition of COBOL programs from use with compilers developed in accordance with the 1968 COBOL Standard (FIPS PUB 21) to compilers developed in accordance with the 1974 COBOL Standard (FIPS PUB 21-1).

Key Words: COBOL; COBOL program conversion; Federal Standard COBOL; program conversion; programming aids; programming languages.

Federal Information Processing Standards Publication 43

1975 December 1

ANNOUNCING THE

AIDS FOR COBOL PROGRAM CONVERSION
(FIPS PUB 21 to FIPS PUB 21-1)


Name of Publication. Aids for COBOL Program Conversion (FIPS PUB 21 to FIPS PUB 21-1).

Category. Software, Programming Language.

Explanation. The purpose of this publication is to provide aid to the Federal community in taking advantage of the new facilities incorporated into the revised Federal COBOL Standard (FIPS PUB 21-1) while insuring, with a minimum of effort, that current programs continue to execute in a predictable manner.


Cross Index. FIPS PUB 21–1, COBOL.

Where To Obtain Copies of this Publication.

a. Copies of this publication are available from the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402 (SD Catalog Number C13.52:43). There is a 25 percent discount on quantities of 100 or more. When ordering, specify document number, title, and SD Catalog Number. Payment may be made by check, money order, coupons, or deposit account.

b. Microfiche of this publication is available from the National Technical Information Service, U.S. Department of Commerce, Springfield, Virginia 22151. When ordering refer to Report Number NBS-FIPS-PUB-43 and title. Payment may be made by check, money order, or deposit account.
I. Background

FIPS PUB 21, dated 1972 March 15, established the American National Standard COBOL (X3.23-1968), minus the Report Writer module, as Federal Standard COBOL. The prime objective of this action was to promote a high degree of interchangeability of COBOL programs for use on a variety of automatic data processing systems.

Because of the dynamic nature of the development and standardization activities for COBOL at the National level, the COBOL specifications adopted by FIPS PUB 21 have now been expanded to include a number of new facilities. In addition, a number of clarifications were approved and incorporated in the language specification which remove ambiguities in the otherwise unchanged facilities previously incorporated in FIPS PUB 21.

In recognition of the needs of Federal Government agencies to have these new COBOL facilities and clarifications available for their use, the FIPS Coordinating and Advisory Committee, with the approval of the National Bureau of Standards, established FIPS Task Group 9–Federal Standard COBOL, in 1971. The initial charter of FIPS Task Group 9 was to develop and recommend a revised Federal COBOL Standard to replace the 1968 specifications adopted by FIPS PUB 21.

During this same time period, the Conference on Data Systems Languages (CODASYL) and the American National Standards Institute (ANSI) were very active in the development and standardization, respectively, of facilities for inclusion in a revised National COBOL Standard. In order to preclude a unilateral, overlapping effort at the Federal level, FIPS Task Group 9 pursued their activities in full recognition of the National efforts and, as a result, were able to recommend a revised Federal COBOL Standard which complements the revised American National Standard (X3.23-1974).

Since COBOL is a “living” language, in the sense that it is under constant development and clarification, the Federal community which relies heavily on COBOL to satisfy their programming needs has a large degree of assurance that COBOL will continue to meet their data processing needs as future generation systems are introduced. However, along with the advantage of having more sophisticated and better COBOL tools to interact with new systems requirements, there is a short term disadvantage. As new facilities are added, these new facilities must, in most cases, interact with the language specification already standardized, and this interaction sometimes produces changes in the behavior of language features currently in use. In other cases, implementor-defined actions are deleted in favor of a “standard” action specified by the language itself.

The purpose of this publication is to aid the Federal community to take advantage of the new facilities incorporated into the revised Federal COBOL Standard (FIPS PUB 21-1) while insuring, with a minimum of effort, that current programs continue to execute in a predictable manner.

II. Introduction

This publication provides two types of information to aid in the transitioning of COBOL programs from use with compilers developed in accordance with the 1968 COBOL Standard (FIPS PUB 21) to compilers developed in accordance with the 1974 COBOL Standard (FIPS PUB 21-1).

A detailed listing of language additions, deletions, and changes is provided in narrative form. This listing is in sequence by Nucleus and Functional Processing Module, as specified in FIPS PUB 21-1, and is intended primarily for use by the manager and systems analyst who needs a detailed overview of language changes.
In addition, a composite language skeleton is provided, which presents a side-by-side comparison of the COBOL General Formats in the 1968 and 1974 COBOL Standards. This composite is ordered by COBOL Division, Section and Paragraph. It is intended primarily for use by the programmer who is responsible for making program changes.

These two aids are presented in a manner which allows them to be used independently of each other. However, the aids may be used in conjunction with each other to effect a complete analysis not only of what features have been added or deleted, but also what specific syntactic changes must be made to the program to make it acceptable for compilation by a compiler conforming to the 1974 COBOL Standard.

III. Description of Narrative Analysis

The narrative analysis of language changes (Appendix A) is presented in fourteen sections. These sections include the Nucleus, the Functional Processing Modules contained in the 1968 and 1974 COBOL Standards, and the Reserved Word List. The Functional Processing Modules are presented in the order in which they appear in the Standards.

Each section of the narrative contains the changes which apply to the particular functions covered by that Section. These changes are categorized according to the following criteria:

Additions—All new language capabilities in the 1974 COBOL Standard are listed.

Deletions—All language facilities included in the 1968 COBOL Standard which were deleted from the 1974 COBOL Standard are listed.

Changes (Not Requiring Program Modification)—The language elements listed under this heading fall into one of two categories:

1. The language element has had additional syntax and/or semantics applied which did not change the result of execution of the facility as defined in the 1968 COBOL Standard; or

2. Language restrictions included in the 1968 COBOL Standard have been relaxed or deleted which will have no effect on the programs written to conform with these previous restrictions.

Changes (Requiring Program Modification)—The language facilities listed under this heading fall into one of two categories:

1. The language facility has been revised syntactically so that the function will require some degree of recoding before it is acceptable to a compiler conforming to the 1974 COBOL Standard; or

2. The behavior of the facility has been revised so that the use of the function with compilers conforming to the 1968 COBOL Standard will not provide the same results, without a modification to the program, when used with compilers conforming to the 1974 COBOL Standard.

Other Changes—This category includes a listing of all of the elements which were previously either undefined, implementor-defined, or the specifications were ambiguous, allowing implementation of the facility to be interpreted in different ways by different manufacturers. Program modification in this area will depend on whether the compiler being used interprets the function in the same manner as the new language standard. Therefore, each of these changes must be examined individually in light of the behavior of the function in the compiler currently being used.

The Functional Processing Modules which represent new facilities not included in the 1968 COBOL Standard are so annotated and a list of major functions is provided in lieu of the categories discussed above.
Within each category (additions, deletions, etc.), the language elements are listed in the order in which they are presented in the 1974 COBOL Standard. The titles which are given for grouping the language elements are the same as the title of the element in the index of the 1974 COBOL Standard, thus providing a cross reference for the reader who needs more detailed information on that particular topic.

In the 1974 COBOL Standard, the Nucleus and the Functional Processing Modules are each divided into one or two non-null levels. The lowest non-null level supplies the elements necessary to perform basic operations; the second level provides more extensive and sophisticated processing capabilities. In all cases, the low level is a proper subset of the high level. To accommodate this concept, each element listed in each category is coded at the right-hand side of the page with the level of the Nucleus or Functional Processing Module in which the element appears. In this way, the reader can differentiate between the changes which affect all programs using only the basic COBOL facilities and those changes which affect only programs written using the more extensive and sophisticated facilities in the 1968 COBOL Standard. This code can also be used as a cross reference for the reader to the specific portion of the 1974 COBOL Standard where more detailed information is available.

Example of use of the narrative:

<table>
<thead>
<tr>
<th>Text</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE HANDLING</td>
<td>—(Functional Processing Module name)</td>
</tr>
<tr>
<td>Changes (Requiring Program Modification)</td>
<td>—(Category)</td>
</tr>
<tr>
<td>The SEARCH Statement</td>
<td>—(Title cross-referencing the index in the Federal COBOL Standard)</td>
</tr>
<tr>
<td></td>
<td>—(Specific change)</td>
</tr>
<tr>
<td></td>
<td>—(Indicates the level of the module to which the change relates)</td>
</tr>
</tbody>
</table>

The object of the condition in the... 2TBL

Finally, a Reserved Word List is included which details the COBOL Reserved Words incorporated in the 1968 COBOL Standard which have been deleted, and the Reserved Words which were added to the 1974 COBOL Standard. If any of the Reserved Words added in the 1974 COBOL Standard appear as a user-defined word in programs conforming to the 1968 COBOL Standard, program modification will be required to substitute new user-defined words.

**IV. Description of Composite Language Skeleton**

The Composite Language Skeleton compares the complete syntactical formats for the 1968 COBOL Standard (FIPS PUB 21) with the 1974 COBOL Standard (FIPS PUB 21-1). The Skeleton is ordered by COBOL Division, Section and Paragraph for the Identification, Environment and Data Divisions. The Procedure Division statements are listed in alphabetical order.

The left-hand column of the Composite Language Skeleton contains all of the syntactic formats for the 1968 COBOL Standard. The information contained in the right-hand column differs depending on the following considerations:

1. If a language element in the 1974 COBOL Standard is the same syntactically as it was for the language element in the 1968 COBOL Standard, the area in the right-hand column horizontal with the language element description is blank.
(2) If the syntax for a language element in the 1968 COBOL Standard has been revised in the 1974 COBOL Standard, the changed syntax appears in the right-hand column directly across from the corresponding syntax as it appeared in the 1968 COBOL Standard.

(3) If a language element in the 1968 COBOL Standard has been deleted from the 1974 COBOL Standard, a comment to that effect appears in the right-hand column.

(4) If a language element has been added to the 1974 COBOL Standard, the complete syntactic format for new language element appears in the right-hand column. The left-hand column is blank.

When converting programs, without Report Writer statements, which conform to the 1968 COBOL Standard to the 1974 COBOL Standard, coding revisions must be made only for those language elements which have been modified or deleted in the 1974 COBOL Standard. This information can be readily detected in the Skeleton by the existence of corresponding entries in both columns.

When converting programs containing Report Writer statements reflected in the 1968 COBOL Standard, extra care must be taken since no syntax changes have been made in some cases where the semantics have been completely changed for the 1974 COBOL Standard.

The notation used in the Composite Language Skeleton is the same as that used in American National Standard COBOL, X3.23-1974 and American Standard COBOL, X3.23-1968. Detailed explanations for the notation may be found in those documents in the section “Overall Language Considerations.” Braces, {}, enclosing a portion of a general format indicate a choice of one of the options within the braces must be made. Brackets, [ ], enclosing a portion of a general format indicates that portion of a general format may be included or omitted at the user's option. All underlined uppercase words in a clause or phrase are required, but uppercase words which are not underlined are optional. Lowercase words are generic terms used to represent COBOL words, literals, PICTURE characterstrings, comment entries, or a complete syntactical entry that must be supplied by the user. The ellipsis ( . . . ) represents the repetition of the previous portion of a format.

Acknowledgment

The Institute for Computer Sciences and Technology acknowledges William C. Rinehuls, United States Air Force, and Margaret M. Cook, United States Navy, as the principal authors of this publication.

The Institute wishes to thank each member of FIPS Task Group 9 for their suggestions regarding the organization and presentation of the material presented.

The Institute also recognizes Mabel V. Vickers, Systems and Software Division of the Institute, for reviewing the entire document, effecting much contextual revision, and editing the manuscript for publication.
APPENDIX A

NARRATIVE ANALYSIS
Section I—Nucleus

The Nucleus module consists of two levels: 1NUC (low level) and 2NUC (high level).

ADDITIONS

Characters Used in Editing

The stroke character (/) is permitted as an editing character.

Literals

Two quotation mark characters (""") can be included in a nonnumeric literal to represent a single quotation mark character.

Comment Lines

Comment lines can appear any place within a program by specifying an asterisk (*) in character position 7 (Indicator Area).

Comment lines with page ejection can appear any place within a program by specifying a stroke (/) in character position 7 (Indicator Area).

Switches

Switches, which are implementor-defined, may be either hardware or software switches. Condition-name may be given the status of an implementor-defined switch.

Collating Sequence

The PROGRAM COLLATING SEQUENCE clause of the OBJECT-COMPUTER paragraph specifies the collating sequence to apply to the program if other than the native collating sequence of the computer is desired.

The alphabet-name clause in the SPECIAL-NAMES paragraph provides the mechanism for relating a name to a specific character code set and/or collating sequence.

The PICTURE Clause

Alphabetic PICTURE character-strings may contain the character ‘B’.

The REDEFINES Clause

The object of a REDEFINES clause can be subordinate to an item described with the OCCURS clause.

The SIGN Clause

The SIGN clause specifies the position and the mode of representation of the operational sign when it is necessary to describe these properties explicitly.

Procedure Division

In the Procedure Division, a section may contain zero or more paragraphs and a paragraph may contain zero or more sentences.

Arithmetic Statements

The GIVING identifier series clause of the arithmetic statements (ADD, SUBTRACT, MULTIPLY and DIVIDE) allows storage of the results of the arithmetic computation in more than one separate area.

* The capability to specify a user-defined collating sequence (literal phrase of the alphabet-name clause) was added to 2NUC only.
Section I—Nucleus (continued)

**ADDITIONS (continued)**

- The unary + is permitted in arithmetic expressions.
- The identifier series of the COMPUTE statement allows more than one data item to be assigned an arithmetic value.
- The INTO identifier series clause of the DIVIDE statement allows division of more than one set of values.
- The remainder item in a DIVIDE statement can be numeric edited.
- The BY identifier series clause of the MULTIPLY statement allows multiplication by more than one set of values.
- The ACCEPT Statement
  - The DAY, DATE and TIME clauses were added to the ACCEPT statement. This allows the programmer access to the internally stored day, date or time.
- The INSPECT Statement
  - The INSPECT statement tallies or replaces occurrences of a single character or group of characters in a data item. This replaces the EXAMINE statement.
- The MOVE Statement
  - A scaled integer item may be moved to an alphanumeric or alphanumeric-edited item.
- The PERFORM Statement
  - In the Format 4 PERFORM statement, identifier need not be described as an integer in the VARYING clause.
- The STRING Statement
  - The STRING statement provides juxtaposition of the partial or complete contents of two or more data items into a single data item.
- The UNSTRING Statement
  - The UNSTRING statement causes contiguous data in a sending field to be separated and placed into multiple receiving fields.

**DELETIONS**

- The Identification Division
  - The REMARKS paragraph in the Identification Division was deleted. It was replaced by an asterisk (*) or stroke (/) in character position 7 (Indicator Area).
- The EXAMINE Statement
  - The EXAMINE statement was deleted. It was replaced by the INSPECT statement.
- The NOTE Statement
  - The NOTE statement was deleted. It was replaced by an asterisk (*) or stroke (/) in character position 7 (Indicator Area).
Section I—Nucleus (continued)

CHANGES (REQUIRING PROGRAM MODIFICATION)

Continuation of Lines

Continuation of Identification Division comment-entries must not have a hyphen in character position 7 (Indicator Area).

Qualification

Complete set of qualifiers for a name may not be the same as the partial list of qualifiers for another name.

The CURRENCY SIGN Clause

The characters ‘L’, ‘/’, and ‘=’ may not be used in the CURRENCY SIGN clause of the SPECIAL-NAMES paragraph.

Data Division

All items which are immediately subordinate to a group level must have the same level-number.

No entry with a lower level-number can appear between the redefined and the redefining items.

CHANGES (REQUIRING PROGRAM MODIFICATION) (continued)

The PICTURE Clause

An asterisk used as a zero suppression symbol in a PICTURE clause and the BLANK WHEN ZERO clause may not appear in the same entry.

The VALUE Clause

A signed numeric literal cannot be used in a VALUE clause unless it is associated with a signed PICTURE character-string.

If an item is numeric-edited, the literal in the VALUE clause must be nonnumeric.

Conditions

In relation and sign conditions, arithmetic-expression must contain at least one reference to a variable.

The DISPLAY Statement

In the DISPLAY statement, numeric literals must be unsigned integers.

The STOP Statement

If the operand in the STOP statement is a numeric literal, it must be an unsigned integer.

CHANGES (NOT REQUIRING PROGRAM MODIFICATION)

Punctuation

Punctuation rules have been relaxed to allow space preceding a comma, period and semicolon, and preceding or following a left parenthesis (except in a PICTURE character-string).

Semicolon and comma are interchangeable.
Section I—Nucleus (continued)

**CHANGES (NOT REQUIRING PROGRAM MODIFICATION) (continued)**

Reference Format

Level-numbers 02 through 49 entries may appear anywhere to the right of Margin A.

Conditions

The word ‘TO’ is not required in the relation condition ‘EQUAL TO’.

Working-Storage Section

Level 77 items need not precede level 01 entries.

The GO TO Statement

The word ‘TO’ is not required in the GO TO statement.

The PERFORM Statement

There is no logical difference between fixed and fixed overlayable segments in the PERFORM statement.

**OTHER CHANGES** (See Paragraph III above)

Mnemonic-Name

Mnemonic-name must have at least one alphabetic character.

Qualification

Number of qualifiers permitted must be at least five.

The PICTURE Clause

PICTURE character-string is limited to 30 characters.

The number of digit positions that can be described by a numeric PICTURE character-string cannot exceed 18.

Conditions

Use of NOT and the left and right parentheses in abbreviated combined conditions was clarified so that all subjects and operators required for expansion of any portion of an abbreviated combined condition must be included in the same set of parentheses.

The numeric test in class condition cannot be used with a group item composed of elementary items described as signed.

Comparisons

In the comparison of a numeric operand to a nonnumeric operand, the numeric operand is treated as though it were moved to an alphanumeric item of the same size, and the contents of this alphanumeric item were then compared to the nonnumeric operand.

Arithmetic Statements

The composite of operands requirements has been added to the MULTIPLY and DIVIDE statements.
Section I—Nucleus (continued)

CHANGES (NOT REQUIRING PROGRAM MODIFICATION) (continued)

The PERFORM Statement

Changing the variable in the FROM phrase of the Format 4 PERFORM statement during execution can affect the number of times the procedures are executed if more than one AFTER phrase is specified.

A PERFORM statement in a non-independent segment may have only non-independent segments or may have only sections wholly contained in a single independent segment within its range.

The PERFORM Statement (continued)

A PERFORM statement in an independent segment may have only non-independent segments or may have only sections wholly contained within the same independent segment within its range.

Control is passed only once for each execution of a Format 2 PERFORM statement.

Section II—Table Handling

The Table Handling module is divided into two levels: 1TBL (low level) and 2TBL (high level). 1TBL contains the elements which appear in levels 1 and 2 of the 1968 COBOL Standard; 2TBL contains the elements which appeared in level 3 of the 1968 COBOL Standard.

ADDITIONS

Literals and index-names may be mixed in a table reference.

An index may be set up or down by a negative value.

DELETIONS

None.

CHANGES (NOT REQUIRING PROGRAM MODIFICATION)

Punctuation

Period, comma or semicolon may be preceded by a space.

Parenthesis enclosing subscripts need not be preceded by a space.

The SET Statement

Integer may be negative in Format 2 of the SET statement.

CHANGES (REQUIRING PROGRAM MODIFICATION)

The OCCURS Clause

OCCURS DEPENDING ON clause may only be followed within the record description by data description entries subordinate to it.

The DEPENDING ON clause is now required in a Format 2 OCCURS clause.

Integer-1 cannot be zero in a Format 2 OCCURS clause.

When a group item, having subordinate to it an entry that specifies Format 2 of the OCCURS clause, is referenced, only that part of the table area which is defined by the value of the operand of the DEPENDING ON phrase will be used in the operation. (The actual size of the variable-length item is used.)
Section II—Table Handling (continued)

CHANGES (REQUIRING PROGRAM MODIFICATIONS) (continued)

The SEARCH Statement

The object of the condition in the WHEN phrase of the SEARCH ALL statement cannot be a data item named in the KEY phrase of the referenced table.

OTHER CHANGES (See Paragraph III above)

The OCCURS Clause

If the SYNCHRONIZED clause is specified for an item containing the OCCURS clause, any implied FILLER generated for items in the table are generated for each occurrence of those items.

The SEARCH Statement

The results of a SEARCH ALL operation are predictable only when the data in the table is ordered as described by the ASCENDING/DESCENDING KEY clause associated with identifier-1.

If identifier-2 of the VARYING clause of the SEARCH statement is an index data item, it is incremented as the associated index is incremented.

Section III—Sequential I-O

The Sequential I-O module consists of two levels: 1SEQ (low level) and 2SEQ (high level).

ADDITIONS

The File-Control Paragraph/Entry

The FILE STATUS clause of the SELECT clause of the File-Control entry specifies the data-name which indicates the status of execution of a statement that references the file.

The ORGANIZATION IS SEQUENTIAL clause of the SELECT clause of the File-Control entry specifies the logical structure of a file.

The RESERVE integer AREAS clause of the SELECT clause of the File-Control entry specifies the number of input-output areas allocated for the file records.

The File Description Entry

The CODE-SET clause of the File Description entry specifies the character code set used to represent data on the external media.

The LINAGE clause of the File Description entry provides a means for specifying the depth of a logical page and the size of the top and bottom margins on the logical page.

The CLOSE Statement

The FOR REMOVAL clause of the CLOSE statement specifies that a reel/unit is logically removed from the run unit.
Section III—Sequential I-O (continued)

**ADDITIONS** (continued)

The OPEN Statement

The EXTEND clause of the OPEN statement positions the file immediately following the last logical record previously written on that file (so that additional records can be placed on the file).

The REWRITE Statement

The REWRITE statement logically replaces a record existing in a mass storage file.

The WRITE Statement

The PAGE clause of the BEFORE/AFTER ADVANCING clause of the WRITE statement controls presentation of a line before or after advancing to the next logical page.

The END-OF-PAGE clause of the WRITE statement provides programmer control when the end of a logical page is reached.

The USE Statement

An EXCEPTION clause was added to the USE AFTER STANDARD PROCEDURE statement. This function includes invocation of the associated procedure for both error (e.g., INVALID KEY) or exception (e.g., AT END) conditions.

**DELETIONS**

User-Defined Labels

The facility to define, initialize and access user-defined labels was deleted. The deletion includes data-names option of LABEL RECORDS clause and the USE statement option for label processing.

The File-Control Entry

The FILE-LIMITS clause in the File-Control entry was deleted.

The integer phrase of the ASSIGN clause in the File-Control entry was deleted.

The MULTIPLE REEL/UNIT clause in the File-Control entry was deleted.

The WRITE Statement

The INVALID KEY clause of the WRITE statement was deleted. This function is now handled by the FILE-STATUS clause.

**CHANGES (NOT REQUIRING PROGRAM MODIFICATION)**

The CLOSE Statement

The CLOSE statement with NO REWIND clause applies to all devices for which support is claimed.

The OPEN Statement

The OPEN statement with the INPUT clause and the OPEN statement with the I-O clause makes the record area available to the programmer.
Section III—Sequential I-O (continued)

CHANGES (NOT REQUIRING PROGRAM MODIFICATION) (continued)

The OPEN statement with NO REWIND clause applies to all devices for which support is claimed.

The OPEN statement with the REVERSED clause applies to all devices for which support is claimed.

The READ Statement

The AT END phrase of the READ statement need not be specified if an applicable USE FOR ERROR/EXCEPTION procedure is specified.

CHANGES (REQUIRING PROGRAM MODIFICATION)

The VALUE OF Clause

The data-name phrase in the VALUE OF clause was changed to an implementor-name phrase.

The CLOSE Statement

The ability to use the CLOSE statement with the REEL or UNIT phrase together with the LOCK phrase was deleted.

The OPEN Statement

The REVERSED clause of the OPEN statement now automatically positions the file at its end without programmer intervention.

The USE Statement

The recursive invocation of USE procedures is prohibited.

OTHER CHANGES (See Paragraph III above)

File Description Entry

The File Description entry for file-name must be equal to that used when the file was created.

Section IV—Random Access

The Random Access module was deleted.

It is replaced by two new modules—Relative I-O and Indexed I-O. There is a functional and syntactic similarity between the new Relative I-O module and the previous Random Access module; however, the Indexed I-O module has no functional equivalent in the previous Federal COBOL Standard.

The replacement deleted or changed a number of facilities previously included in the Random Access module. The changed facilities are included in the discussion of Relative I-O and Indexed I-O. The deleted facilities are:

User-Defined Labels

The facility to define, initialize and access user-defined labels was deleted. (Data-names option of LABEL RECORDS clause and the USE statement option for label processing.)
Section IV—Random Access (continued)

The File-Control Entry

The ACTUAL KEY clause of the File-Control entry was deleted.

The PROCESSING MODE clause of the File-Control entry was deleted.

The FILE-LIMITS clause of the File-Control entry was deleted.

The SEEK Statement

The SEEK statement was deleted.

Section V—Relative I-O

This is a new module consisting of three levels, one of which is null. The two non-null levels are: 1REL (low level) and 2REL (high level). The Relative I-O module includes the following major features:

The File-Control Paragraph/Entry

An ACCESS MODE clause specifies random, sequential or dynamic (both random and sequential) access. 1REL* 2REL*

The File-Control paragraph names the file and specifies other file-related information.

The RELATIVE KEY clause of the ACCESS MODE clause of the SELECT clause of the File-Control entry specifies the key for a retrieval generated by the START statement.

The FILE-STATUS clause of the SELECT clause of the File-Control entry indicates the status of execution of a statement that references the file.

The ORGANIZATION IS RELATIVE clause of the select clause of the File-Control entry specifies the logical structure of a file.

The RESERVE integer AREAS clause of the SELECT clause of the File-Control entry specifies the number of input-output areas allocated for the file records.

The DELETE Statement

The DELETE statement logically removes a record from a mass storage file. 1REL

The READ Statement

The NEXT clause of the READ statement specifies that the next logical record relative to the last one retrieved is to be retrieved. 2REL

The REWRITE Statement

The REWRITE statement logically replaces a record existing in a mass storage file. 1REL

The START Statement

The START statement logically positions the Current Record Pointer within a relative file for subsequent sequential retrieval of records. 2REL

* Dynamic access is in 2REL only; Random and Sequential access are in 1REL and 2REL.
Section V—Relative I-O (continued)

The USE Statement

The USE AFTER STANDARD ERROR/EXCEPTION statement provides procedures for error or exception handling.

In addition to the major features detailed above, a number of changes have been made to features which originally were included in the Random Access module. The major changes include:

File Description Entry

A File Description entry for file-name must be equal to that used when the file was created.

The OPEN Statement

The OPEN FOR INPUT statement and the OPEN FOR I-O statement make the record area available to the programmer.

The READ Statement

The AT END phrase and the INVALID KEY phrase of the READ statement need not be specified if an applicable USE AFTER ERROR/EXCEPTION procedure is specified.

The WRITE Statement

The INVALID KEY clause of the WRITE statement need not be specified if an applicable USE AFTER STANDARD ERROR/EXCEPTION statement is specified.

The USE Statement

Recursive invocation of USE procedures is prohibited.

Section VI—Indexed I-O

This is a new module consisting of three levels, one of which is null. The two non-null levels are 1INX (low level) and 2INX (high level). The Indexed I-O module includes the following major features:

The File-Control Paragraph/Entry

The File-Control paragraph names the file and specifies other file-related information.

The ACCESS MODE clause specifies random, sequential or dynamic (both random and sequential) access.

The ALTERNATE RECORD KEY clause of the SELECT clause of the File-Control entry provides an alternate access path to records in the file.

The FILE-STATUS clause of the SELECT clause of the File-Control entry specifies the data-name which indicates the status of execution of a statement that references the file.

* Dynamic access is in 2INX only; Random and Sequential access are in 1INX.
Section VI—Indexed I-O (continued)

The ORGANIZATION IS Indexed clause of the SELECT clause of the File-Control entry specifies the logical structure of a file.

The RECORD KEY clause of the SELECT clause of the File-Control entry specifies the prime record key for the file.

The RESERVE integer AREAS clause of the SELECT clause File-Control entry specifies the number of input-output areas allocated for the file records.

The DELETE Statement

The DELETE statement logically removes a record from a mass storage file.

The READ Statement

The KEY IS clause of the READ statement establishes the key for a retrieval from the file if the key is different from the prime record key.

The START Statement

The START statement logically positions the Current Record Pointer within an indexed file for subsequent sequential retrieval of records.

The REWRITE Statement

The REWRITE statement logically replaces a record in a mass storage file.

The USE Statement

The USE AFTER STANDARD ERROR/EXCEPTION statement provides procedures for error or exception handling.

In addition to the major features detailed above, a number of changes have been made to features which originally were included in the Random Access module which affect the Indexed I-O module. These major changes include:

File Description Entry

A File Description entry for file-name must be equal to that used when the file was created.

The OPEN Statement

The OPEN INPUT statement and the OPEN I-O statement make the record area available to the programmer.

The READ Statement

The AT END phrase and the INVALID KEY phrase of the READ statement need not be specified if an applicable USE AFTER ERROR/EXCEPTION procedure is specified.

The WRITE Statement

The INVALID KEY clause of the WRITE statement need not be specified if an applicable USE AFTER STANDARD ERROR/EXCEPTION statement is specified.

The USE Statement

Recursive invocation of USE procedures is prohibited.
Section VII—Sort-Merge

This module replaces the Sort module contained in the previous Federal COBOL Standard. It contains three levels, one of which is null. The two non-null levels are: 1SRT (low level) and 2SRT (high level).

**ADDITIONS**

The I-O-CONTROL Paragraph

The SORT-MERGE option was added to the SAME AREA clause.

The MERGE Statement

The MERGE statement combines two or more identically sequenced files.

The SORT Statement

The COLLATING SEQUENCE clause of the SORT statement establishes the collating sequence for the sort if it is different from the program collating sequence.

The USING file-name series of the SORT statement specifies the files to be sorted. Formerly only one file was named in a USING clause.

**DELETIONS**

The OR implementor-name option of the ASSIGN TO clause of the SELECT clause of the FILE-CONTROL paragraph was deleted.

**CHANGES (NOT REQUIRING PROGRAM MODIFICATION)**

None.

**CHANGES (REQUIRING PROGRAM MODIFICATION)**

The SORT Statement

Semicolon was deleted from the SORT statement format.

No more than one file-name from a multiple-file reel can appear in a SORT statement.

When a SORT (or MERGE) statement appears in a non-independent segment, any associated input-output procedures must appear only in non-independent segments or must appear only in sections wholly contained in a single independent segment.

When a SORT (or MERGE) statement appears in an independent segment any associated input-output procedures must appear only in a non-independent segment or must appear only in sections wholly contained in the same independent segment.

**OTHER CHANGES (See Paragraph III above)**

None.
Section VIII—Report Writer

The Report Writer specifications are an optional appendage to the Federal levels to be used by agencies when the acquisition of a COBOL Report Writer facility is required. The Report Writer specifications contained in the 1974 Standard are a complete revision and rewrite of the Report Writer specifications excluded from the previous Federal Standard. The two versions of the Report Writer specifications are not compatible.

The Report Writer module contains two levels, one of which is null.

Following is a list of the major report writer features:

A Report Section in the Data Division, consisting of one or more Report Description (RD) entries.
A COLUMN NUMBER clause for horizontal spacing.
A LINE NUMBER clause for vertical spacing.
A NEXT GROUP clause for spacing after the last line of a group.
A PAGE clause to specify the length of the page, heading and footing areas, and the area on which the detail line appears.
A SOURCE clause for sending data items to the printable items.
A SUM clause for establishment of a sum counter.
A VALUE clause for defining a literal that appears in the printable item.
The GENERATE statement for formatting detail report groups.
The INITIATE statement to initialize functions.
The SUPPRESS statement to suppress printing of a report group.
The TERMINATE statement to terminate a report.
The USE BEFORE REPORTING statement.

Section IX—Segmentation

This module contains three levels, one of which is null. The non-null levels are: 1SEG (low level) and 2SEG (high level).

ADDITIONS

None.

DELETIONS

None.

CHANGES (NOT REQUIRING PROGRAM MODIFICATION)

Segment-Numbers

Sections in declaratives may have segment-numbers.

All restrictions were deleted on the range of a PERFORM statement involving fixed overlayable segments.
Section IX—Segmentation (continued)

CHANGES (REQUIRING PROGRAM MODIFICATIONS)

The SORT and MERGE Statements

The input-output procedure associated with a SORT or MERGE statement which appear in a non-independent segment may have only non-independent segments or may have only sections wholly contained in a single independent segment within its range.

The input-output procedure associated with a SORT or MERGE statement which appears in an independent segment may have only non-independent segments or may have only sections wholly contained in the same independent segment within its range.

OTHER CHANGES (See Paragraph III above)

The PERFORM Statement

Control is passed only once for each execution of a Format 2 PERFORM statement.

PERFORM statements which appear in a non-independent segment may have only non-independent segments or may have only sections wholly contained in a single independent segment within its range.

PERFORM statements which appear in an independent segment may have only non-independent segments or may have only sections wholly contained in the same independent segment within its range.

Section X—Library

This module contains three levels, one of which is null. The non-null levels are: 1LIB (low level) and 2LIB (high level).

ADDITIONS

More than one library can be available.

All occurrences of defined groups of words in the library text can be replaced.

DELETIONS

None.

CHANGES (NOT REQUIRING PROGRAM MODIFICATION)

A COPY statement may appear any place in the program that a COBOL word or separator may appear.

CHANGES (REQUIRING PROGRAM MODIFICATION)

None.

OTHER CHANGES (See Paragraph III above)

Library-name is now a user-defined word.

The effect of processing a COPY statement is that the library text is copied logically, replacing the entire COPY statement, beginning with the reserved word COPY and ending with the period.
Section XI—Debug

This is a new module containing three levels, one of which is null. The non-null levels are: 1DEB (low level) and 2DEB (high level).

The Debug module contains the following major features:

A USE FOR DEBUGGING statement identifies the items to be monitored.

The Special Register DEBUG-ITEM provides information about the conditions that caused the execution of a debugging section.

The WITH DEBUGGING MODE clause of the SOURCE-COMPUTER paragraph serves as a compile-time switch over the debugging statements in a program.

Debugging lines 'D' in character position 7 (Indicator Area), identifies a line of coding which applies only to debugging.

Section XII—Inter-Program Communication

This is a new module containing three levels, one of which is null. The non-null levels are 1IPC (low level) and 2IPC (high level).

The Inter-Program Communication module consists of the following major features:

A USING phrase in the Procedure Division header associates the data items in the called program with the data items in the calling program.

The LINKAGE SECTION describes data items common to the called and calling programs.

The CALL statement causes control to be transferred from one object program to another.

The CANCEL statement releases the memory areas occupied by the called program.

The EXIT PROGRAM statement marks the logical end of a called program causing control to be returned to the calling program.

Section XIII—Communication

This is a new module containing three levels, one of which is null. The non-null levels are: 1COM (low level) and 2COM (high level).

The Communication module consists of the following major features:

The Communication Section with a Communication Description (CD) entry specifies the interface area between the message control system and a COBOL program.

The ACCEPT MESSAGE COUNT statement causes the message control system to indicate to the COBOL program the number of complete messages in a specific queue.

The DISABLE statement breaks a logical connection between the message control system and one or more given communications devices.

The ENABLE statement establishes a logical connection between the message control system and one or more given communications devices.

The RECEIVE statement causes data in a queue to be passed to the COBOL program.

The SEND statement causes data associated with the COBOL program to be passed to one or more queues.
### ADDITIONS

<table>
<thead>
<tr>
<th>ALSO</th>
<th>DYNAMIC</th>
<th>REFERENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOTTOM</td>
<td>EGI</td>
<td>RELATIVE</td>
</tr>
<tr>
<td>CALL</td>
<td>EMI</td>
<td>REMOVAL</td>
</tr>
<tr>
<td>CANCEL</td>
<td>ENABLE</td>
<td>REWRITE</td>
</tr>
<tr>
<td>CD</td>
<td>END-OF-PAGE</td>
<td>SEGMENT</td>
</tr>
<tr>
<td>CHARACTER</td>
<td>EOP</td>
<td>SEND</td>
</tr>
<tr>
<td>CODE-SET</td>
<td>ESI</td>
<td>SEPARATE</td>
</tr>
<tr>
<td>COLLATING</td>
<td>EXCEPTION</td>
<td>SEQUENCE</td>
</tr>
<tr>
<td>COMMUNICATION</td>
<td>EXTEND</td>
<td>SORT-MERGE</td>
</tr>
<tr>
<td>COUNT</td>
<td>INITIAL</td>
<td>STANDARD-1</td>
</tr>
<tr>
<td>DATE</td>
<td>INSPECT</td>
<td>START</td>
</tr>
<tr>
<td>DAY</td>
<td>LENGTH</td>
<td>STRING</td>
</tr>
<tr>
<td>DEBUG-CONTENTS</td>
<td>LINEAGE</td>
<td>SUB-QUEUE-1</td>
</tr>
<tr>
<td>DEBUG-ITEM</td>
<td>LINEAGE-COUNTER</td>
<td>SUB-QUEUE-2</td>
</tr>
<tr>
<td>DEBUG-LINE</td>
<td>LINKAGE</td>
<td>SUB-QUEUE-3</td>
</tr>
<tr>
<td>DEBUG-NAME</td>
<td>MERGE</td>
<td>SUPPRESS</td>
</tr>
<tr>
<td>DEBUG-SUB-1</td>
<td>MESSAGE</td>
<td>SYMBOLIC</td>
</tr>
<tr>
<td>DEBUG-SUB-2</td>
<td>NATIVE</td>
<td>TABLE</td>
</tr>
<tr>
<td>DEBUG-SUB-3</td>
<td>ORGANIZATION</td>
<td>TERMINAL</td>
</tr>
<tr>
<td>DEBUGGING</td>
<td>OVERFLOW</td>
<td>TEXT</td>
</tr>
<tr>
<td>DELETE</td>
<td>POINTER</td>
<td>TIME</td>
</tr>
<tr>
<td>DELIMITED</td>
<td>PRINTING</td>
<td>TOP</td>
</tr>
<tr>
<td>DELIMITER</td>
<td>PROCEDURES</td>
<td>TRAILING</td>
</tr>
<tr>
<td>DESTINATION</td>
<td>PROGRAM</td>
<td>UNSTRING</td>
</tr>
<tr>
<td>DISABLE</td>
<td>QUEUE</td>
<td></td>
</tr>
<tr>
<td>DUPLICATES</td>
<td>RECEIVE</td>
<td></td>
</tr>
</tbody>
</table>

### DELETIONS

<table>
<thead>
<tr>
<th>ACTUAL</th>
<th>EXAMINE</th>
<th>NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>FILE-LIMIT</td>
<td>PROCESSING</td>
</tr>
<tr>
<td>BEGINNING</td>
<td>FILE-LIMITS</td>
<td>REMARKS</td>
</tr>
<tr>
<td>ENDING</td>
<td>KEYS</td>
<td>SEEK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TALLY</td>
</tr>
</tbody>
</table>
APPENDIX B

COMPOSITE LANGUAGE SKELETON
IDENTIFICATION DIVISION.

PROGRAM-ID. program-name.

[AUTHOR. [comment-entry]...]

[INSTALLATION. [comment-entry]...]

[DATE-WRITTEN. [comment-entry]...]

[DATE-COMPILED. [comment-entry]...]

[SECURITY. [comment-entry]...]

[REMARKS. [comment-entry]...]

REMARKS was deleted from the Standard, and replaced by the asterisk or slash in column 7 (Indicator Area).
X3.23-1968 General Format

ENVIRONMENT DIVISION.

CONFIGURATION SECTION.

Format 1:

SOURCE-COMPUTER. COPY library-name

\[\text{REPLACING word-1 BY \{identifier-1, literal-1\}}\]
\[\text{word-3 BY \{identifier-2, literal-2\}}\]...

Format 2:

SOURCE-COMPUTER. computer-name.

Format 1:

OBJECT-COMPUTER. COPY library-name

\[\text{REPLACING word-1 BY \{identifier-1, literal-1\}}\]
\[\text{word-3 BY \{identifier-2, literal-2\}}\]...

Format 2:

OBJECT-COMPUTER. computer-name

\[\text{MEMORY SIZE integer \{WORDS, CHARACTERS, MODULES\}}\]

\[\text{, SEGMENT-LIMIT IS segment-number}].\]

Changes for X3.23-1974

Format 1 of the SOURCE-COMPUTER paragraph has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)

SOURCE-COMPUTER. computer-name [WITH DEBUGGING MODE].

Format 1 of the OBJECT-COMPUTER paragraph has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)

\[\text{WORDS, PROGRAM COLLATING SEQUENCE IS alphabet-name}\]
X3.23-1968 General Format

CONFIGURATION SECTION (Continued)

Format 1:

SPECIAL-NAMES. COPY library-name

[REPLACING word-1 BY {identifier-1}
{literal-1}

[ word-3 BY {identifier-2}
{literal-2} ] ... ]

Format 2:

SPECIAL-NAMES. [implementor-name

IS mnemonic-name [, ON STATUS IS condition-name-1
IS mnemonic-name [, OFF STATUS IS condition-name-2
ON STATUS IS condition-name-1
OFF STATUS IS condition-name-2

[, OFF STATUS IS condition-name-2]]

[, ON STATUS IS condition-name-1]]

[, OFF STATUS IS condition-name-2]

[, ON STATUS IS condition-name-1]]

...]

[, CURRENCY SIGN IS literal]

[, DECIMAL-POINT IS COMMA].

Changes for X3.23-1974

Format 1 of the SPECIAL-NAMES paragraph has been deleted. A COPY statement can now appear anywhere in the COBOL program. (See the COPY Statement under the Procedure Division.)
INPUT-OUTPUT SECTION.

Format 1:

FILE-CONTROL COPY library-name

(REPLACING word-1 BY identifier-1)

(word-2)

(REPLACING word-1 BY identifier-2)

(literal-1)

(word-4)

(word-3 BY identifier-2) ...

Format 2:

FILE-CONTROL SELECT file-name

ASSIGN TO [integer-1] implementor-name-1 [, implementor-name-2] ...

FOR MULTIPLE [REEL UNIT]

[RESERVE integer-2] ALTERNATE [AREA AREAS]

{FILE-LIMIT IS data-name-1} {THROUGH} {data-name-2}

{FILE-LIMITS ARE literal-1} {THRU} {literal-2}

{data-name-3} {THRU} {data-name-4} ...

{l literal-3} {THRU} {literal-4} ...

ACCESS MODE IS [SEQUENTIAL]

[PROCESSING MODE IS SEQUENTIAL]

[ACTUAL KEY IS data-name-5]...

Changes for X3.23-1974

Format 1 of the FILE-CONTROL paragraph has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)

FILE-CONTROL {file-control-entry}...

File Control Entry

Format 1:

SELECT [OPTIONAL] file-name

ASSIGN TO implementor-name-1 [, implementor-name-2] ...

RESERVE integer-1 [AREA AREAS]

[; ORGANIZATION IS SEQUENTIAL]

[; ACCESS MODE IS SEQUENTIAL]

[; FILE STATUS IS data-name-1].
Changes for X3.23-1974

File Control Entry (Continued)

Format 2:

\[
\text{SELECT file-name}
\]
\[
\text{ASSIGN TO implementor-name-1 [, implementor-name-2]}...
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{RESERVE integer-1 [AREA AREAS]}
\end{array}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{ORGANIZATION IS RELATIVE}
\end{array}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{ACCESS MODE IS} \\
\quad \left\{ \begin{array}{l}
\text{SEQUENTIAL} [	ext{, RELATIVE KEY IS data-name-1}] \\
\text{RANDOM} \\
\text{DYNAMIC}
\end{array} \right. , \text{RELATIVE KEY IS data-name-1} \right\}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{FILE STATUS IS data-name-2}.
\end{array}
\end{array}\right]
\]

Format 3:

\[
\text{SELECT file-name}
\]
\[
\text{ASSIGN TO implementor-name-1 [, implementor-name-2]}...
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{RESERVE integer-1 [AREA AREAS]}
\end{array}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{ORGANIZATION IS INDEXED}
\end{array}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{ACCESS MODE IS} \\
\quad \left\{ \begin{array}{l}
\text{SEQUENTIAL} \\
\text{RANDOM} \\
\text{DYNAMIC}
\end{array} \right\} 
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{RECORD KEY IS data-name-1}
\end{array}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{ALTERNATE RECORD KEY IS data-name-2 [WITH DUPLICATES]}...
\end{array}
\end{array}\right]
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{FILE STATUS IS data-name-3}.
\end{array}
\end{array}\right]
\]

Format 4:

\[
\text{SELECT file-name}
\]
\[
\text{ASSIGN TO implementor-name-1 [, implementor-name-2]}...
\]

FILE-CONTROL. \{ SELECT file-name
\]
\[
\text{ASSIGN TO implementor-name-1 [, implementor-name-2]}...
\]
\[
\text{OR implementor-name-3 [, implementor-name-4]}...
\]
\[
\left[\begin{array}{l}
\begin{array}{l}
\text{FOR MULTIPLE [REEL UNIT]}.
\end{array}
\end{array}\right]...
\]

X3.23-1968 General Format

File Control Paragraph (Continued)
X3.23-1968 General Format

INPUT-OUTPUT SECTION (Continued)

Format 1:

I-O-CONTROL. COPY library-name

REPLACING word-1 BY {word-2
{identifier-1}
literal-1

word-3 BY {identifier-2}
literal-2

...

Format 2:

I-O-CONTROL. [; REPRUN [ON {file-name-1
{implementor-name}]

every

{[END OF] REEL UNIT} OF file-name-2

integer-1 RECORDS

integer-2 CLOCK-UNITS

condition-name

[; SAME {RECORD SORT} AREA FOR file-name-3 {, file-name-4}... ]...

[; MULTIPLE FILE TAPE CONTAINS file-name-5 [POSITION integer-3]

[, file-name-6 [POSITION integer-4]]... ]...

Changes for X3.23-1974

Format 1 of the I-O-CONTROL paragraph has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)
DATA DIVISION.

FILE SECTION.

File Description Entry

Format 1:

FD file-name; COPY library-name

[ REPLACING word-1 BY \{identifier-1
\{literal-1 \}
\}, word-3 BY \{identifier-2
\{literal-2 \} \} ... ].

Format 2:

FD file-name

[ ; BLOCK CONTAINS \{integer-1 TO integer-2 \} \{RECORDS \{CHARACTERS\} \} ]

[ ; DATA \{RECORD IS \{RECORDS ARE \} \{data-name-1 \{, data-name-2\} ... \} ]

[ ; LABEL \{RECORD IS \{RECORDS ARE \} \{STANDARD \{OMITTED \} \} \{data-name-3 \{, data-name-4\} ... \} ]

[ ; RECORD CONTAINS \{integer-3 TO integer-4 \} \{CHARACTERS\} ]

[ ; VALUE OF data-name-5 IS \{data-name-6 \{literal-1 \} ]

[ , data-name-7 IS \{data-name-8 \{literal-2 \} ... \} ].

Format 1 of the FD has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)

Changes for X3.23-1974

Label \{RECORD IS \{RECORDS ARE \} \{STANDARD \{OMITTED \} ]

[ ; VALUE OF implementer-name-1 IS \{data-name-3 \{literal-1 \} ]

[ , implementer-name-2 IS \{data-name-4 \{literal-2 \} ... \} ]

[ ; LINAGE IS \{data-name-5 \{integer-5 \} \{LINES \{WITH FOOTING AT \{data-name-6 \{integer-6 \} \} ]

[ , LINES AT TOP \{data-name-7 \{integer-7 \} ]

[ , LINES AT BOTTOM \{data-name-8 \{integer-8 \} ]

[ ; CODE-SET IS alphabet-name].
X3.23-1968 General Format

File Description Entry (Continued)

Format 3:

FD file-name

[; BLOCK CONTAINS [integer-1 TO integer-2] RECORDS (CHARACTERS)]

; LABEL {RECORD IS } {STANDARD}

; LABEL {RECORDS ARE} {OMITTED}

[; RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS]

[; REPORT IS ] report-name-1 [, report-name-2]...

[; VALUE OF data-name-3 IS {data-name-4}

[ , data-name-5 IS {data-name-6} ] ... ].

Sort Description Entry

Format 1:

SD file-name; COPY library-name

[ REPLACING word-1 BY {word-2} identifier-1

[ , word-3 BY {word-4} identifier-2 } ] ... ].

Changes for X3.23-1974

Format 1 of the SD has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)
COMMUNICATION SECTION (Continued)

REPORT SECTION.

Report Description Entry

Format 1

RD report-name; COPY library-name

[ REPLACING word-1 BY { word-2
  identifier-1
  literal-1 }

[ word-3 BY { word-4
  identifier-2
  literal-2 } ] ]...

Changes for X3.23-1974

COMMUNICATION SECTION (Continued)

Format 2

CD cd-name; FOR OUTPUT

[ ; DESTINATION COUNT IS data-name-1 ]
[ ; TEXT LENGTH IS data-name-2 ]
[ ; STATUS KEY IS data-name-3 ]
[ ; DESTINATION TABLE OCCURS integer-2 TIMES
  [ ; INDEXED BY index-name-1 [, index-name-2]... ] ]
[ ; ERROR KEY IS data-name-4 ]
[ ; SYMBOLIC DESTINATION IS data-name-5 ].

Format 1 of the RD has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)
X3.23-1968 General Format

Report Description Entry (Continued)

Format 2:

RD report-name
[; CODE mnemonic-name-1]

[] {CONTROL IS } {FINAL, identifier-1 [ ], identifier-2] ... }

[; CONTROLS ARE ] {FINAL, identifier-1 [ , identifier-2] ... }]

[]; PAGE [LIMIT IS ] integer-1 [ LINE ]

[ , LIMITS ARE ] integer-1 [ LINES ]

[ , FIRST DETAIL integer-3]

[ , LAST DETAIL integer-4] [ , FOOTING integer-5] .

Report Group Description Entry

Format 1:

01 data-name-1; COPY library-name

[] REPLACING word-1 BY {identifier-1}

[ , word-2 BY {identifier-2 }]

[ , word-3 BY {identifier-3 }]

[ , word-4 BY {identifier-4 }]

[ , word-5 BY {identifier-5 }] .

Changes for X3.23-1974

Format 1 of the Report Group Description entry has been deleted.
A COPY statement can now appear any place in the COBOL program.
(See the COPY Statement under the Procedure Division.)
X3.23-1968 General Format

Report Group Description Entry (Continued)

Format 2:

01 [data-name-1]

[ ; LINE NUMBER IS {integer-1 PLUS integer-2} ]

[ ; NEXT GROUP IS {integer-3 PLUS integer-4} ]

REPORT HEADING
RH
PAGE HEADING
PH
(CONTROL HEADING) {identifier-1}
CH
DETAIL
DE
(CONTROL FOOTING) {identifier-2}
CF
PAGE FOOTING
PF
REPORT FOOTING
RF

[ ; USAGE IS] DISPLAY.

Changes for X3.23-1974

[ ; LINE NUMBER IS {integer-1 [ON NEXT PAGE]} ]

[ ; TYPE IS ]

(REPORT HEADING)
RH
(PAGE HEADING)
PH
(CONTROL HEADING)
CH
(DETAIL)
DE
(CONTROL FOOTING)
CF
(PAGE FOOTING)
PF
(REPORT FOOTING)
RF

New Format 2:

level-number [data-name-1]

[ ; LINE NUMBER IS {integer-1 [ON NEXT PAGE]} ]

[ ; USAGE IS] DISPLAY.
X3.23-1968 General Format

Report Group Description Entry (Continued)

Format 3:

level-number [data-name-1]

[ ; BLANK WHEN ZERO ]

[ ; COLUMN NUMBER IS integer-1 ]

[ ; GROUP INDICATE ]

[ ; {JUSTIFIED} } RIGHT ]

[ ; LINE NUMBER IS \{ integer-2 \} \{ PLUS integer-3 \} ]

{NEXT PAGE}

[ ; {PICTURE} IS character-string ]

[ ; RESET ON \{ identifier-1 \} ]

\{FINAL\}

\{SOURCE IS identifier-2 \}

\{SUM identifier-3 [, identifier-4] [UPON data-name-2] \}

\{VALUE IS literal-1 \}

\{USAGE IS DISPLAY\}. 

Changes for X3.23-1974

[ ; LINE NUMBER IS \{integer-1 [ON NEXT PAGE] \} ]

\{PLUS integer-2 \}

\{PICTURE \} IS character-string

RESET clause is now included within the SUM clause

\{SOURCE IS identifier-1 \}

\{VALUE IS literal \}

\{SUM identifier-2 [, identifier-3] [UPON data-name-2 [, data-name-3] ... ] \}

[UPON data-name-2 [, data-name-3] ... ]

\{RESET ON \{data-name-4 \} \}

\{FINAL\}
Record Description Entry

Format 1:

01 data-name-1; COPY library-name

[REPLACING word-1 BY {identifier-1, literal-1}]

[ word-3 BY {identifier-2, literal-2} ] ...

Format 2:

level-number {data-name-1}

[; REDEFINES data-name-2]

[; BLANK WHEN ZERO]

[; {JUSTIFIED} RIGHT]

[ ; OCCURS {integer-1 TO integer-2 TIMES [DEPENDING ON data-name-3], integer-2 TIMES

{ASCENDING, DESCENDING} KEY IS data-name-4 [, data-name-5]...}]

[INDEXED BY index-name-1 [, index-name-2]...]

[; {PICTURE} IS character-string]

[; {SYNCHRONIZED} LEFT]

[; {USAGE} COMPUTATIONAL, DISPLAY INDEX]

[; {SIGN} LEADING, TRAILING, SEPARATE CHARACTER]

[; VALUE IS literal-3].

Changes for X3.23-1974

Format 1 of the Record Description Entry has been deleted. A COPY statement can now appear any place in the COBOL program. (See the COPY Statement under the Procedure Division.)
X3.23-1968 General Format

Record Description Entry (Continued)

Format 3:

66 data-name-1; RENAMES data-name-2 [{THROUGH\THRU} data-name-3].

Format 4:

88 condition-name

; \{VALUE IS \}
\{VALUES ARE\} literal-1 [{THROUGH\THRU} literal-2]

[, literal-3 [{THROUGH\THRU} literal-4]] ... .
X3.23-1968 General Format

Procedure Division Structure

Format 1:

PROCEDURE DIVISION.

[DECLARATIVES.
{section-name SECTION. declarative-sentence
{paragraph-name. [sentence]...}...]
END DECLARATIVES.]

{section-name SECTION [segment-number].
{paragraph-name. [sentence]...}...}

Format 2:

{paragraph-name. [sentence]...}...

ACCEPT Statement

Format 1:

ACCEPT identifier [FROM mnemonic-name]

Format 2:

ACCEPT identifier FROM (DATE)

ACCEPT identifier FROM (DATE

ACCEPT identifier FROM (TIME)

Format 3:

ACCEPT cd-name MESSAGE COUNT

Changes for X3.23-1974

PROCEDURE DIVISION [USING data-name-1 [, data-name-2]...].

[DECLARATIVES.
{section-name SECTION [segment-number]. declarative-sentence
{paragraph-name. [sentence]...}...]
END DECLARATIVES.]

{section-name SECTION [segment-number].
{paragraph-name. [sentence]...}...}

{paragraph-name. [sentence]...}...

ACCEPT identifier [FROM mnemonic-name]
```
ADD Statement

Format 1:
ADD {identifier-1} [identifier-2] ...
    TO identifier-m [ROUNDED] [identifier-n [ROUNDED]] ...
    [; ON SIZE ERROR imperative-statement]

Format 2:
ADD {identifier-1} {identifier-2} [identifier-3] ...
    GIVING identifier-m [ROUNDED]
    [; ON SIZE ERROR imperative-statement]

Format 3:
ADD {CORRESPONDING} {identifier-1}
    TO identifier-2 [ROUNDED]
    [; ON SIZE ERROR imperative-statement]

ALTER Statement

ALTER procedure-name-1 TO [PROCEED TO] procedure-name-2
    [, procedure-name-3 TO [PROCEED TO] procedure-name-4] ...

CALL Statement

CALL {identifier-1} [USING data-name-1 [, data-name-2]] ...
    [; ON OVERFLOW imperative-statement]

CANCEL Statement

CANCEL {identifier-1} [identifier-2] ...
```
CLOSE Statement

CLOSE file-name-1 [REEL UNIT] (WITH {NO Rewind}]

[, file-name-2 REEL UNIT] (WITH {NO Rewind}]

COMPUTE Statement

COMPUTE identifier-1 [ROUNDED]

\[
\begin{align*}
&= \text{identifier-2} \\
&\text{\textit{arithmetic-expression}} \\
&; \text{ON SIZE ERROR imperative-statement}
\end{align*}
\]

COPY Statement

COPY library-name

\[
\begin{align*}
&\text{REPLACING word-1 BY} \{ \text{identifier-1} \\
&\text{\textit{literal-1}} \} \\
&\text{\textit{word-2}} \text{\textbf{BY}} \{ \text{identifier-2} \} \\
&\text{\textit{.literal-2}} \text{\ldots}
\end{align*}
\]

DELETE Statement

DISABLE Statement

Changes for X3.23-1974

CLOSE file-name-1 [REEL UNIT] (WITH {NO Rewind}]

[, file-name-2 REEL UNIT] (WITH {NO Rewind}]

COMPUTE identifier-1 [ROUNDED] [, identifier-2 [ROUNDED]]

\[
= \text{arithmetic-expression}
\]

COPY text-name [OF] \text{library-name}

\[
\begin{align*}
&\text{REPLACING} \{ \text{identifier-1} = \text{pseudo-text-1} \} \\
&\text{\textit{identifier-2}} \text{\textbf{BY}} \{ \text{pseudo-text-2} \} \\
&\text{\textit{word-1}} \text{\textbf{BY}} \{ \text{literal-2} \} \\
&\text{\textit{word-2}} \text{\ldots}
\end{align*}
\]

DELETE file-name RECORD [; INVALID KEY imperative-statement]

DISABLE \{INPUT [TERMINAL] \text{\textbf{cd-name}} WITH KEY \{identifier-1\} \text{\textbf{literal-1}} \}
DISPLAY Statement

DISPLAY \{identifier-1\} \{literal-1\} \{identifier-2\} \{literal-2\} \... \UPON mnemonic-name

DIVIDE Statement

Format 1:

DIVIDE \{identifier-1\} \{literal-1\} INTO identifier-2 \[ROUNDED\]

[; ON SIZE ERROR imperative-statement]

Format 2:

DIVIDE \{identifier-1\} \{literal-1\} INTO \{identifier-2\} \{literal-2\}

GIVING identifier-3 \[ROUNDED\]

[; ON SIZE ERROR imperative-statement]

Format 3:

DIVIDE \{identifier-1\} \{literal-1\} BY \{identifier-2\} \{literal-2\}

GIVING identifier-3 \[ROUNDED\]

[; ON SIZE ERROR imperative-statement]

Format 4:

DIVIDE \{identifier-1\} \{literal-1\} INTO \{identifier-2\} \{literal-2\}

GIVING identifier-3 \[ROUNDED\] REMAINDER identifier-4

[; ON SIZE ERROR imperative-statement]
X3.23-1968 General Format

DIVIDE Statement (Continued)

Format 5:

```
DIVIDE {identifier-1} BY {identifier-2}

GIVING identifier-3 [ROUNDED] REMAINDER identifier-4

[; ON SIZE ERROR imperative-statement]
```

ENABLE Statement

```
ENABLE { INPUT [TERMINAL] } cd-name WITH KEY {identifier-1}
```

ENTER Statement

```
ENTER language-name [routine-name].
```

EXAMINE Statement

```
EXAMINE identifier

TALLYING { ALL [LEADING] } literal-1 [REPLACING BY literal-2]

REPLACING { ALL [LEADING] [UNTIL] FIRST } literal-3 BY literal-4
```

EXIT Statement

```
EXIT.
```

GENERATE Statement

```
GENERATE identifier
```

GO Statement

```
GO TO [procedure-name-1]
```

Changes for X3.23-1974

The EXAMINE statement was deleted from the Standard. The INSPECT statement includes the functions of the EXAMINE statement.
X3.23-1968 General Format

GO Statement (Continued)

Format 2:

GO TO procedure-name-1 [, procedure-name-2]...

, procedure-name-n DEPENDING ON identifier

IF Statement

IF condition; \{statement-1 \} ; ELSE \{statement-2 \}

INITIATE Statement

INITIATE report-name-1 [, report-name-2]...

INSPECT Statement

Changes for X3.23-1974

GO TO procedure-name-1 [, procedure-name-2]...

, procedure-name-n DEPENDING ON identifier

Format 1:

INSPECT identifier-1 TALLYING [, identifier-2 FOR

\{ \{ ALL \{ identifier-3 \} \}

\{ LEADING \{ literal-1 \} \}

\{ CHARACTERS \}

[ \{ BEFORE \} INITIAL \{ identifier-4 \} \} ...

Format 2:

INSPECT identifier-1 REPLACING

\{ CHARACTERS BY \{ identifier-6 \}

\{ literal-4 \}

\{ \{ BEFORE \} INITIAL \{ identifier-7 \} \}

\{ literal-5 \}

\{ \{ ALL \{ LEADING \{ identifier-5 \} \} \} BY \{ identifier-6 \}

\{ literal-3 \} \}

\{ \{ BEFORE \} INITIAL \{ identifier-7 \} \} ...

\{ literal-4 \} \}

\{ \{ AFTER \} INITIAL \{ identifier-5 \} \} ...

\{ literal-5 \} \} ...
INSPECT Statement (Continued)

Format 3:

INSPECT identifier-1

TALLYING 

\{ identifier-2 FOR \} \{ ALL \} \{ identifier-3 \} \{ identifier-4 \} \{ literal-1 \} \{ literal-2 \} \{ \ldots \} \ldots

\{ Character BY \} \{ identifier-5 \} \{ literal-3 \} \{ literal-4 \} \{ \ldots \} \ldots

REPLACING 

\{ ALL \} \{ identifier-5 \} \{ literal-3 \} \{ \ldots \} \ldots

\{ Character BY \} \{ identifier-6 \} \{ literal-4 \} \{ \ldots \} \ldots

MERGE file-name-1 ON \{ ASCENDING \} KEY data-name-1 \{ \ldots \}

\{ ON \} \{ DESCENDING \} KEY data-name-3 \{ \ldots \}

[COLLATING SEQUENCE IS alphabet-name]

USING file-name-2, file-name-3 \{ \ldots \}

\{ OUTPUT PROCEDURE IS section-name-1 \{ \ldots \}

\{ Giving file-name-5 \}
MOVE Statement

Format 1:

MOVE {identifier-1} TO identifier-2 [, identifier-3]...

Format 2:

MOVE {CORRESPONDING} {identifier-1 TO identifier-2}

MULTIPLY Statement

Format 1:

MULTIPLY {identifier-1} BY identifier-2 [ROUNDED]

[; ON SIZE ERROR imperative-statement]

Format 2:

MULTIPLY {identifier-1} BY {identifier-2}

GIVING identifier-3 [ROUNDED]

[; ON SIZE ERROR imperative-statement]

NOTE Statement

NOTE character-string.

OPEN Statement

OPEN {INPUT file-name-1 [REVERSED] [WITH NO REWIND]}

{OUTPUT file-name-3 [WITH NO REWIND] I-O file-name-5 [WITH NO REWIND]...}

[, file-name-2 [REVERSED] [WITH NO REWIND] ]... }

[, file-name-4 [ WITH NO REWIND] ]... }

[, file-name-6]...

[, file-name-0]...

The NOTE statement was deleted from the Standard and replaced by the asterisk or slash in column 7.
PERFORM Statement

Format 1:

PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]

Format 2:

PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]

{identifier-1} TIMES

Format 3:

PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]

UNTIL condition-1

Format 4:

PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]

VARYING {index-name-1} {identifier-1} FROM {identifier-2}

BY {liter-1} UNTIL condition-1

[AFTER {index-name-2} FROM {identifier-3}] BY {identifier-4}

[AFTER {index-name-3} FROM {identifier-5}] BY {identifier-6}

[AFTER {index-name-4} FROM {identifier-7}] BY {identifier-8}

[AFTER {index-name-5} FROM {identifier-9}] UNTIL condition-3]
READ Statement

Format 1:

READ file-name RECORD [INTO identifier]

; AT END imperative-statement

Format 2:

READ file-name RECORD [INTO identifier]

; INVALID KEY imperative-statement

RECEIVE Statement

RELEASE Statement

RELEASE record-name [FROM identifier]

RETURN Statement

RETURN file-name RECORD [INTO identifier]

; AT END imperative-statement

REWRITE Statement

Changes for X3.23-1974

READ file-name [NEXT] RECORD [INTO identifier]

[; AT END imperative-statement]

READ file-name RECORD [INTO identifier] [; KEY IS data-name]

[; INVALID KEY imperative-statement]

RECEIVE cd-name [MESSAGE] [SEGMENT] INTO identifier-1

[; NO DATA imperative-statement]

REWRITE record-name [FROM identifier]

[; INVALID KEY imperative-statement]
SEARCH Statement

Format 1:

```
SEARCH identifier-1 [ VARYING {index-name-1} ]

[; AT END imperative-statement-1]

; WHEN condition-1 {imperative-statement-2} NEXT SENTENCE

[; WHEN condition-2 {imperative-statement-3} ] ...
```

Format 2:

```
SEARCH ALL identifier-1 [; AT END imperative-statement-1]

; WHEN {data-name-1 \{IS EQUAL TO\} identifier-3

\{IS = literal-1 \{arithmetic-expression-1\}\)

\{condition-name-1\}

AND {data-name-2 \{IS EQUAL TO\} identifier-4

\{IS = literal-2 \{arithmetic-expression-2\}\)

\{condition-name-2\}

\{imperative-statement-2\} NEXT SENTENCE
```

SEEK Statement

```
SEEK file-name RECORD
```

SEND Statement

```
SEND cd-name FROM identifier-1
```
X3.23-1968 General Format

SEND Statement (Continued)

SET Statement

Format 1:

\[
\text{SET} \left\{ \text{identifier-1} [, \text{identifier-2}] \ldots \right\} \text{ TO } \left\{ \text{index-name-3} \right\}
\]

Format 2:

\[
\text{SET} \left\{ \text{index-name-1} [, \text{index-name-2}] \ldots \right\} \text{ UP BY } \left\{ \text{identifier-4} \right\}
\]

SORT Statement

\[
\text{SORT} \left\{ \text{file-name-1} \right\} \text{ ON } \left\{ \text{DESCENDING} \right\} \text{ KEY } \left\{ \text{data-name-1} [, \text{data-name-2}] \ldots \right\}
\]

\[
\text{ON} \left\{ \text{DESCENDING} \right\} \text{ KEY } \left\{ \text{data-name-3} [, \text{data-name-4}] \ldots \right\}
\]

\[
\text{COLLATING SEQUENCE IS } \text{alphabet-name}
\]

Changes for X3.23-1974

Format 2:

\[
\text{SEND} \left\{ \text{cd-name} \text{ FROM } \text{identifier-1} \right\} \text{ WITH identifier-2}
\]

\[
\left\{ \text{identifier-3} \right\} \text{ LINE LINES}
\]

\[
\text{SET} \left\{ \text{index-name-1} [, \text{index-name-2}] \ldots \right\} \text{ UP BY } \left\{ \text{identifier-4} \right\}
\]

\[
\text{SET} \left\{ \text{index-name-1} [, \text{index-name-2}] \ldots \right\} \text{ DOWN BY } \left\{ \text{integer-2} \right\}
\]

\[
\text{USING} \left\{ \text{file-name-2} \right\}
\]

\[
\text{USING} \left\{ \text{file-name-2} [, \text{file-name-3}] \right\}
\]
X3.23-1968 General Format

START Statement

STOP Statement

STOP [RUN] [literal]

STRING Statement

SUBTRACT Statement

Format 1:

SUBTRACT [literal-1] [identifier-1] [; literal-2] [identifier-2] ... FROM identifier-m [ROUNDED] [ , identifier-n [ROUNDED]]...

[; ON SIZE ERROR imperative-statement]

Format 2:

SUBTRACT [literal-1] [identifier-1] [; literal-2] [identifier-2] ... FROM [literal-m] [identifier-m]

GIVING identifier-n [ROUNDED]

[; ON SIZE ERROR imperative-statement]

Changes for X3.23-1974

START file-name [KEY [IS EQUAL TO | IS =
IS GREATER THAN | IS >
IS NOT LESS THAN | IS NOT <
IS NOT <]
data-name

[; INVALID KEY imperative-statement]

STRING [identifier-1] [; identifier-2] ... DELIMITED BY [identifier-3]

[ ; identifier-4] [ ; identifier-5] ... DELIMITED BY [identifier-6]

SIZE

INTO identifier-7 [WITH POINTER identifier-0]

[ ; ON OVERFLOW imperative-statement]
SUBTRACT Statement (Continued)

Format 3:

```
SUBTRACT (CORRESPONDING) \{identifier-1 FROM identifier-2 [ROUNDED]

[; ON SIZE ERROR imperative-statement]
```

SUPPRESS Statement

TERMINATE Statement

```
TERMINATE report-name-1 [, report-name-2]...
```

UNSTRING Statement

UNSTRING identifier-1

```
[DELIMITED BY [ALL] \{identifier-2 \} [, OR [ALL] \{identifier-3 \}] ... ]

INTO identifier-4 [, DELIMITER IN identifier-5]

[, COUNT IN identifier-6]

[, identifier-7 [, DELIMITER IN identifier-8]

[, COUNT IN identifier-9]]...

[WITH POINTER identifier-10] [TALLYING IN identifier-11]

[; ON OVERFLOW imperative-statement]
```

USE Statement

```
USE AFTER STANDARD ERROR PROCEDURE ON

\{
  file-name-1 [, file-name-2]...
  \}

\{
  INPUT
  OUTPUT
  I-O
}\`
```

USE AFTER STANDARD (EXCEPTION ERROR) PROCEDURE ON

```
\{
  file-name-1 [, file-name-2]...
  \}

\{
  INPUT
  OUTPUT
  I-O
  EXTEND
\}
X3.23-1968 General Format

USE Statement (Continued)

Format 2:

USE {BEFORE} STANDARD {BEGINNING} [REEL]
   {END} FILE
   UNIT

LABEL PROCEDURE ON {file-name-1 [, file-name-2] ...}

INPUT
   OUTPUT
   I-O

Format 3:

USE BEFORE REPORTING identifier-1.

WRITE Statement

Format 1:

WRITE record-name [FROM identifier-1]
   {BEFORE} ADVANCING {identifier LINES}
   {AFTER} integer LINES
   {mnemonic-name}

Changes for X3.23-1974

Format 2 of the USE statement was deleted from the Standard.

Format 4:

USE FOR DEBUGGING ON {cd-name-1
   [ALL REFERENCES OF] identifier-1}
   {file-name-1
   procedure-name-1
   ALL PROCEDURES
   cd-name-2
   [ALL REFERENCES OF] identifier-2
   file-name-2
   procedure-name-2
   ALL PROCEDURES
   ...

   {BEFORE} ADVANCING {identifier-2
   integer
   mnemonic-name
   LINE
   LINES
   PAGE
   IMPERATIVE-STATEMENT
   AT END-OF-PAGE
   IMP
WRITE Statement (Continued)

Format 2:
WRITE record-name [FROM identifier-1]
; INVALID KEY imperative-statement

Changes for X3.23-1974

; INVALID KEY imperative-statement
**NBS TECHNICAL PUBLICATIONS**

**PERIODICALS**

**JOURNAL OF RESEARCH** reports National Bureau of Standards research and development in physics, mathematics, and chemistry. It is 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, $8.90; Foreign, $11.25.

**DIMENSIONS/NBS** (formerly Technical News Bulletin)—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. The magazine 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, it 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, $9.45; Foreign, $11.85.

**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 conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, 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).

**NOTE:** At present the principal publication outlet for these data is the Journal of Physical and Chemical Reference Data (JPCRD) published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements available from ACS, 1155 Sixteenth St. N.W., Wash. D.C. 20036.

**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 Federal Code of 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. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.


**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** (NBSIR)—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. 22161) in paper copy or microfiche form.


**BIBLIOGRAPHIC SUBSCRIPTION SERVICES**

The following current-awareness and literature-survey bibliographies are issued periodically by the Bureau: Cryogenic Data Center Current Awareness Service

A literature survey issued bimonthly. Annual subscription: Domestic, $20.00; foreign, $25.00.


Send subscription orders and remittances for the preceding bibliographic services to National Technical Information Service, Springfield, Va. 22161.

**Electromagnetic Metrology Current Awareness Service**

Issued monthly. Annual subscription: $100.00 (Special rates for multi-subscriptions). Send subscription order and remittance to Electromagnetics Division, National Bureau of Standards, Boulder, Colo. 80302.