무료 전자 책

배우기

cobol

Free unaffiliated eBook created from
Stack Overflow contributors.

#cobol

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: cobol

It is an unofficial and free cobol ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official cobol.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: cobol

COBOL **O L** anguage riented MMON **usiness B.**

COBOL COBOL COBOL ISO INCITS .

.

ISO / IEC 1989 : 2014 - , - COBOL

2014 5 .

http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=51416

. . , COBOL . IBM COBOL . 100 COBOL , 10 . COBOL .

COBOL . COBOL 2002 Object Oriented .

COBOL . :

```
COMPUTE I = R * B
```

.

```
MULTIPLY INTEREST-RATE BY BALANCE GIVING CURRENT-INTEREST ROUNDED MODE IS NEAREST-EVEN
```

. , .

COBOL 2 . COBOL . *COBOL USAGE BINARY (10) .*

COBOL 1950 , 1960 .

(Grace Hopper) COBOL , . COBOL COBOL .

COBOL . . . , COBOL . .

COBOL 4 .

-
-
-
-

10 COBOL PICTURE .

```
01 record-group.
   05 balance        pic s9(8)v99.
   05 rate           pic 999v999.
   05 show-balance   pic $Z(7)9.99.
```

`balance` 8 . `rate` 3 3 . `show-balance` ( ) .

`balance` , `show-balance` .

COBOL . MOVE, COMPUTE, MULTIPLY, PERFORM . COBOL 2014 300 47 . .

Examples

.

```
HELLO * HISTORIC EXAMPLE OF HELLO WORLD IN COBOL
       IDENTIFICATION DIVISION.
       PROGRAM-ID. HELLO.
       PROCEDURE DIVISION.
           DISPLAY "HELLO, WORLD".
           STOP RUN.
```

. COBOL . ( ) .

.

```
*> Hello, world
identification division.
program-id. hello.

procedure division.
display "Hello, world"
goback.
end program hello.
```

COBOL .

```
display "Hello, world".
```

`DIVISION` `DIVISION` COBOL .

COBOL FIXED .

2002 COBOL

1-6	
7	
8-12	A
12-72	B
73-80	

IBM .

2002 COBOL 2014 A B 255 .

1-6
7
8-

C *R* 8 72 255 .

COBOL 2002 `FORMAT FREE` . *Sequence Number Area* , *Indicator Area* (*R* , 2048 , 255).

FORMAT FIXED . .

```
bbbbbb >>SOURCE FORMAT IS FREE
```

`bbbbbb` 6 i . (.)

Mac OS X gnu-cobol .

gnu-cobol homebrew .

`/Applications/Utilities/Terminal` `Command+Space` `"Terminal"` .

:

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

.

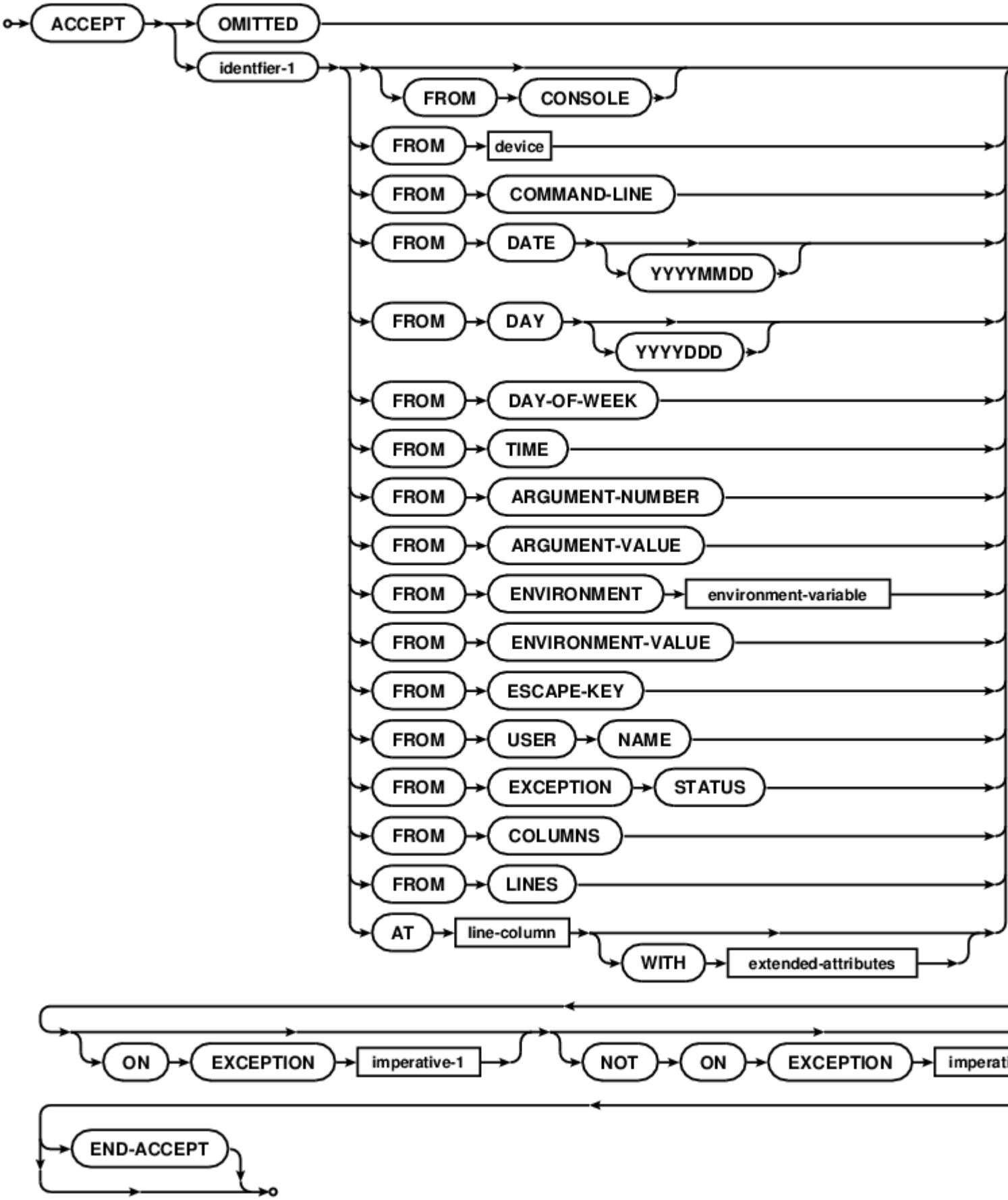
```
brew install gnu-cobol
```

, Mac Cobol .

cobol : https://riptutorial.com/ko/cobol/topic/4728/cobol-

2: ACCEPT

COBOL ACCEPT .



Examples

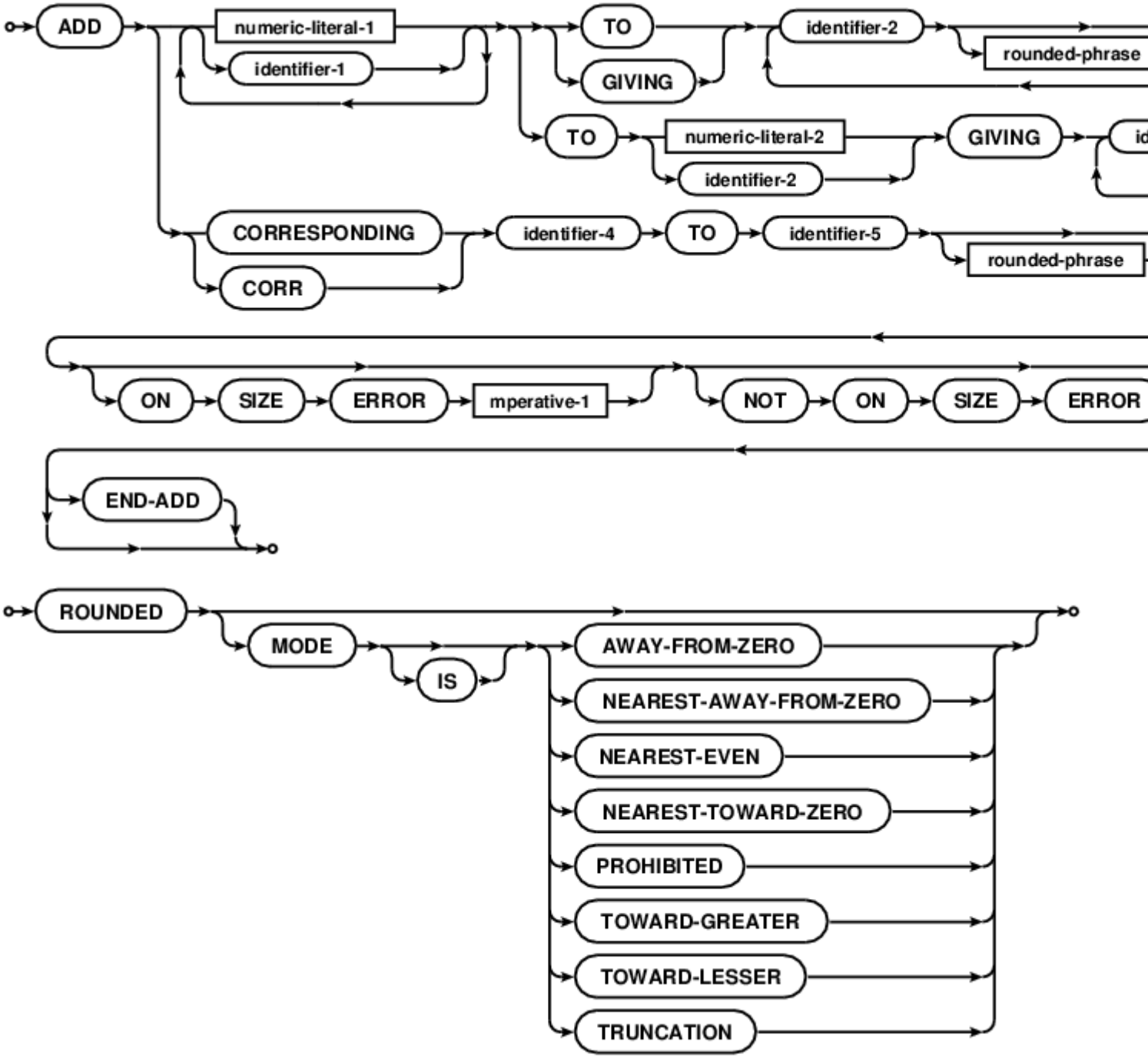
ACCEPT

```
ACCEPT variable.
ACCEPT variable FROM CONSOLE.

ACCEPT variable FROM ENVIRONMENT "path".
ACCEPT variable FROM COMMAND-LINE.

ACCEPT variable FROM ARGUMENT-NUMBER
ACCEPT variable FROM ARGUMENT-VALUE

ACCEPT variable AT 0101.
ACCEPT screen-variable.

ACCEPT today FROM DATE.
ACCEPT today FROM DATE YYYYMMDD.
ACCEPT thetime FROM TIME.

ACCEPT theday FROM DAY.
ACCEPT theday FROM DAY YYYYDDD.

ACCEPT weekday FROM DAY-OF-WEEK.

ACCEPT thekey FROM ESCAPE KEY.

ACCEPT username FROM USER NAME.

ACCEPT exception-stat FROM EXCEPTION STATUS.

ACCEPT some-data FROM device-name.
```

http://open-cobol.sourceforge.net/faq/index.html#accept .

ACCEPT : https://riptutorial.com/ko/cobol/topic/5512/accept-

3: ADD



Examples

ADD

```
ADD 1 TO cobol
```

```
cobol . .
```

```
ADD 1 TO cobol GIVING GnuCOBOL
```

GnuCOBOL ADD cobol . , ( ).

```
ADD
    a b c d f g h i j k l m n o p q r s t u v w x y z
    GIVING total-of
    ON SIZE ERROR
        PERFORM log-problem
    NOT ON SIZE ERROR
        PERFORM graph-result
END-ADD
```

. COBOL FUNCTION E .

COBOL SIZE ERROR / PICTURE .PIC 9 0 9 , 10 ON SIZE ERROR .

ADD : https://riptutorial.com/ko/cobol/topic/5533/add-

4: ALLOCATE

BASED (heap) .

:

Examples

ALLOCATE

```
01 pointer-var         usage POINTER.
01 character-field     pic x(80) BASED value "Sample".

ALLOCATE 1024 characters returning pointer-var
ALLOCATE character-field
ALLOCATE character-field INITIALIZED RETURNING pointer-var
```

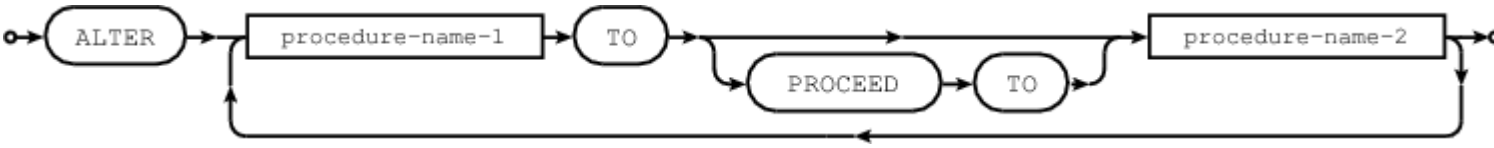
http://open-cobol.sourceforge.net/faq/index.html#allocate .

ALLOCATE : https://riptutorial.com/ko/cobol/topic/5556/allocate-

5: ALTER

ALTER . GO TO .

COBOL . *(COBOL).*



Examples

ALTER

```
 identification division.
 program-id. altering.
 date-written. 2015-10-28/06:36-0400.
 remarks. Demonstrate ALTER.

 procedure division.
 main section.

*> And now for some altering.
 contrived.
 ALTER story TO PROCEED TO beginning
 GO TO story
 .

*> Jump to a part of the story
 story.
 GO.
 .

*> the first part
 beginning.
 ALTER story TO PROCEED to middle
 DISPLAY "This is the start of a changing story"
 GO TO story
 .

*> the middle bit
 middle.
 ALTER story TO PROCEED to ending
 DISPLAY "The story progresses"
 GO TO story
 .

*> the climatic finish
 ending.
 DISPLAY "The story ends, happily ever after"
 .

*> fall through to the exit
 exit program.
```

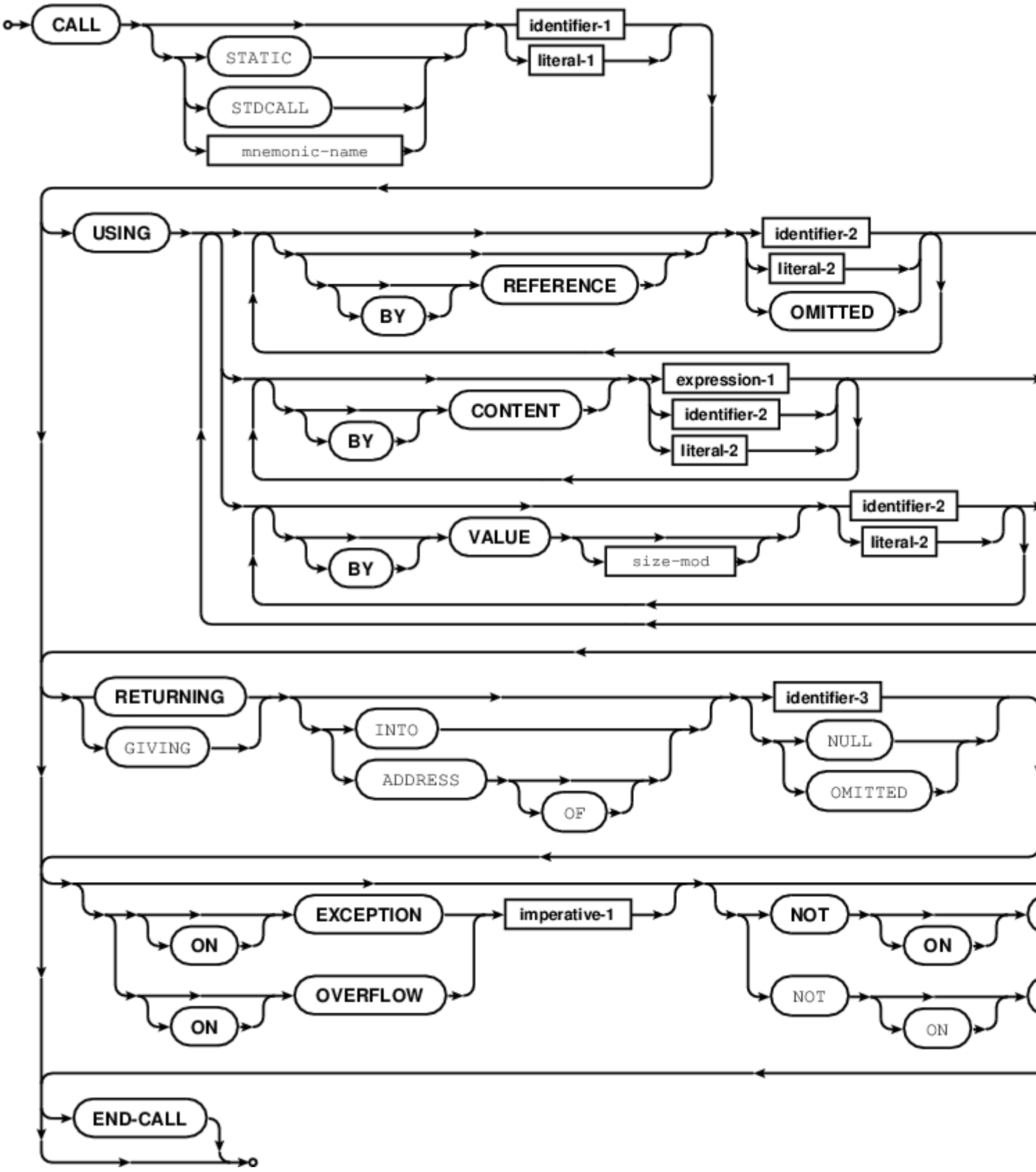
```
prompt$ cobc -xj -debug altering.cob
This is the start of a changing story
The story progresses
The story ends, happily ever after

prompt$ COB_SET_TRACE=Y ./altering
Source:     'altering.cob'
Program-Id: altering         Entry:     altering              Line: 8
Program-Id: altering         Section:   main                  Line: 8
Program-Id: altering         Paragraph: contrived             Line: 11
Program-Id: altering         Statement: ALTER                 Line: 12
Program-Id: altering         Statement: GO TO                 Line: 13
Program-Id: altering         Paragraph: story                 Line: 17
Program-Id: altering         Paragraph: beginning             Line: 22
Program-Id: altering         Statement: ALTER                 Line: 23
Program-Id: altering         Statement: DISPLAY               Line: 24
This is the start of a changing story
Program-Id: altering         Statement: GO TO                 Line: 25
Program-Id: altering         Paragraph: story                 Line: 17
Program-Id: altering         Paragraph: middle                Line: 29
Program-Id: altering         Statement: ALTER                 Line: 30
Program-Id: altering         Statement: DISPLAY               Line: 31
The story progresses
Program-Id: altering         Statement: GO TO                 Line: 32
Program-Id: altering         Paragraph: story                 Line: 17
Program-Id: altering         Paragraph: ending                Line: 36
Program-Id: altering         Statement: DISPLAY               Line: 37
The story ends, happily ever after
Program-Id: altering         Statement: EXIT PROGRAM          Line: 41
Program-Id: altering         Exit:      altering
prompt$
```

http://open-cobol.sourceforge.net/faq/index.html#alter .

ALTER : https://riptutorial.com/ko/cobol/topic/5584/alter-

6: CALL

COBOL CALL .



Examples

CALL

COBOL . GnuCOBOL .

```
CALL "subprogram" USING a b c *> run a (possibly static linked) sub program
                              *> passing three fields

CALL some-prog USING a b c    *> some-prog is a PIC X item and can be changed
                              *> at run-time to do a dynamic lookup
```

. *(,)* :

```
CALL STATIC "subprogram" USING a b c
```

COBOL `BY REFERENCE` (: - `sticky`), `BY CONTENT` (REFERENCE) `BY VALUE` .

```
CALL "calculation" USING BY REFERENCE a BY VALUE b BY CONTENT c RETURNING d
    ON EXCEPTION DISPLAY 'No linkage to "calculation"' UPON SYSERR
END-CALL
```

COBOL `BY REFERENCE` `BY VALUE` . , COBOL . `BY VALUE` .

http://open-cobol.sourceforge.net/faq/index.html#call .

CALL COBOL . "" .

IBM COBOL "" . "" . .

ILBOWAT0 COBOL . BXP1SLP BXP4SLP Unix System Services (USS) . "sleep".

IBM ((LE)) CEE3DLY LE z / OS .

ILBOWAT0 (40) , . CEE3DLY BXP1SLP .

(FTP NDM) / .

OS / VS COBOL COBOL .

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID.  SLEEPYTM.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       01  WAIT-PARM.
           05  WAIT-TIME            PIC S9(8) COMP VALUE 90.
           05  WAIT-RESPONSE        PIC S9(8) COMP VALUE 0.
           05  WAIT-PROGRAM-24BIT   PIC  X(8)      VALUE 'ILBOWAT0'.
           05  WAIT-PROGRAM-31BIT   PIC  X(8)      VALUE 'BPX1SLP '.
           05  WAIT-PROGRAM-64BIT   PIC  X(8)      VALUE 'BPX4SLP '.
```

```
       PROCEDURE DIVISION.
       GENESIS.
           DISPLAY 'START CALLING WAIT PROGRAM'
           CALL WAIT-PROGRAM-24BIT USING WAIT-TIME WAIT-RESPONSE
           DISPLAY 'END   CALLING WAIT PROGRAM'
           GOBACK
PERIOD     .
```

Microfocus "SleepEx"API . ;

```
environment division.
special-names.
    call-convention 74 is winAPI.
         :
         :
01  wSleep-time              pic 9(8) comp-5.
01  wSleep-ok                pic 9(8) comp-5.
         :
         :
move 10000 to wSleep-time  *>10seconds
call winAPI "SleepEx" using by value wSleep-time
                        by value 0 size 4
             returning wSleep-ok
end-call.
```

z / OS

24/31 64 CEE3DLY . CICS .

:

```
    IDENTIFICATION DIVISION.
    PROGRAM-ID.  SLEEPYTM.
    ENVIRONMENT DIVISION.
    DATA DIVISION.
    WORKING-STORAGE SECTION.
    01  WAIT-PARM.
      05  WAIT-SECS            PIC S9(8) COMP VALUE 90.
      05  WAIT-FC              PIC X(12).

    PROCEDURE DIVISION.

      CALL CEE3DLY USING WAIT-SECS WAIT-FC

      GOBACK.
```

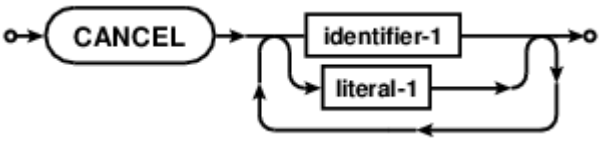
.

IBM -

CALL : https://riptutorial.com/ko/cobol/topic/5601/call-

7: CANCEL

CANCEL .



Examples

CANCEL

```
CALL "submodule"
CALL "submodule"

CANCEL "submodule"
CALL "submodule"
```

`submodule` `CALL` . `CALL` X `CALL` `CALL` .

COBOL ( ) / ( ) . .

http://open-cobol.sourceforge.net/faq/index.html#cancel .

CANCEL : https://riptutorial.com/ko/cobol/topic/5600/cancel-

8: cobol ?

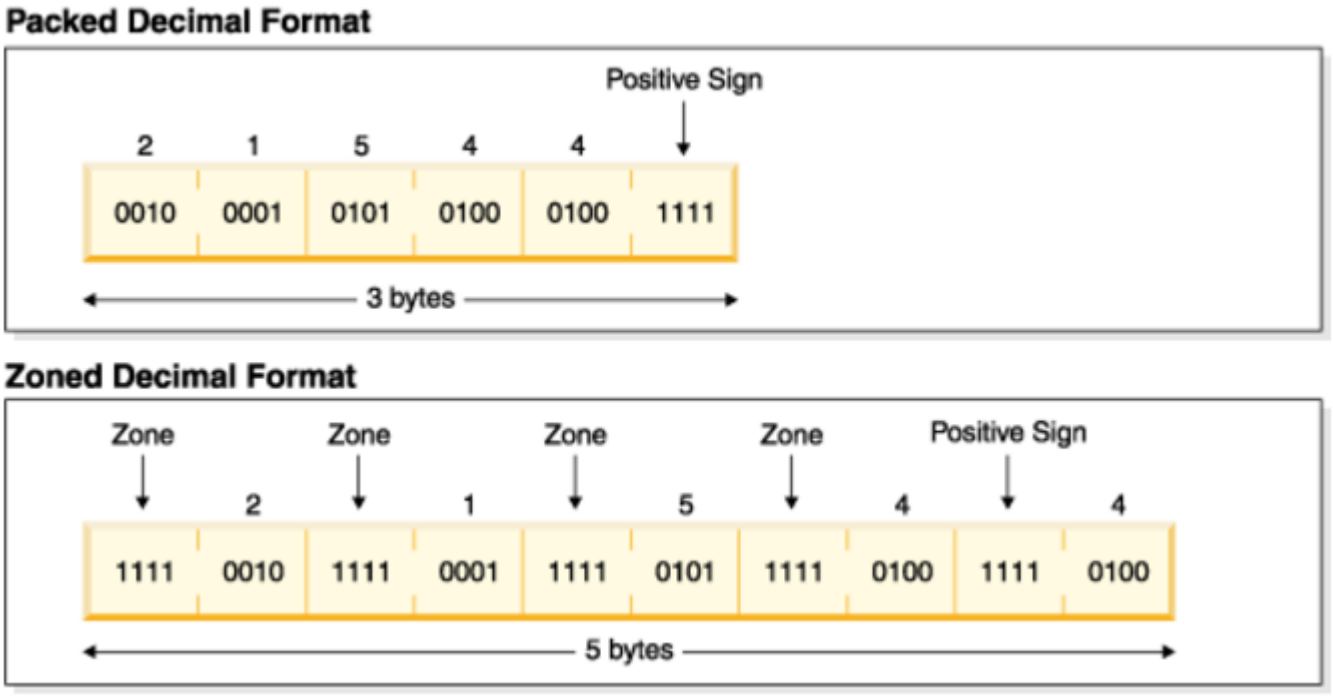
COBOL . COMP-1, COMP-2 COMP-3 3 . COMP-3. "COMP" .

Examples

COMP-3

COMP-3 10 . (packed-decimal) ( ) . ( ) .

"Zoned decimal format" COBOL .




```
01 WS-NUM PIC 9(5) USAGE IS COMP-3 VALUE 21544.
```

.

comp, comp-1 ... comp-5 .

```
Format     Normal Implementation

Comp       Big endian binary integer
Comp-1     4 byte floating point
Comp-2     8 byte floating point

Comp-3     Packed decimal 123 is stored as x'123c'

Comp-5     Binary Integer optermised for performance.
           Big Endian on the Mainframe, Little Endian on Intel Hardware
```

Ibm Comp, Comp-4, Comp-5 2,4,8 . GNU Cobolo 1,2,4,8 .

Comp-1, Comp-2 .

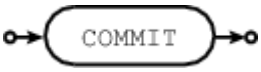
```
03 Floating-Field      Comp-1.
03 Double-Field        Comp-2
```

Comp :

```
03 Big-Endian           Pic S9(4) Comp.
03 Packed-Decimal       Pic S9(5) Comp.
```

cobol ? : https://riptutorial.com/ko/cobol/topic/10873/cobol----

9: COMMIT



I / O .

`ROLLBACK` COBOL .

Examples

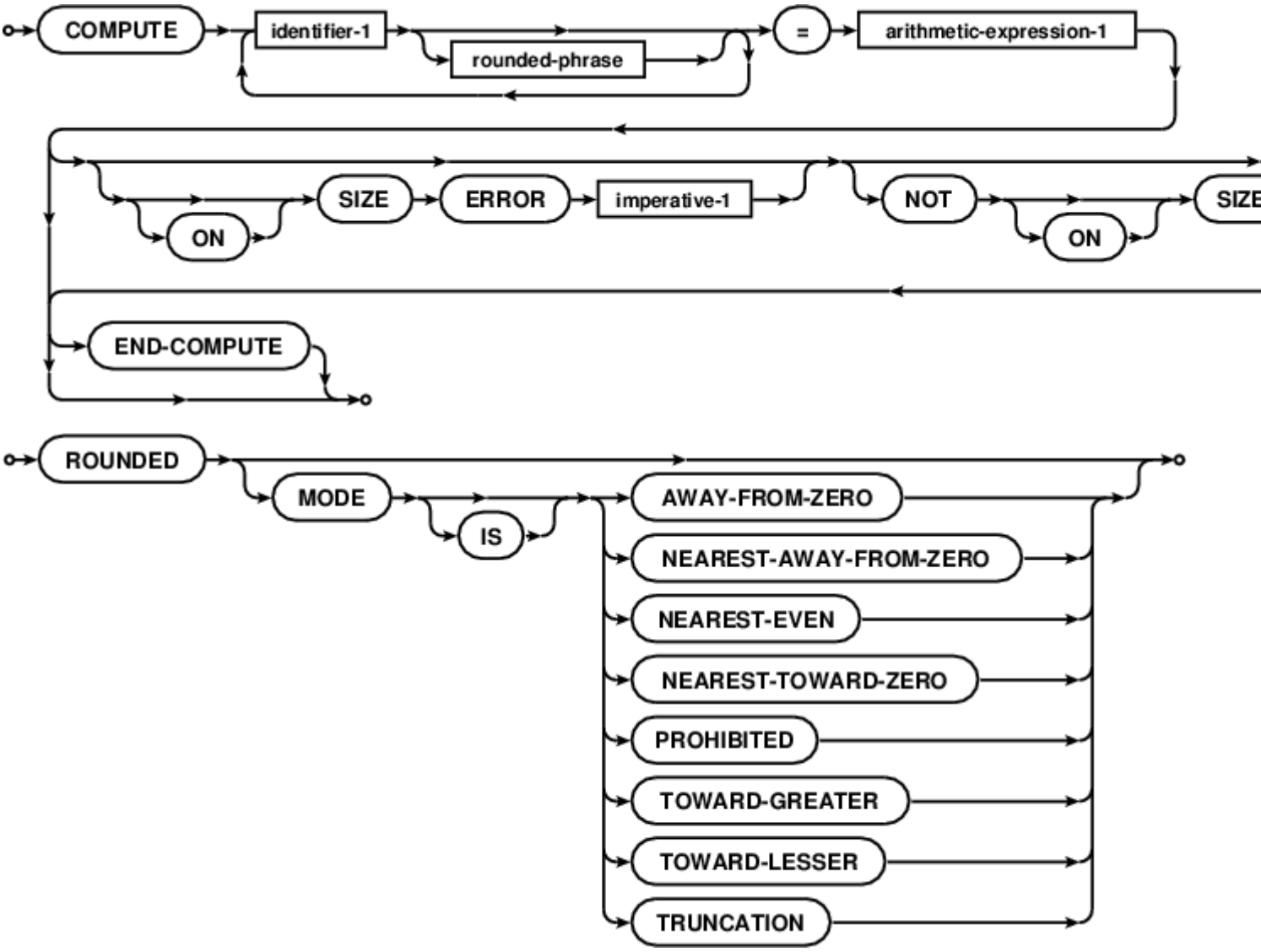
COMMIT

```
WRITE record
COMMIT
```

COMMIT : https://riptutorial.com/ko/cobol/topic/6357/commit-

10: COMPUTE

COMPUTE .



Examples

: .

```
COMPUTE answer = 3*var-1
```

var-1 var - 1 var-1 .

```
COMPUTE answer = 3 * var - 1
```

, .

COMPUTE : https://riptutorial.com/ko/cobol/topic/6726/compute-

11: CONTINUE

CONTINUE . , IF / THEN / ELSE .

? .

```
CALL "CBL_OC_DUMP" USING structure ON EXCEPTION CONTINUE END-CALL
```

CBL_OC_DUMP .* . . . EXCEPTION CONTINUE .

Examples

. COBOL NOT (var NOT = value OR other-value) .

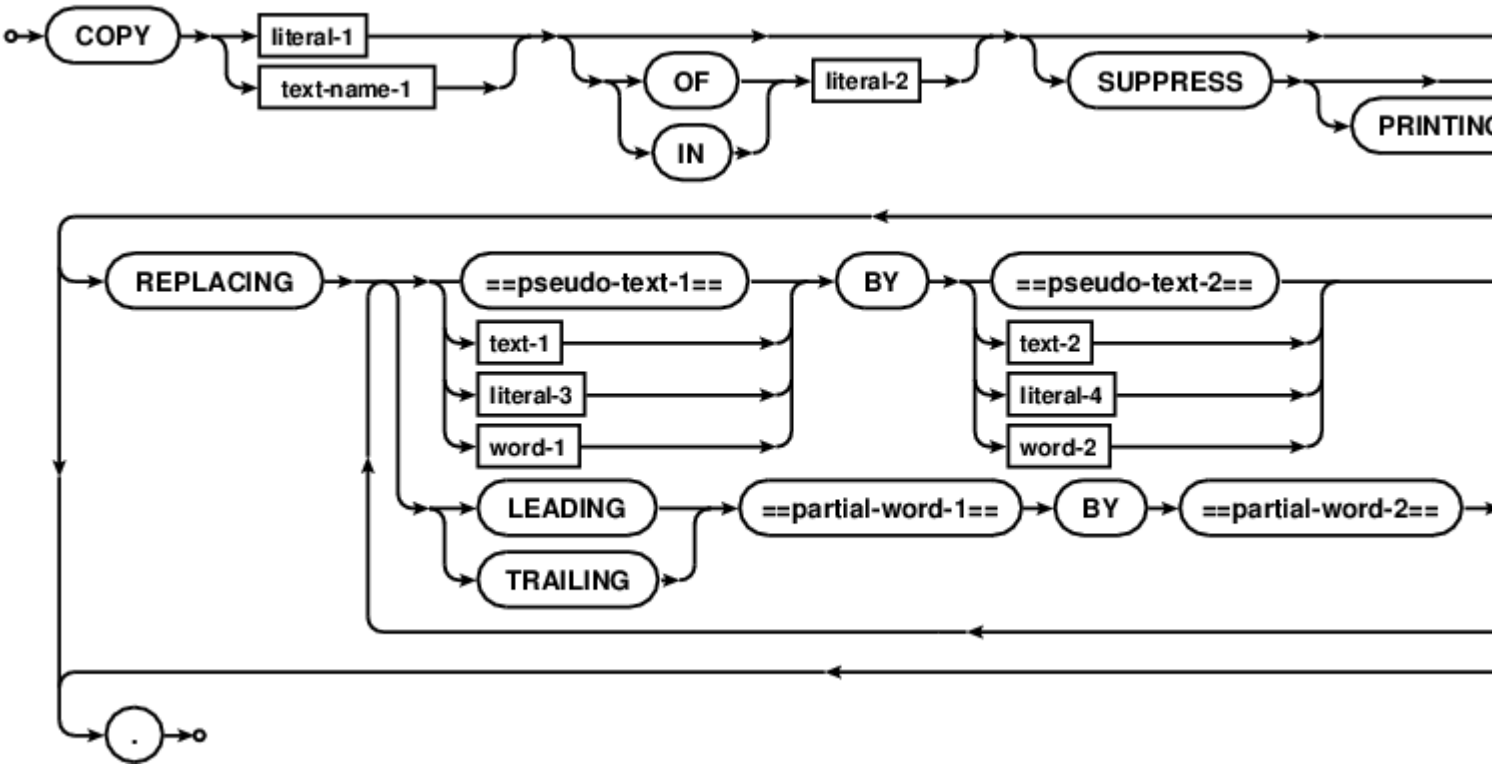
```
if action-flag = "C" or "R" or "U" or "D"
    continue
else
    display "invalid action-code" upon syserr
    perform report-exception
    exit section
end-if
```

CONTINUE : https://riptutorial.com/ko/cobol/topic/6981/continue-

12: COPY

COBOL C `#include` *COBOL 10* .

COBOL (`PERFORM`) DRY . , . copybooks . COPY . .



Examples

COPY .

- .

```
FD important-file.
01 file-record.
   COPY record-layout.

DATA DIVISION.
01 memory-record.
   COPY record-layout.

PROCEDURE DIVISION.
   ...
   COPY record-move.
   ...
   COPY record-move.
```

program-two.

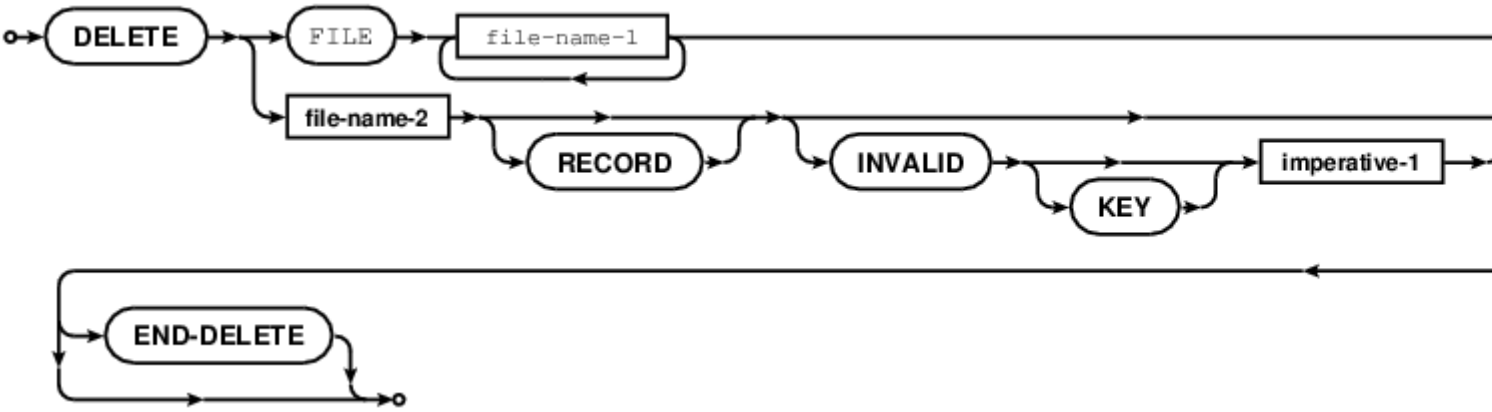
```
   DATA DIVISION.
```

```
01 print-record.
   COPY record-layout.
...

PROCEDURE DIVISION.
...
print-line.
    COPY record-move.
```

COPY : https://riptutorial.com/ko/cobol/topic/6982/copy-

13: DELETE

DELETE . DELETE FILE FD ( ).



Examples

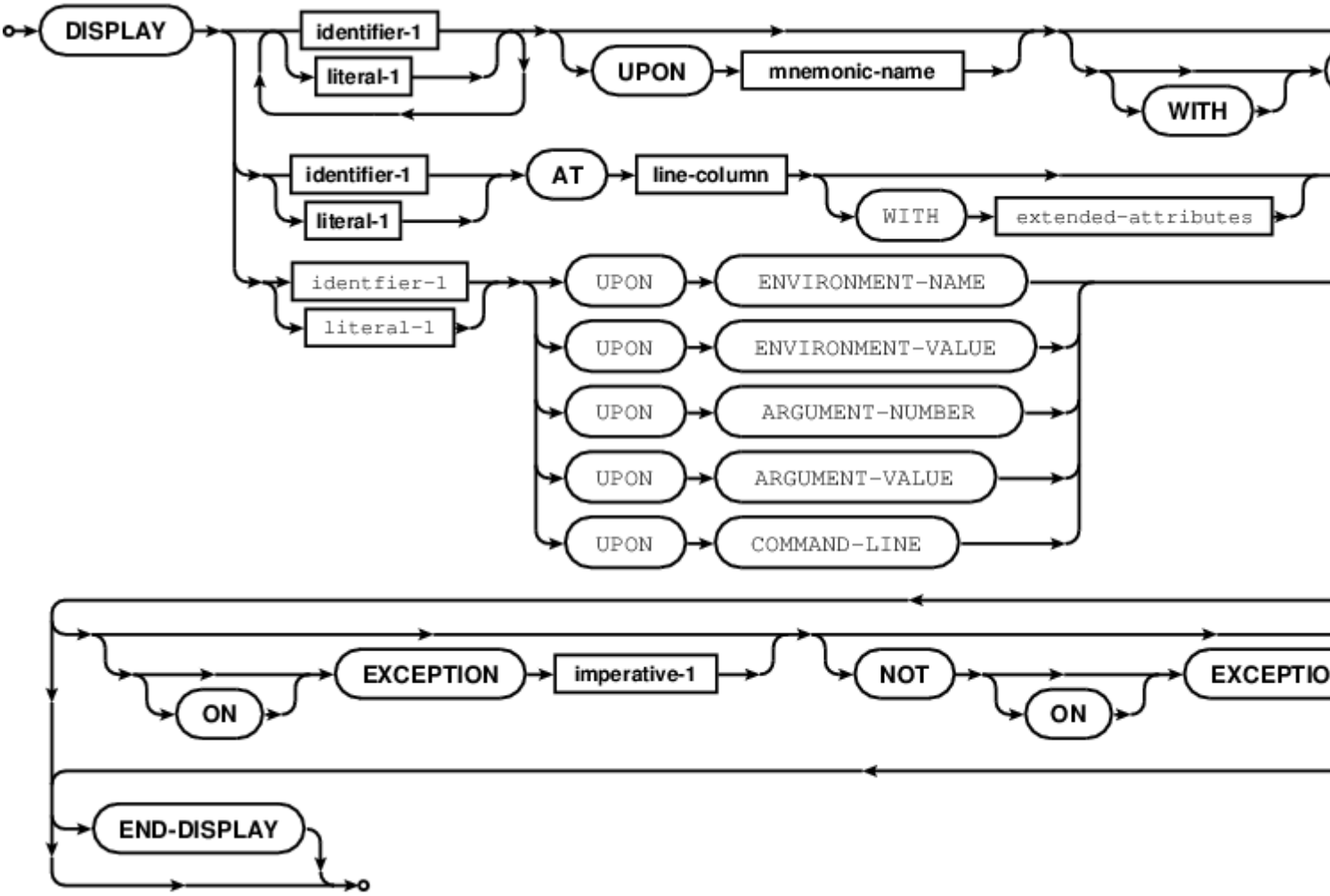
,

```
    identification division.
    program-id. deleting.

    environment division.
    configuration section.

    input-output section.
    file-control.
        select optional indexed-file
        assign to "indexed-file.dat"
        status is indexing-status
        organization is indexed
        access mode is dynamic
        record key is keyfield
        alternate record key is altkey with duplicates
        .

    ...

    procedure division.

    move "abcdef" to keyfield

    *> Delete a record by index
    delete indexed-file record
       invalid key
           display "No delete of " keyfield end-display
       not invalid key
           display "Record " keyfield " removed" end-display
    end-delete

    perform check-delete-status

    ...
```

DELETE : https://riptutorial.com/ko/cobol/topic/7063/delete-

14: DISPLAY

DISPLAY . DISPLAY UPON device SCREEN . COBOL DISPLAY UPON .



Examples

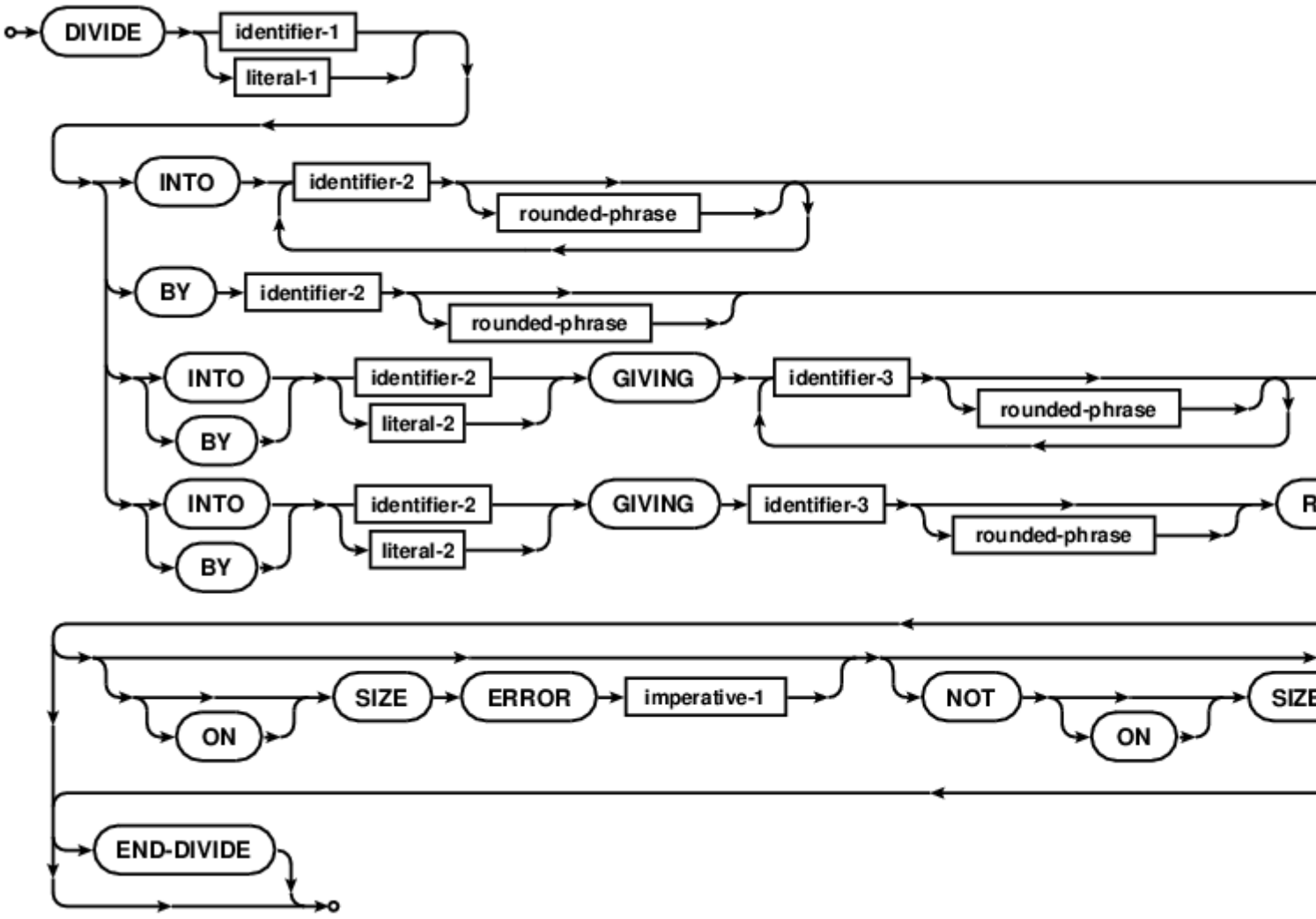
```
DISPLAY "An error occurred with " tracked-resource UPON SYSERR

DISPLAY A, B, C UPON CONSOLE

DISPLAY group-data UPON user-device
    ON EXCEPTION
        WRITE device-exception-notice
    NOT ON EXCEPTION
        WRITE device-usage-log
END-DISPLAY
```

UPON CONSOLE . DISPLAY COBOL , COBOL , .

DISPLAY : https://riptutorial.com/ko/cobol/topic/7082/display-

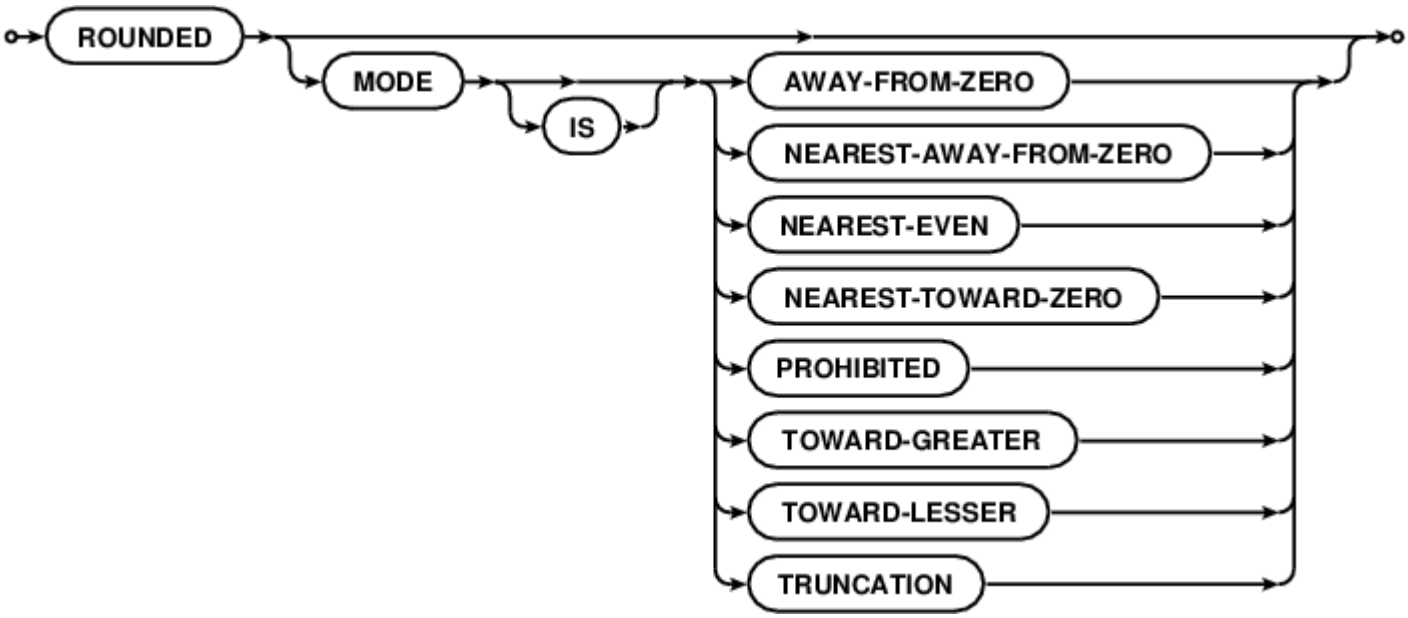
15: DIVIDE

COBOL `DIVIDE` `DIVIDE` , .



`ROUNDED` :

`TRUNCATION` . `ROUNDED` `NEAREST-TOWARD-ZERO` () . *Banker* `NEAREST-EVEN` .



Examples

DIVIDE

```
DIVIDE a INTO b c d
```

```
b , c  d  b/a , c/a  d/a .
```

```
DIVIDE a INTO b GIVING c
```

```
c  b/a  .
```

```
DIVIDE a BY b GIVING c
```

```
c  a/b  .
```

```
DIVIDE a INTO b GIVING q REMAINDER r
```

```
q  r  b/a   b/a
```

```
DIVIDE a BY b GIVING q REMAINDER r
```

```
q  r  b/a   b/a
```

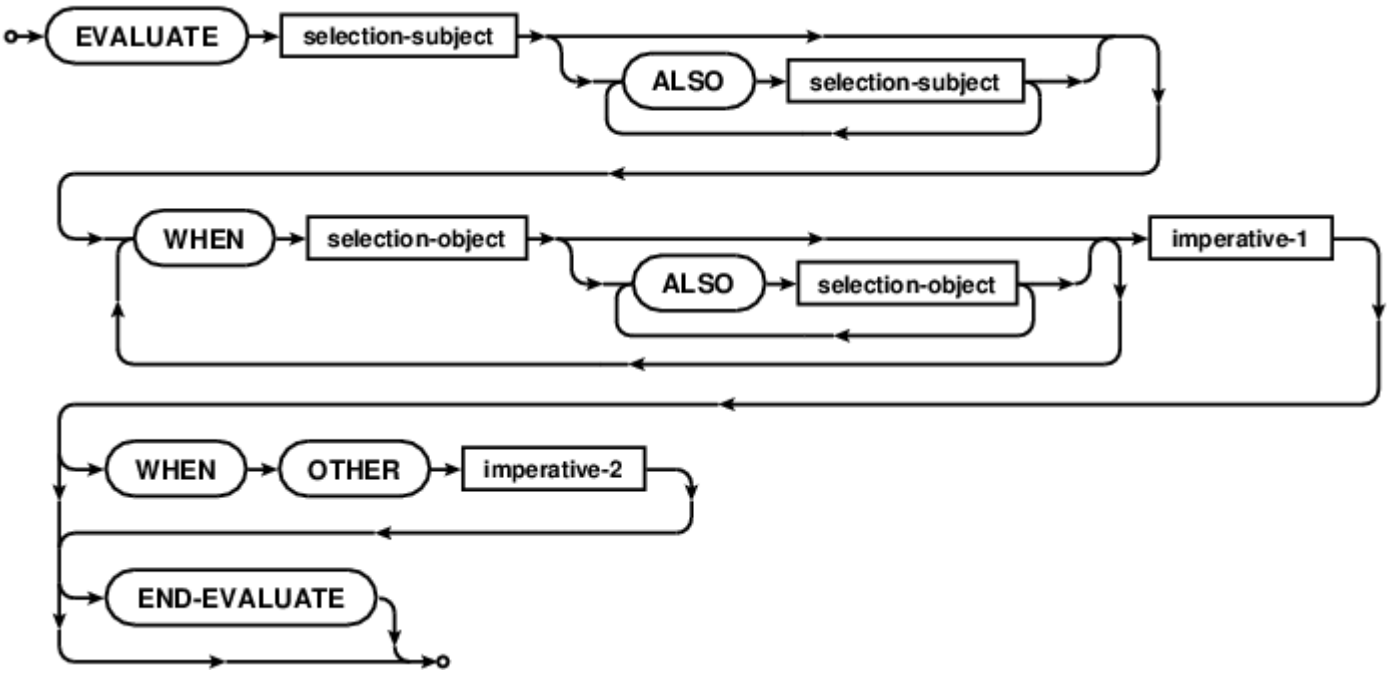
```
DIVIDE   ROUNDED MODE IS  .
```

```
DIVIDE  ON SIZE ERROR  NOT ON SIZE ERROR         .
```

DIVIDE : https://riptutorial.com/ko/cobol/topic/7081/divide-

16: EVALUATE

```
EVALUATE  , ,   .
```



Examples

3

```
EVALUATE a ALSO b ALSO TRUE
    WHEN 1 ALSO 1 THRU 9 ALSO c EQUAL 1 PERFORM all-life
    WHEN 2 ALSO 1 THRU 9 ALSO c EQUAL 2 PERFORM life
    WHEN 3 THRU 9 ALSO 1 ALSO c EQUAL 9 PERFORM disability
    WHEN OTHER PERFORM invalid
END-EVALUATE
```

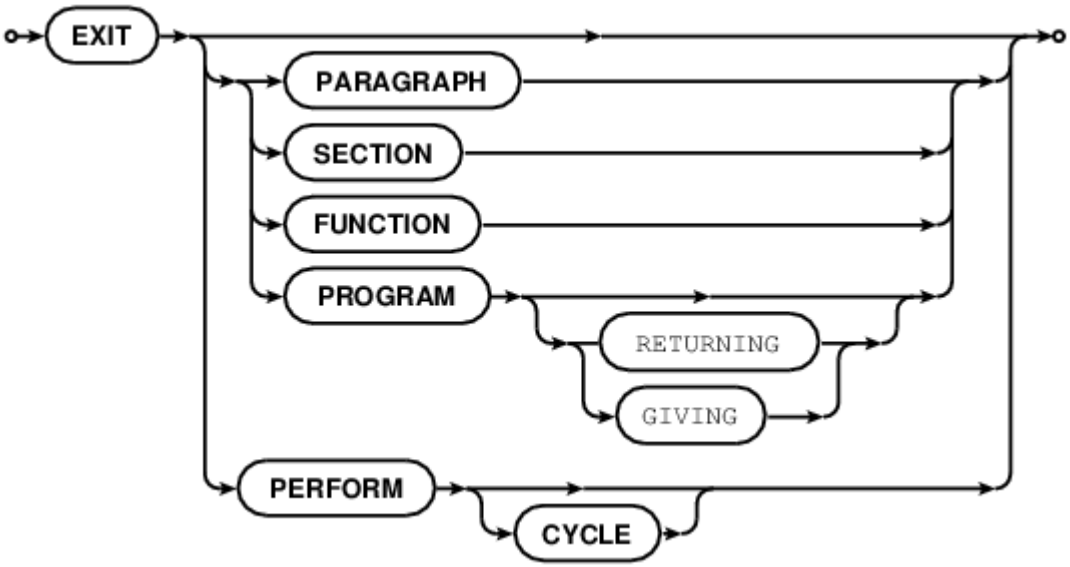
EVALUATE : https://riptutorial.com/ko/cobol/topic/7083/evaluate-

17: EXIT

COBOL `EXIT` .

`EXIT` .

- `EXIT` .
- `EXIT PARAGRAPH` , `EXIT SECTION` .
- `EXIT FUNCTION` , `EXIT METHOD` , `EXIT PROGRAM` .
- `EXIT PERFORM` .
- `EXIT PERFORM CYCLE` .



Examples

EXIT

```
PERFORM VARYING counter FROM 1 BY 1 UNTIL counter > 10
    IF debug-override THEN EXIT PERFORM
    IF counter = 5 THEN EXIT PERFORM CYCLE
    PERFORM some-miracle
END-PERFORM
```

EXIT : https://riptutorial.com/ko/cobol/topic/7084/exit-

18: GENERATE

COBOL `GENERATE` .

Examples

```
GENERATE detail-line
```

GENERATE : https://riptutorial.com/ko/cobol/topic/7161/generate-

19: GNU / Linux GnuCOBOL

Examples

GNU /

GNU / Linux `GnuCOBOL` . `GnuCOBOL` `OpenCOBOL` GNU . `open-cobol` (2016 8).

Fedora RPM

```
sudo yum install open-cobol
```

, APT

```
sudo apt install open-cobol
```

1.1 GnuCOBOL .

(https://sourceforge.net/projects/open-cobol/ SourceForge) .

- AC ; `build-essential` ( )
- BerkeleyDB BerkelyDB . `libdb` , `libdb-dev` ( )
- GNU ; `libgmp` , `libgmp-dev`
- `curses` . `ncurses` , `ncurses-dev`
- `gnucobol-1.1.tar.gz` (, `gnucobol-2.0.tar.gz`)
- `GNU Autoconf` .

:

```
prompt$ tar xvf gnucobol.tar.gz
prompt$ cd gnucobol
```

.

```
prompt$ ./configure --help
```

```
prompt$ ./configure
prompt$ make
```

.

```
prompt$ make check
```

```
prompt$ make checkall
```

(`make check`) NIST COBOL85 verification suite (`make checkall`) . OpenCOBOL 1.1 9100 NIST , 9700 . *NIST COBOL85* . *COBOL* *COBOL 2002* *COBOL 2014* *NIST* .

500 .

```
prompt$ sudo make install
```

`sudo make install` `./configure .` `/usr/local .`

```
prompt$ sudo ldconfig
```

`ld` .

`cobc` .

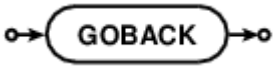
`cobc --help` `info open-cobol` , `info open-cobol` (`info gnucobol`) http://open-cobol.sourceforge.net/ Programmer 's Guide 1200+ FAQ .

, GnuCOBOL , SourceForge `Help getting started` , SourceForge .

GNU / Linux GnuCOBOL : https://riptutorial.com/ko/cobol/topic/5446/gnu---linux-gnucobol-

20: GOBACK

COBOL `GOBACK` . `EXIT PROGRAM STOP RUN` `GOBACK` . "main" `GOBACK` . `GOBACK` .



Examples

GOBACK

```
identification division.
program-id. subprog.
procedure division.
display "in subprog"
goback.

...

call "subprog"
goback.
```

`GOBACK` subprog . `GOBACK` .

GOBACK : https://riptutorial.com/ko/cobol/topic/7173/goback-

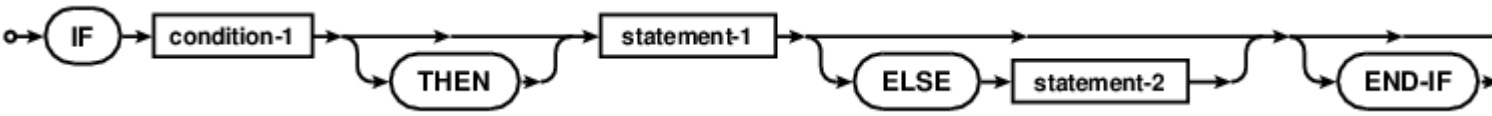
21: IF

. . COBOL ( ) .

```
IF A = 1 OR 2 ...
```

~ .

```
IF A = 1 OR A = 2 ...
```



Examples

IF

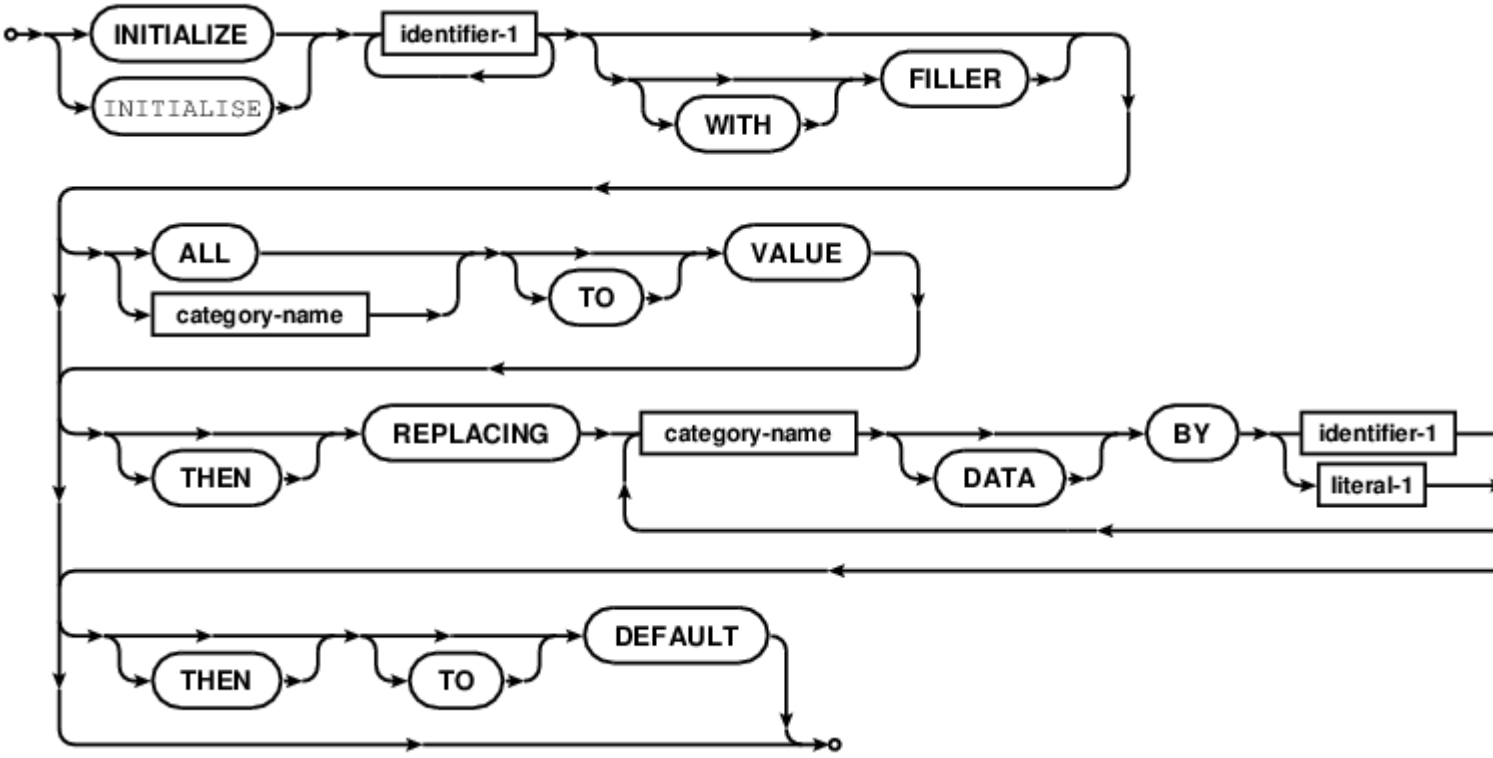
```
IF A = 1 OR 2 THEN
    perform miracles
END-IF

IF A = 1 OR 2 AND B = 1 THEN
    perform rites-of-passage
ELSE
    perform song-and-dance
END-IF
```

IF END-IF . . IF . , IF .

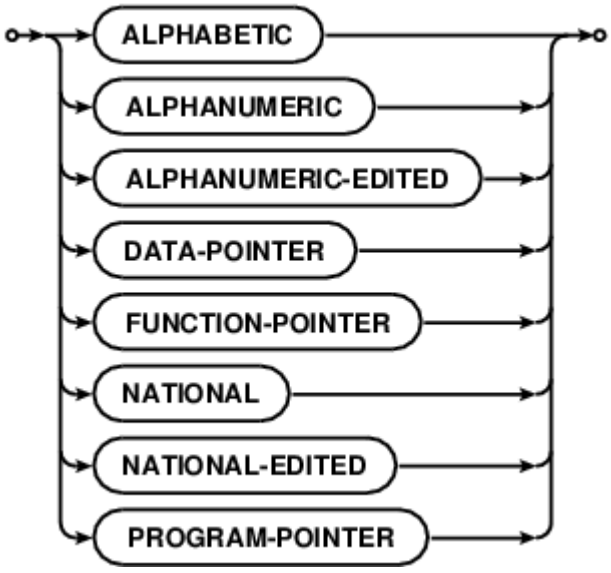
IF : https://riptutorial.com/ko/cobol/topic/7174/if-

22: INITIALIZE

```
INITIALIZE .   .
```




```
category-name  .
```



Examples

INITIALIZE

```
01  fillertest.
    03 fillertest-1 PIC 9(10) value 2222222222.
```

```
    03 filler       PIC X     value '|'.
    03 fillertest-2 PIC X(10) value all 'A'.
    03 filler       PIC 9(03) value 111.
    03 filler       PIC X     value '.'.

INITIALIZE fillertest

INITIALIZE fillertest REPLACING NUMERIC BY 9

INITIALIZE fillertest REPLACING ALPHANUMERIC BY 'X'

INITIALIZE fillertest REPLACING ALPHANUMERIC BY ALL 'X'

INITIALIZE fillertest WITH FILLER

INITIALIZE fillertext ALL TO VALUE
```

:

```
fillertest on start:
2222222222|AAAAAAAAAA111.
fillertest after initialize:
0000000000|          111.
fillertest after initialize replacing numeric by 9:
0000000009|          111.
fillertest after initialize replacing alphanumeric by "X":
0000000009|X         111.
fillertest after initialize replacing alphanumeric by all "X":
0000000009|XXXXXXXXXX111.
fillertest after initialize with filler:
0000000000           000
fillertest after initialize all to value:
2222222222|AAAAAAAAAA111.
```

INITIALIZE : https://riptutorial.com/ko/cobol/topic/7179/initialize-

23: INITIATE

```
INITIATE  Report Writer  .   DATA DIVISION   PROCEDURE DIVISION . GENERATE      .
```

Examples

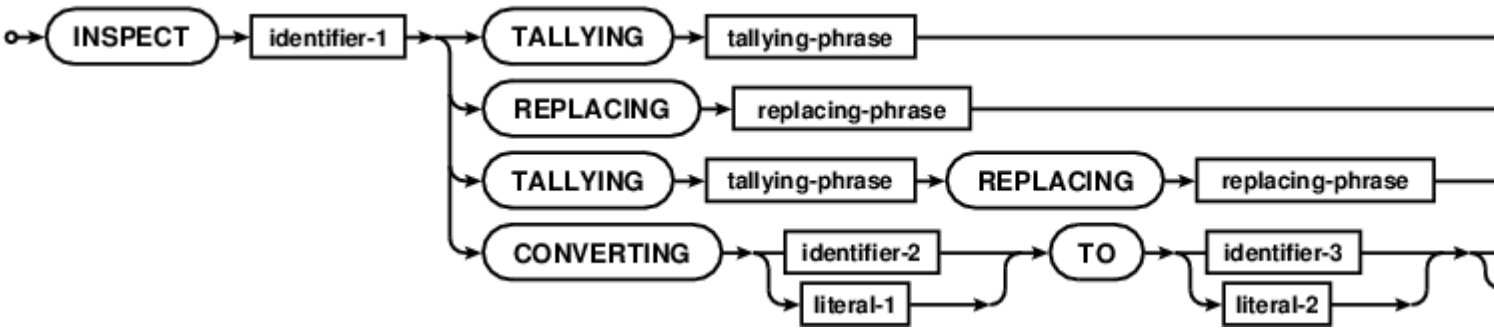
INITIATE

```
INITIATE report-1 report-2
```

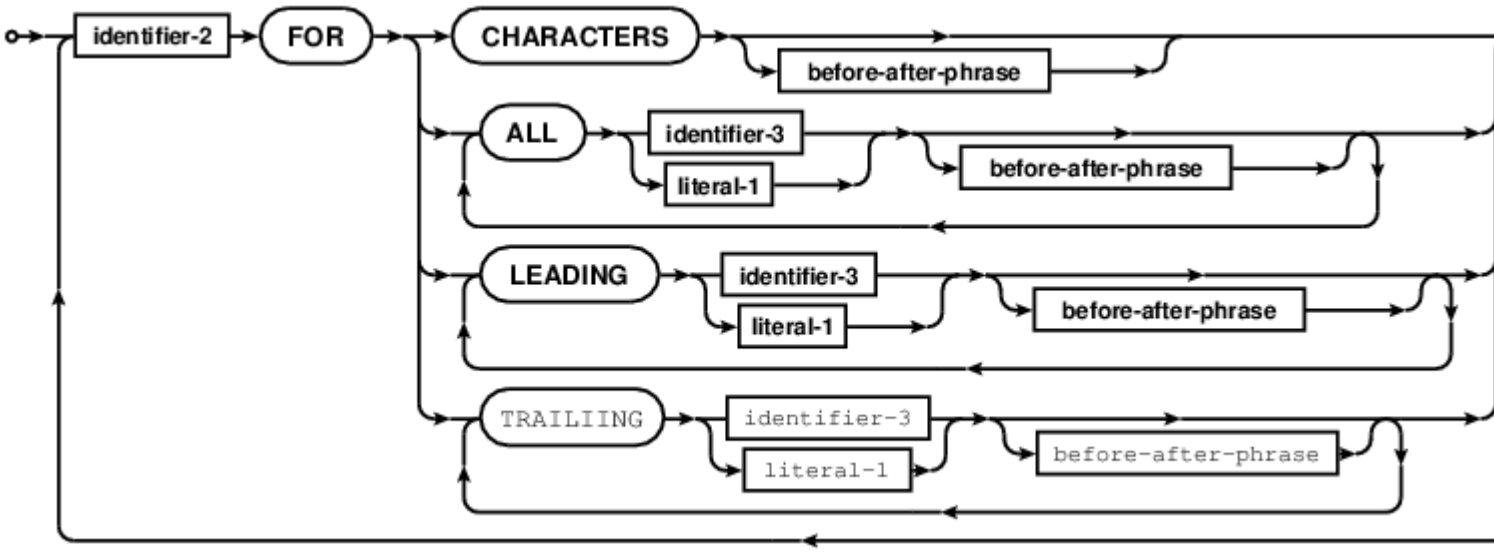
INITIATE : https://riptutorial.com/ko/cobol/topic/7180/initiate-

24: INSPECT

`INSPECT` COBOL .



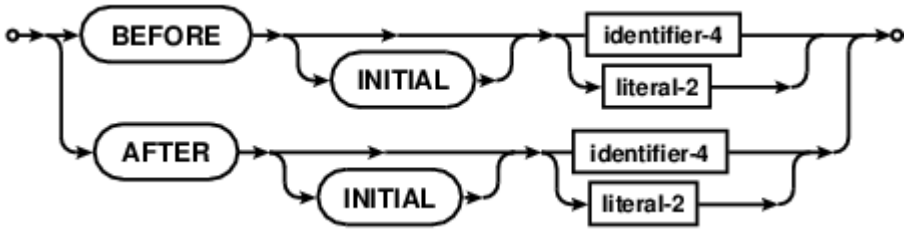
`tallying-phrase` .



`replacing-phrase` .

```
missing image
```

`before-after-phrase` .



Examples

INSPECT

```
GCobol identification division.
       program-id. inspecting.

       data division.
       working-storage section.
       01  ORIGINAL            pic XXXX/XX/XXBXX/XX/XXXXXXX/XX.
       01  DATEREC             pic XXXX/XX/XXBXX/XX/XXXXXXX/XX.

       procedure division.

       move function when-compiled to DATEREC ORIGINAL

       INSPECT DATEREC REPLACING ALL "/" BY ":" AFTER INITIAL SPACE

       display "Formatted function WHEN-COMPILED " ORIGINAL
       display " after INSPECT REPLACING         " DATEREC

       goback.
       end program inspecting.
```

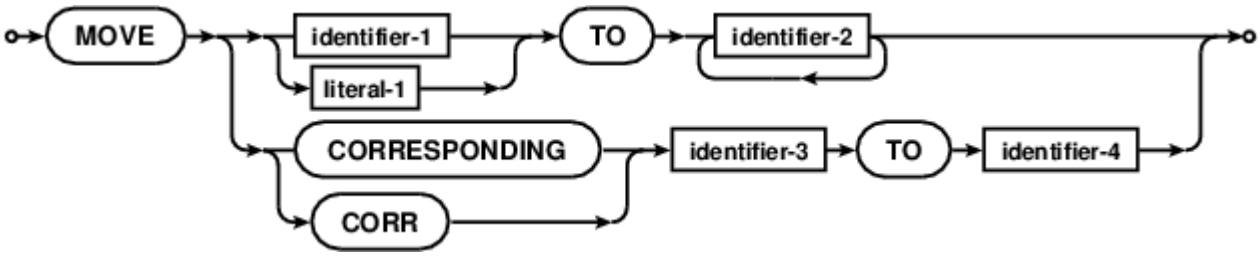
:

```
Formatted function WHEN-COMPILED 2010/03/25 23/05/0900-04/00
 after INSPECT REPLACING         2010/03/25 23:05:0900-04:00
```

INSPECT : https://riptutorial.com/ko/cobol/topic/7182/inspect-

25: MOVE

MOVE COBOL . . COBOL MOVE . . . , 0 (). , . MOVE BINARY PICTURE DISPLAY MOVE .



Examples

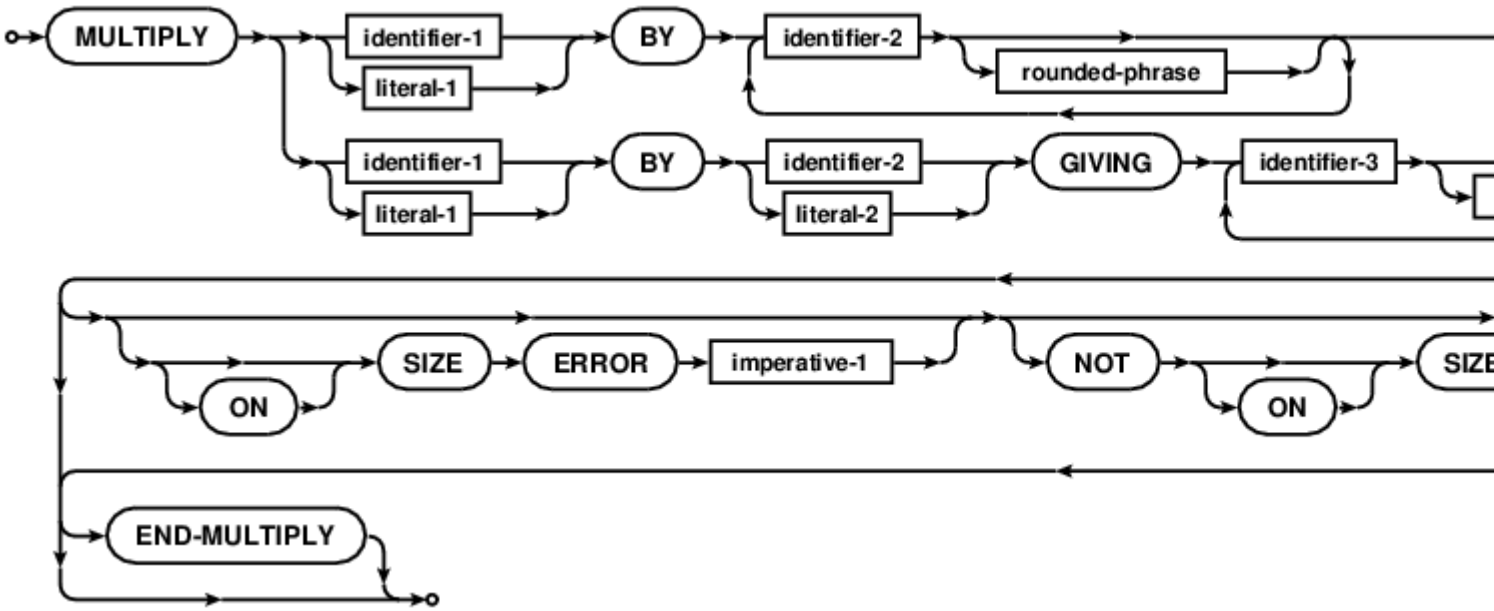
, .

```
01 a PIC 9.
01 b PIC 99.
01 c PIC 999.

01 s PIC X(4).

01 record-group.
   05 field-a PIC 9.
   05 field-b PIC 99.
   05 field-c PIC 999.
01 display-record.
   05 field-a PIC Z.
   05 field-b PIC ZZ.
   05 field-c PIC $Z9.

*> numeric fields are moved left to right
*> a set to 3, b set to 23, c set to 123
MOVE 123 TO a b c

*> moves can also be by matching names within groups
MOVE a TO field-a OF record-group
MOVE b TO field-b OF record-group
MOVE c TO field-c OF record-group
MOVE CORRESPONDING record-group TO display-record

*> character data is moved right to left
*> s will be set to xyzz
MOVE "xyzzy" TO s
```

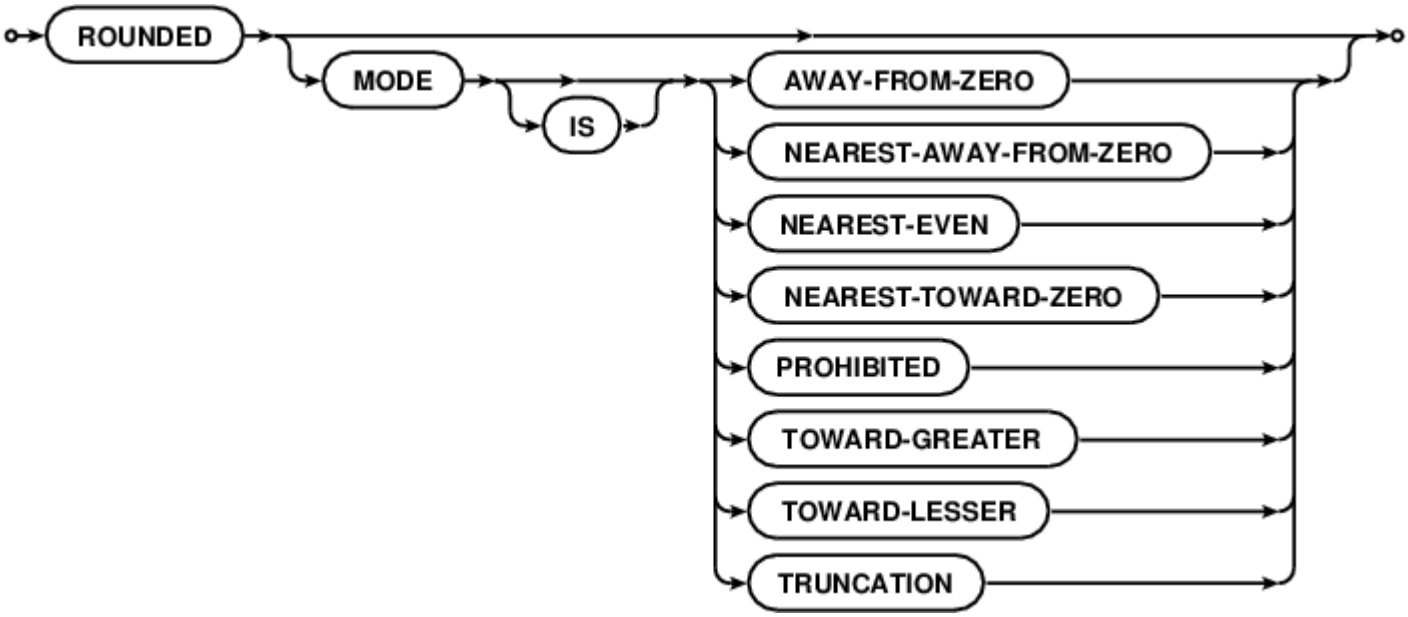
MOVE : https://riptutorial.com/ko/cobol/topic/7263/move-

26: MULTIPLY

`MULTIPLY` .



`rounded-phrase` ?



Examples

MULTIPLY

```
MULTIPLY 5 BY a

MULTIPLY a BY b
    ON SIZE ERROR
        PERFORM error-handling
    NOT ON SIZE ERROR
```

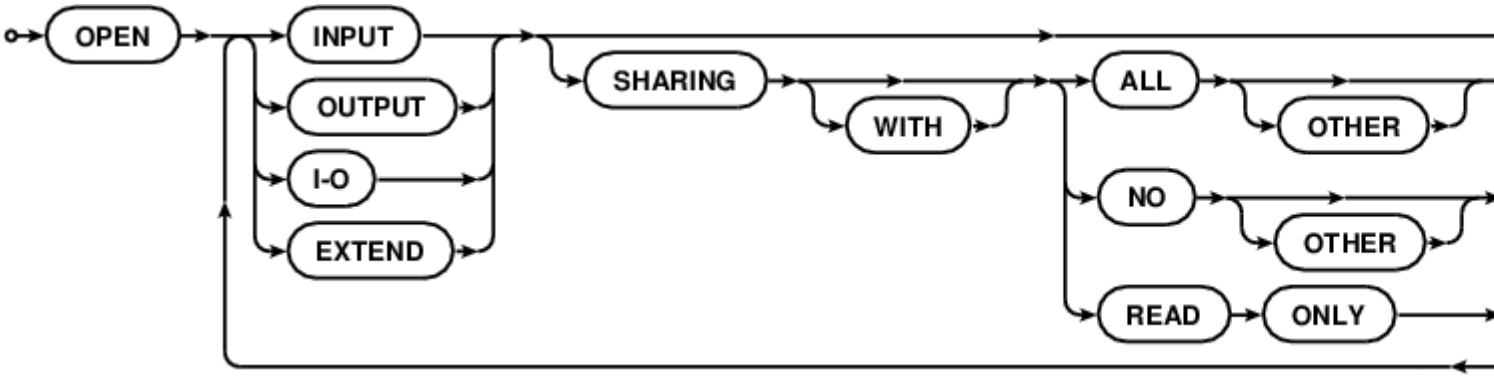
```
        PERFORM who-does-that
END-MULTIPLY

MULTIPLY a BY b GIVING x ROUNDED MODE IS PROHIBITED
                         y ROUNDED MODE IS NEAREST-EVEN
                         z ROUNDED
```

MULTIPLY : https://riptutorial.com/ko/cobol/topic/7264/multiply-

27: OPEN

COBOL OPEN . COBOL ENVIRONMENT DIVISION , FD ( ) . fd INPUT-OUTPUT SECTION FILE-CONTROL SELECT . FILE STATUS .

INPUT , OUTPUT , IO EXTEND .



Examples

LINAGE OPEN

```
COBOL ******************************************************************
      * Example of LINAGE File Descriptor
      * Tectonics: $ cocb -x linage.cob
      *            $ ./linage <filename ["linage.cob"]>
      *            $ cat -n mini-report
      ******************************************************************
       IDENTIFICATION DIVISION.
       PROGRAM-ID. linage-demo.

       ENVIRONMENT DIVISION.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
           select optional data-file assign to file-name
               organization is line sequential
               file status is data-file-status.
           select mini-report assign to "mini-report".

       DATA DIVISION.
       FILE SECTION.
       FD  data-file.
       01  data-record.
           88 endofdata        value high-values.
           02 data-line         pic x(80).
       FD  mini-report
           linage is 16 lines
               with footing at 15
               lines at top 2
               lines at bottom 2.
       01  report-line          pic x(80).

       WORKING-STORAGE SECTION.
       01  command-arguments    pic x(1024).
```

```
01  file-name           pic x(160).
01  data-file-status    pic 99.
01  lc                  pic 99.
01  report-line-blank.
    02 filler           pic x(18) value all "*".
    02 filler           pic x(05) value spaces.
    02 filler           pic x(34)
        VALUE "THIS PAGE INTENTIONALLY LEFT BLANK".
    02 filler           pic x(05) value spaces.
    02 filler           pic x(18) value all "*".
01  report-line-data.
    02 body-tag         pic 9(6).
    02 line-3           pic x(74).
01  report-line-header.
    02 filler           pic x(6) VALUE "PAGE: ".
    02 page-no          pic 9999.
    02 filler           pic x(24).
    02 filler           pic x(5) VALUE " LC: ".
    02 header-tag       pic 9(6).
    02 filler           pic x(23).
    02 filler           pic x(6) VALUE "DATE: ".
    02 page-date        pic x(6).

01  page-count          pic 9999.

PROCEDURE DIVISION.

accept command-arguments from command-line end-accept.
string
    command-arguments delimited by space
    into file-name
end-string.
if file-name equal spaces
    move "linage.cob" to file-name
end-if.

open input data-file.
read data-file
    at end
        display "File: " function trim(file-name) " open error"
        go to early-exit
end-read.

open output mini-report.

write report-line
    from report-line-blank
end-write.

move 1 to page-count.
accept page-date from date end-accept.
move page-count to page-no.
write report-line
    from report-line-header
    after advancing page
end-write.

perform readwrite-loop until endofdata.

display
    "Normal termination, file name: "
```

```
           function trim(file-name)
           " ending status: "
           data-file-status
       close mini-report.

      * Goto considered harmful?  Bah!  :)
       early-exit.
       close data-file.
       exit program.
       stop run.

      ****************************************************************
       readwrite-loop.
       move data-record to report-line-data
       move linage-counter to body-tag
       write report-line from report-line-data
           end-of-page
               add 1 to page-count end-add
               move page-count to page-no
               move linage-counter to header-tag
               write report-line from report-line-header
                   after advancing page
               end-write
       end-write
       read data-file
           at end set endofdata to true
       end-read
       .

      ****************************************************************
      * Commentary
      * LINAGE is set at a 20 line logical page
      *  16 body lines
      *   2 top lines
      *   A footer line at 15 (inside the body count)
      *   2 bottom lines
      * Build with:
      * $ cobc -x -Wall -Wtruncate linage.cob
      * Evaluate with:
      * $ ./linage
      * This will read in linage.cob and produce a useless mini-report
      * $ cat -n mini-report
      ****************************************************************
       END PROGRAM linage-demo.
```

OPEN : https://riptutorial.com/ko/cobol/topic/7288/open-

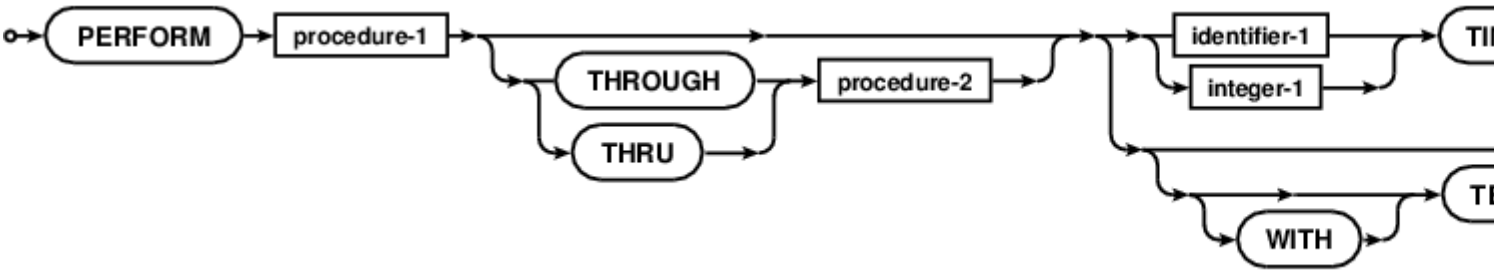
28: PERFORM

PERFORM . PERFORM PERFORM .

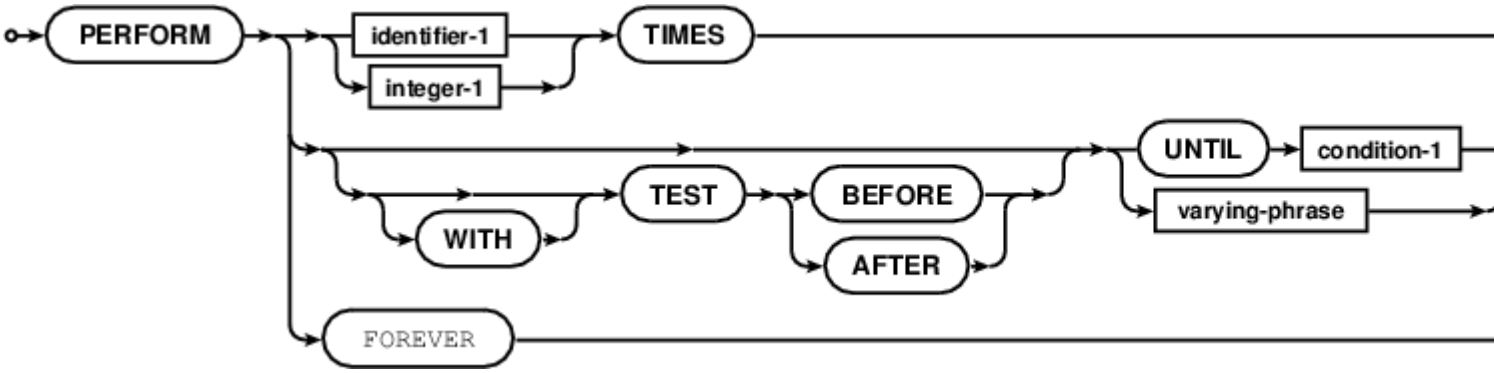
VARYING AFTER BEFORE () AFTER .

THRU procedure-1 procedure-2 . *THRU* , PERFORM SECTION THRU . PERFORM THRU , THRU .

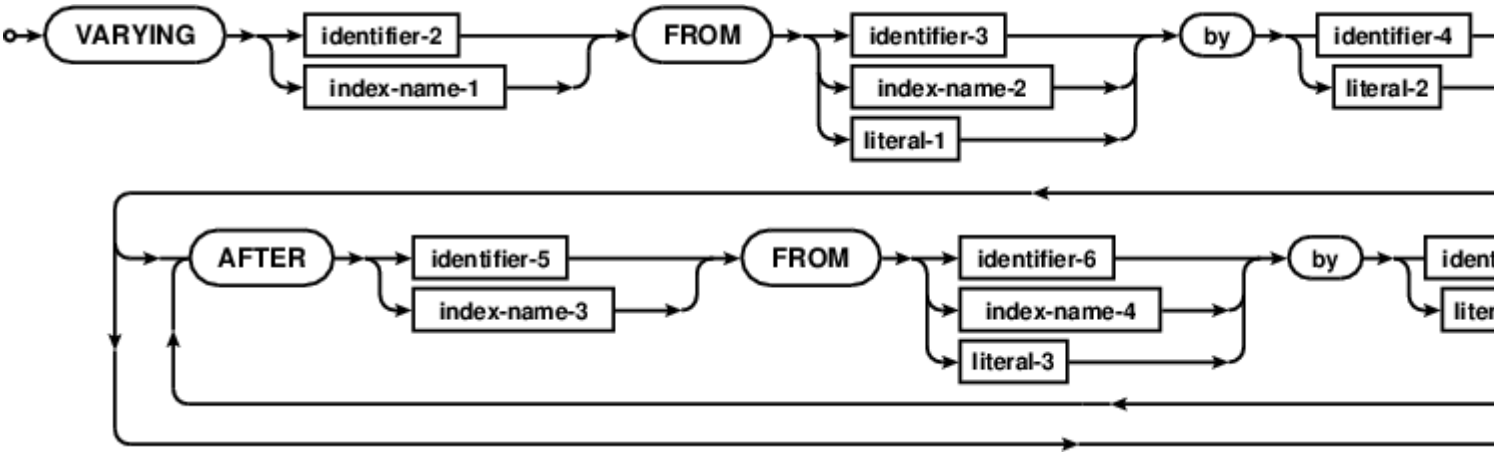
:



:



varying-phrase :



Examples

PERFORM VARYING

```
PERFORM VARYING TALLY FROM 1 BY 1 UNTIL TALLY > 5
    DISPLAY TALLY
END-PERFORM
```

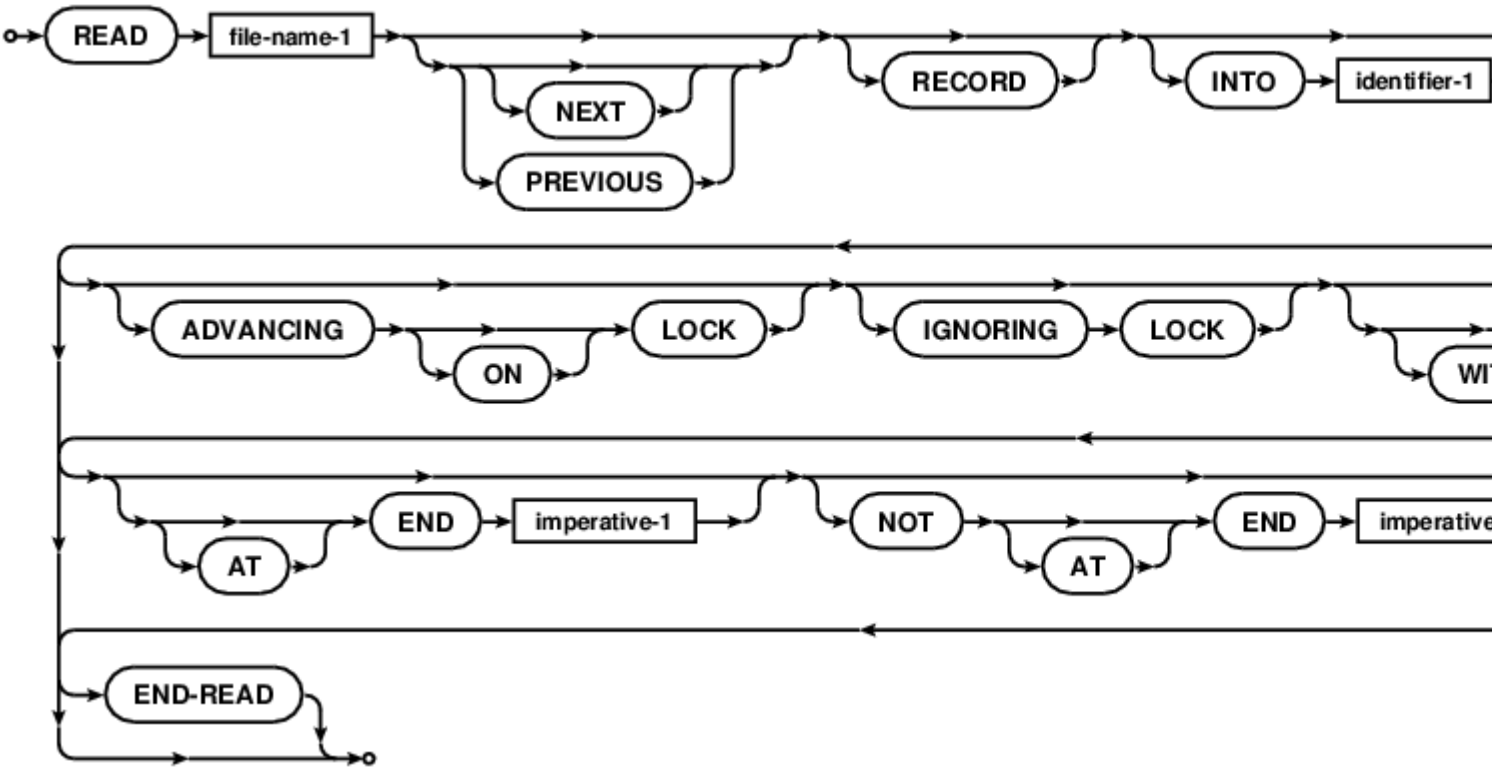
```
PERFORM some-paragraph
```

PERFORM : https://riptutorial.com/ko/cobol/topic/7334/perform-

29: READ

`READ` COBOL . . , `AT END` `FILE STATUS` .

COBOL " ", " ", `READ` (FD) .



Examples

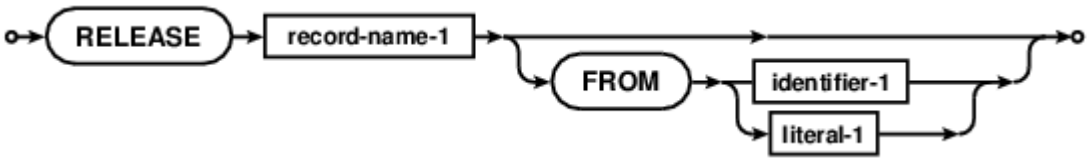
FD

```
READ data-file
```

READ : https://riptutorial.com/ko/cobol/topic/7336/read-

30: RELEASE

RELEASE COBOL SORT .



Examples

SORT INPUT PROCEDURE .

. ALPHABET . A a . . SORT INPUT PROCEDURE RELEASE . OUTPUT PROCEDURE SORT RETURN .

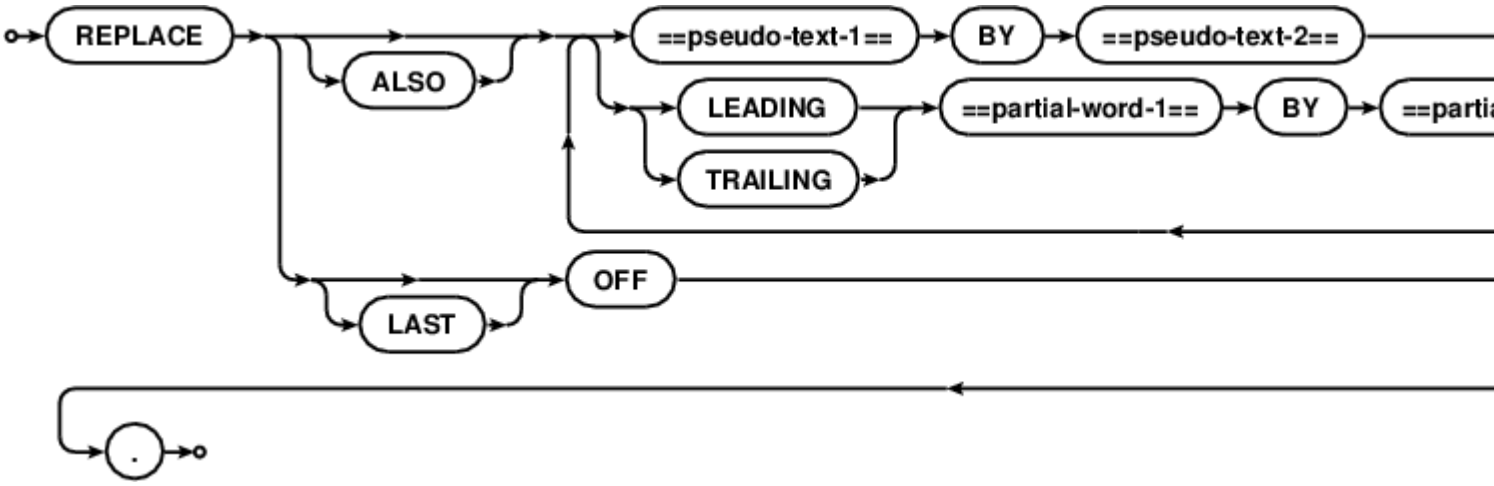
```
GCobol >>SOURCE FORMAT IS FIXED
      ************************************************************
      * Purpose:   A GnuCOBOL SORT verb example
      * Tectonics: cobc -x sorting.cob
      *     ./sorting <input >output
      *   or simply
      *     ./sorting
      *   for keyboard and screen demos
      ************************************************************
       identification division.
       program-id. sorting.

       environment division.
       configuration section.
      * This sets up a sort order lower/upper except for "A" and "a"
       special-names.
           alphabet mixed is " AabBcCdDeEfFgGhHiIjJkKlLmMnNoOpPqQrRsStTu
      -"UvVwWxXyYzZ0123456789".

       input-output section.
       file-control.
           select sort-in
               assign keyboard
               organization is line sequential.
           select sort-out
               assign display
               organization is line sequential.
           select sort-work
               assign "sortwork".

       data division.
       file section.
       fd sort-in.
          01 in-rec        pic x(255).
       fd sort-out.
          01 out-rec       pic x(255).
       sd sort-work.
          01 work-rec      pic x(255).

       working-storage section.
```

```
       01 loop-flag        pic x value low-value.

       procedure division.
       sort sort-work
           on descending key work-rec
           collating sequence is mixed
           input procedure is sort-transform
           output procedure is output-uppercase.

       display sort-return.
       goback.

      ****************************************************************
       sort-transform.
       move low-value to loop-flag
       open input sort-in
       read sort-in
           at end move high-value to loop-flag
       end-read
       perform
           until loop-flag = high-value
               move in-rec to work-rec
               RELEASE work-rec
               read sort-in
                   at end move high-value to loop-flag
               end-read
       end-perform
       close sort-in
       .

      ****************************************************************
       output-uppercase.
       move low-value to loop-flag
       open output sort-out
       return sort-work
           at end move high-value to loop-flag
       end-return
       perform
           until loop-flag = high-value
               move work-rec to out-rec
               write out-rec end-write
               return sort-work
                   at end move high-value to loop-flag
               end-return
       end-perform
       close sort-out
       .

       exit program.
       end program sorting.
```

RELEASE : https://riptutorial.com/ko/cobol/topic/7337/release-

31: REPLACE

`REPLACE` COBOL . .



Examples

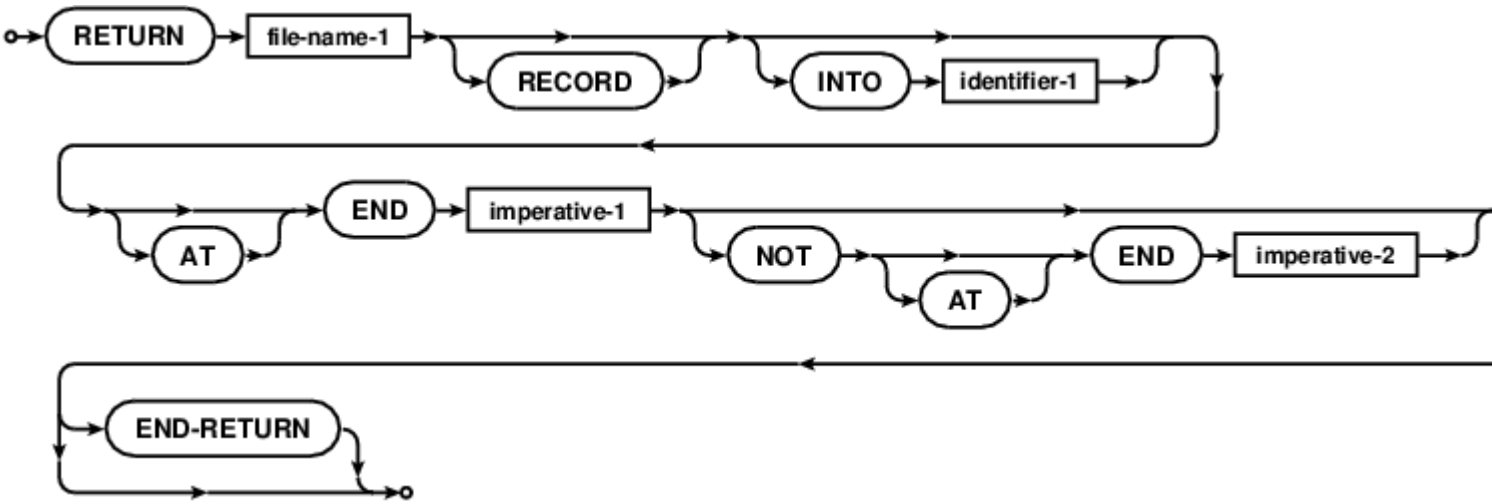
.

```
REPLACE ==magic-number== BY ==65535==.
```

REPLACE : https://riptutorial.com/ko/cobol/topic/7459/replace-

32: RETURN

`RETURN OUTPUT PROCEDURE` COBOL . .



Examples

SORT OUTPUT PROCEDURE .

`. SORT OUTPUT PROCEDURE` COBOL . `work-rec out-rec` .

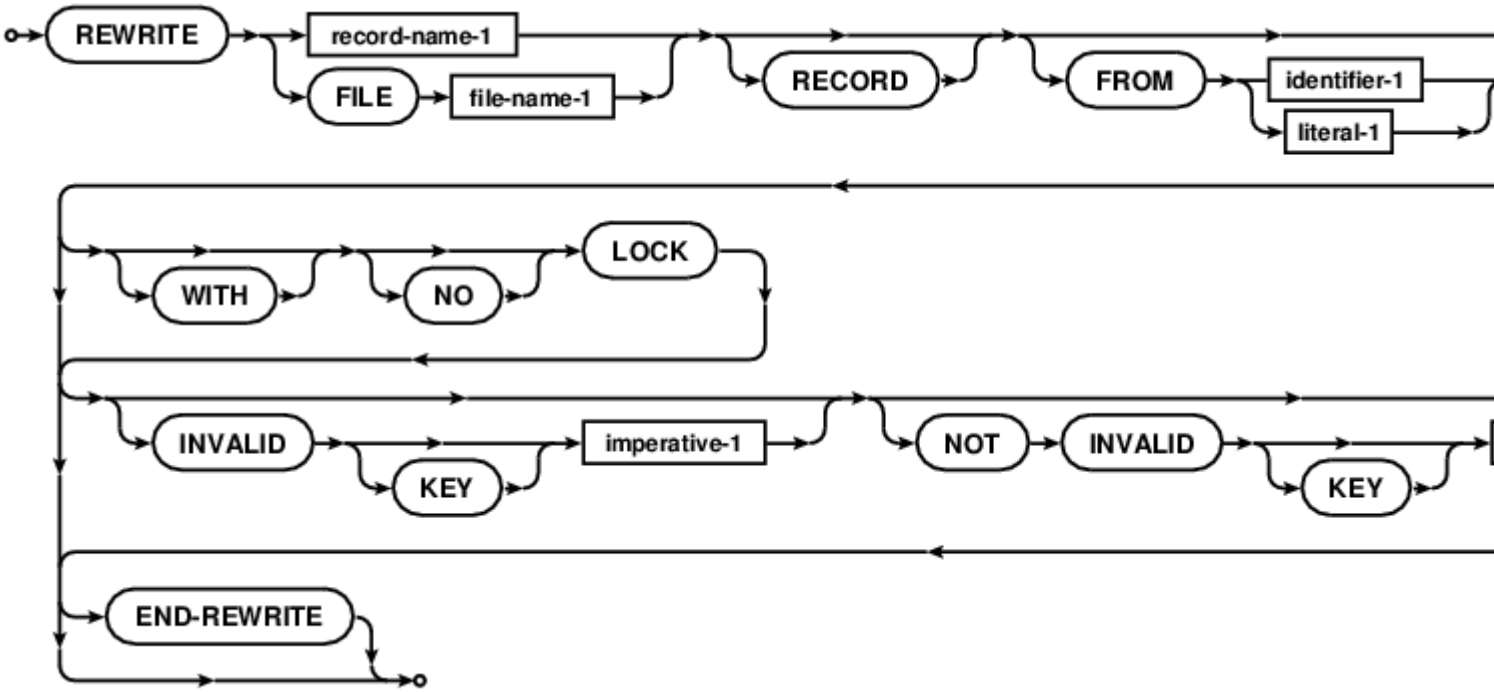
```
GCobol >>SOURCE FORMAT IS FIXED
      ************************************************************
      * Purpose:   A GnuCOBOL SORT verb example
      * Tectonics: cobc -x sorting.cob
      *     ./sorting <input >output
      *   or simply
      *     ./sorting
      *   for keyboard and screen demos
      ************************************************************
       identification division.
       program-id. sorting.

       environment division.
       configuration section.
      * Set up a sort order where lower and upper case stay together
       special-names.
           alphabet mixed is " aAbBcCdDeEfFgGhHiIjJkKlLmMnNoOpPqQrRsStTu
      -"UvVwWxXyYzZ0123456789".

       input-output section.
       file-control.
           select sort-in
               assign keyboard
               organization is line sequential.
           select sort-out
               assign display
               organization is line sequential.
           select sort-work
               assign "sortwork".
```

```
 data division.
 file section.
 fd sort-in.
    01 in-rec        pic x(255).
 fd sort-out.
    01 out-rec       pic x(255).
 sd sort-work.
    01 work-rec      pic x(255).

 working-storage section.
 01 loop-flag        pic x value low-value.

 procedure division.
 sort sort-work
     on descending key work-rec
     collating sequence is mixed
     input procedure is sort-reader
     output procedure is sort-writer.

 display sort-return.
 goback.

*****************************************************************
 sort-reader.
 move low-value to loop-flag
 open input sort-in
 read sort-in
     at end move high-value to loop-flag
 end-read
 perform
     until loop-flag = high-value
         move in-rec to work-rec
         release work-rec
         read sort-in
             at end move high-value to loop-flag
         end-read
 end-perform
 close sort-in
 .

*****************************************************************
 sort-writer.
 move low-value to loop-flag
 open output sort-out
 return sort-work
     at end move high-value to loop-flag
 end-return
 perform
     until loop-flag = high-value
         move work-rec to out-rec
         write out-rec end-write
         RETURN sort-work
             at end move high-value to loop-flag
         end-return
 end-perform
 close sort-out
 .

 exit program.
 end program sorting.
```

RETURN : https://riptutorial.com/ko/cobol/topic/7338/return-

33: REWRITE

REWRITE .



Examples

RELATIVE REWRITE

```
GCobol >>SOURCE FORMAT IS FIXED
      *> ****************************************************************
      *> Purpose:   RELATIVE file organization  REWRITE example
      *> Tectonics: cobc -g -debug -W -x relatives.cob
      *> ****************************************************************
       identification division.
       program-id. relatives.

       environment division.
       configuration section.
       repository.
           function all intrinsic.

       input-output section.
       file-control.
           select optional relatives
               assign to "relatives.dat"
               file status is filestatus
               organization is relative
               access mode is dynamic
               relative key is nicknum.

       data division.
       file section.
       fd relatives.
          01 person.
```

```
       05 firstname      pic x(48).
       05 lastname       pic x(64).
       05 relationship   pic x(32).

 working-storage section.
 77 filestatus pic 9(2).
    88 ineof value 1 when set to false is 0.

 77 satisfaction pic 9.
    88 satisfied value 1 when set to false is 0.

 77 nicknum   pic 9(2).

 77 title-line pic x(34).
    88 writing-names value "Adding, Overwriting.  00 to finish".
    88 reading-names value "Which record?         00 to quit".
 77 problem   pic x(80).

 screen section.
 01 detail-screen.
    05           line 1 column 1  from title-line erase eos.
    05           line 2 column 1  value "Record: ".
    05 pic 9(2)  line 2 column 16 using nicknum.
    05           line 3 column 1  value "First name: ".
    05 pic x(48) line 3 column 16 using firstname.
    05           line 4 column 1  value "Last name: ".
    05 pic x(64) line 4 column 16 using lastname.
    05           line 5 column 1  value "Relation: ".
    05 pic x(32) line 5 column 16 using relationship.
    05 pic x(80) line 6 column 1  from problem.

 01 show-screen.
    05           line 1 column 1  from title-line erase eos.
    05           line 2 column 1  value "Record: ".
    05 pic 9(2)  line 2 column 16 using nicknum.
    05           line 3 column 1  value "First name: ".
    05 pic x(48) line 3 column 16 from firstname.
    05           line 4 column 1  value "Last name: ".
    05 pic x(64) line 4 column 16 from lastname.
    05           line 5 column 1  value "Relation: ".
    05 pic x(32) line 5 column 16 from relationship.
    05 pic x(80) line 6 column 1  from problem.

*> -*********-*********-*********-*********-*********-**********-**
 procedure division.
 beginning.

*> Open the file and find the highest record number
*> which is a sequential read operation after START
     open input relatives

     move 99 to nicknum
     start relatives key is less than or equal to nicknum
         invalid key
             move concatenate('NO START' space filestatus)
               to problem
             move 00 to nicknum
         not invalid key
             read relatives next end-read
     end-start
```

```
*> Close and open for i-o
     close relatives
     open i-o relatives

*> Prompt for numbers and names to add until 00
     set writing-names to true
     set satisfied to false
     perform fill-file through fill-file-end
         until satisfied

     close relatives

*> Prompt for numbers to view names of until 00
     open input relatives

     set reading-names to true
     set satisfied to false
     perform record-request through record-request-end
         until satisfied

     perform close-shop
 .
 ending.
     goback.

*> get some user data to add
 fill-file.
     display detail-screen.
     accept detail-screen.
     move spaces to problem
     if nicknum equal 0
         set satisfied to true
         go to fill-file-end
     end-if.
 .
 write-file.
     write person
         invalid key
             move concatenate("overwriting: " nicknum) to problem
             REWRITE person
                 invalid key
                     move concatenate(
                         exception-location() space nicknum
                         space filestatus)
                     to problem
             END-REWRITE
     end-write.
     display detail-screen

 .
 fill-file-end.
 .

*> get keys to display
 record-request.
     display show-screen
     accept show-screen
     move spaces to problem
     if nicknum equals 0
         set satisfied to true
         go to record-request-end
```

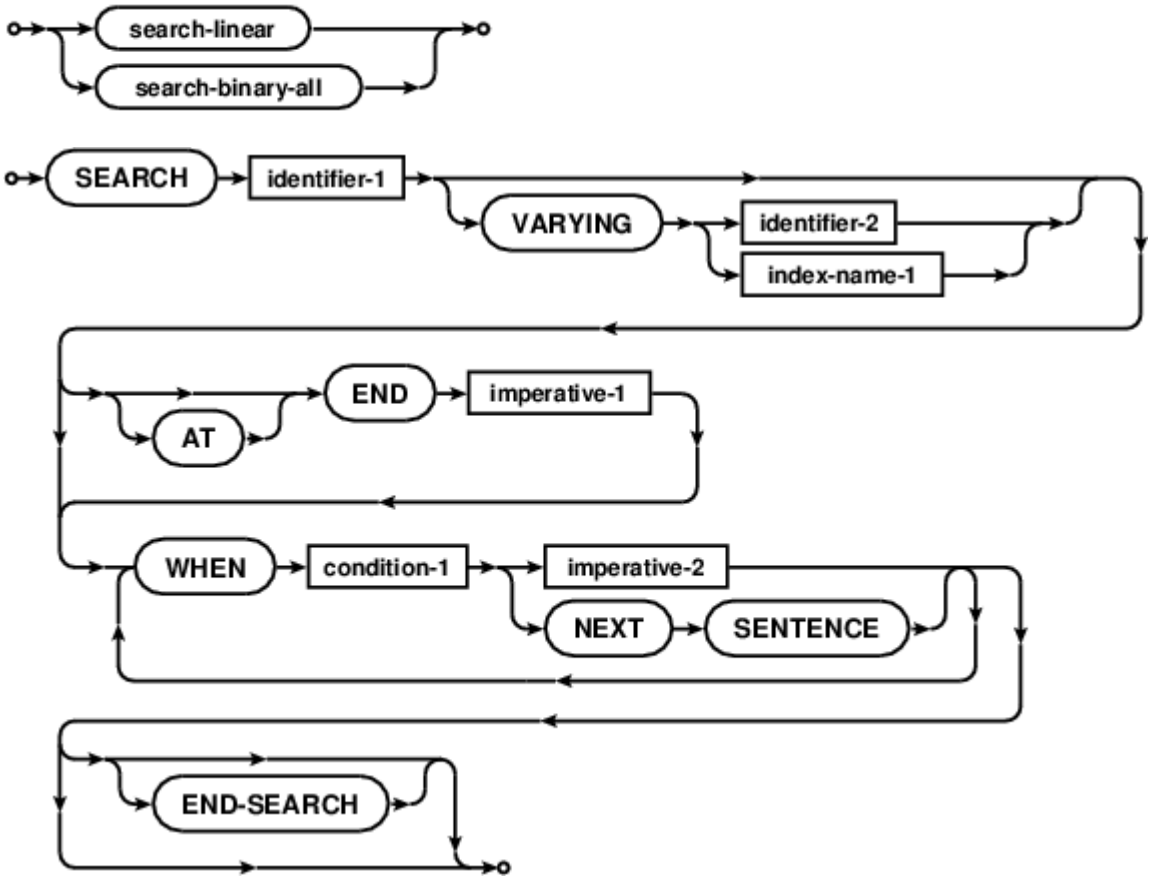
```
           end-if
       .

      *> The magic of relative record number reads
       read-relation.
           read relatives
               invalid key
                   move exception-location() to problem
               not invalid key
                   move spaces to problem
           end-read
           display show-screen
       .

       record-request-end.
       .

      *> get out <*
       close-shop.
           close relatives.
           goback.
       .
       end program relatives.
```

REWRITE : https://riptutorial.com/ko/cobol/topic/7460/rewrite-

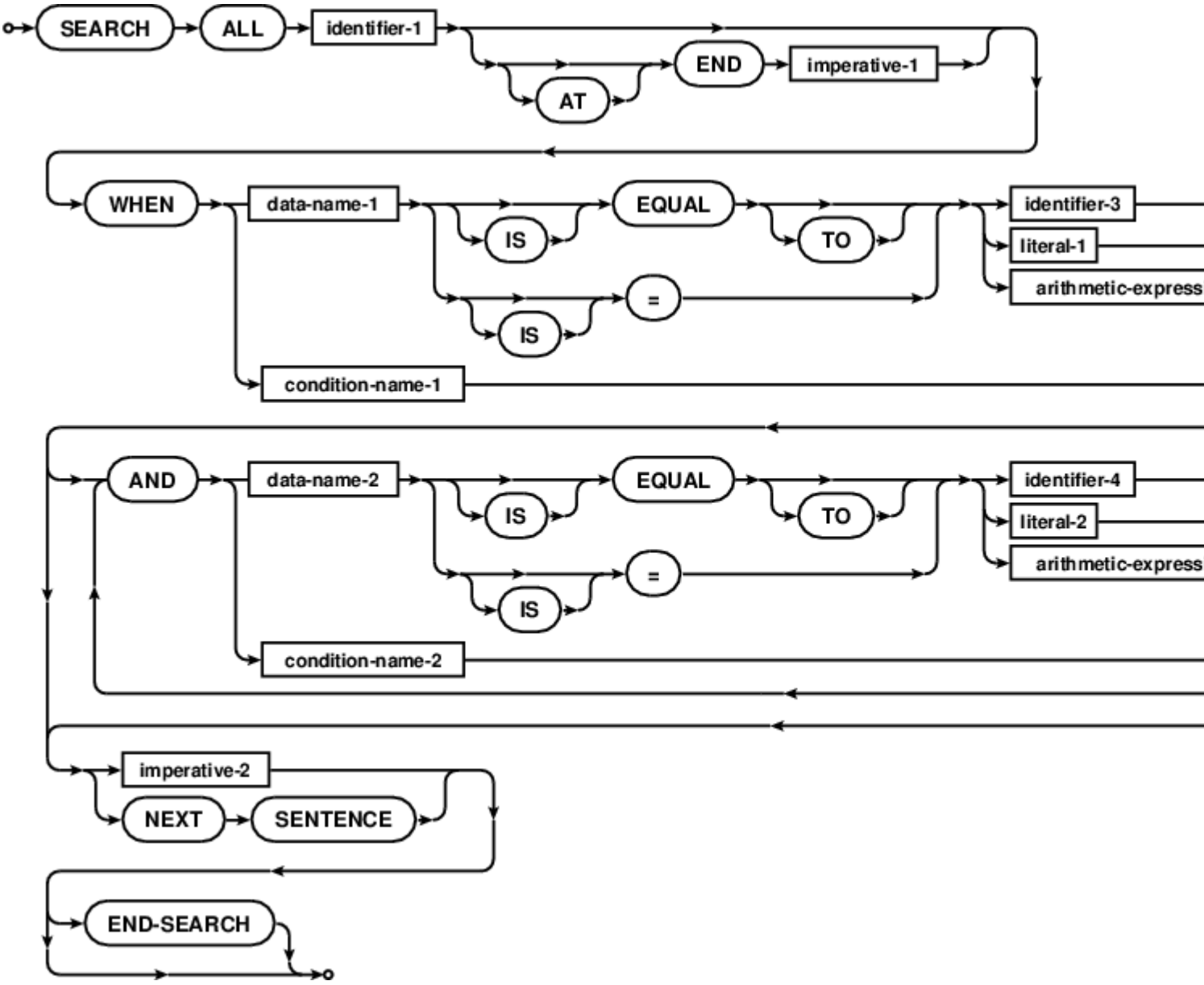
34: SEARCH

COBOL `SEARCH` . `SEARCH` `SEARCH ALL` . 2 SEARCH ALL 2 .

SEARCH



ALL



Examples

```
GCobol >>SOURCE FORMAT IS FIXED
      *> ***************************************************************
      *> Purpose:   Demonstration of the SEARCH verb
      *> Tectonics: cobc -x searchlinear.cob
      *> ***************************************************************
       identification division.
       program-id. searchlinear.

       data division.

       working-storage section.
       01 taxinfo.
          05 tax-table occurs 4 times indexed by tt-index.
             10 province       pic x(2).
             10 taxrate        pic 999v9999.
             10 federal        pic 999v9999.
       01 prov                 pic x(2).
       01 percent              pic 999v9999.
       01 percentage           pic zz9.99.
```

```
      *> ****************************************************************
       procedure division.
       begin.

      *> ****************************************************************
      *> Sample for linear SEARCH, requires INDEXED BY table
      *> populate the provincial tax table;
      *>  *** (not really, only a couple of sample provinces) ***
      *> populate Ontario and PEI using different field loaders
       move 'AB' to province(1)
       move 'ON' to province(2)
       move 0.08 to taxrate(2)
       move 0.05 to federal(2)
       move 'PE00014000000000' to tax-table(3)
       move 'YT' to province(4)

      *> Find Ontario tax rate
       move "ON" to prov
       perform search-for-taxrate

      *> Setup for Prince Edward Island
       move 'PE' to prov
       perform search-for-taxrate

      *> Setup for failure
       move 'ZZ' to prov
       perform search-for-taxrate

       goback.
      *> ****************************************************************

       search-for-taxrate.
           set tt-index to 1
           search tax-table
               at end display "no province: " prov end-display
               when province(tt-index) = prov
                   perform display-taxrate
           end-search
       .

       display-taxrate.
           compute percent = taxrate(tt-index) * 100
           move percent to percentage
           display
               "found: " prov " at " taxrate(tt-index)
               "," percentage "%, federal rate of " federal(tt-index)
           end-display
       .

       end program searchlinear.
```

ALL

```
GCobol >>SOURCE FORMAT IS FIXED
      *> ****************************************************************
      *> Purpose:   Demonstration of the SEARCH ALL verb and table SORT
      *> Tectonics: cobc -x -fdebugging-line searchbinary.cob
      *> ****************************************************************
```

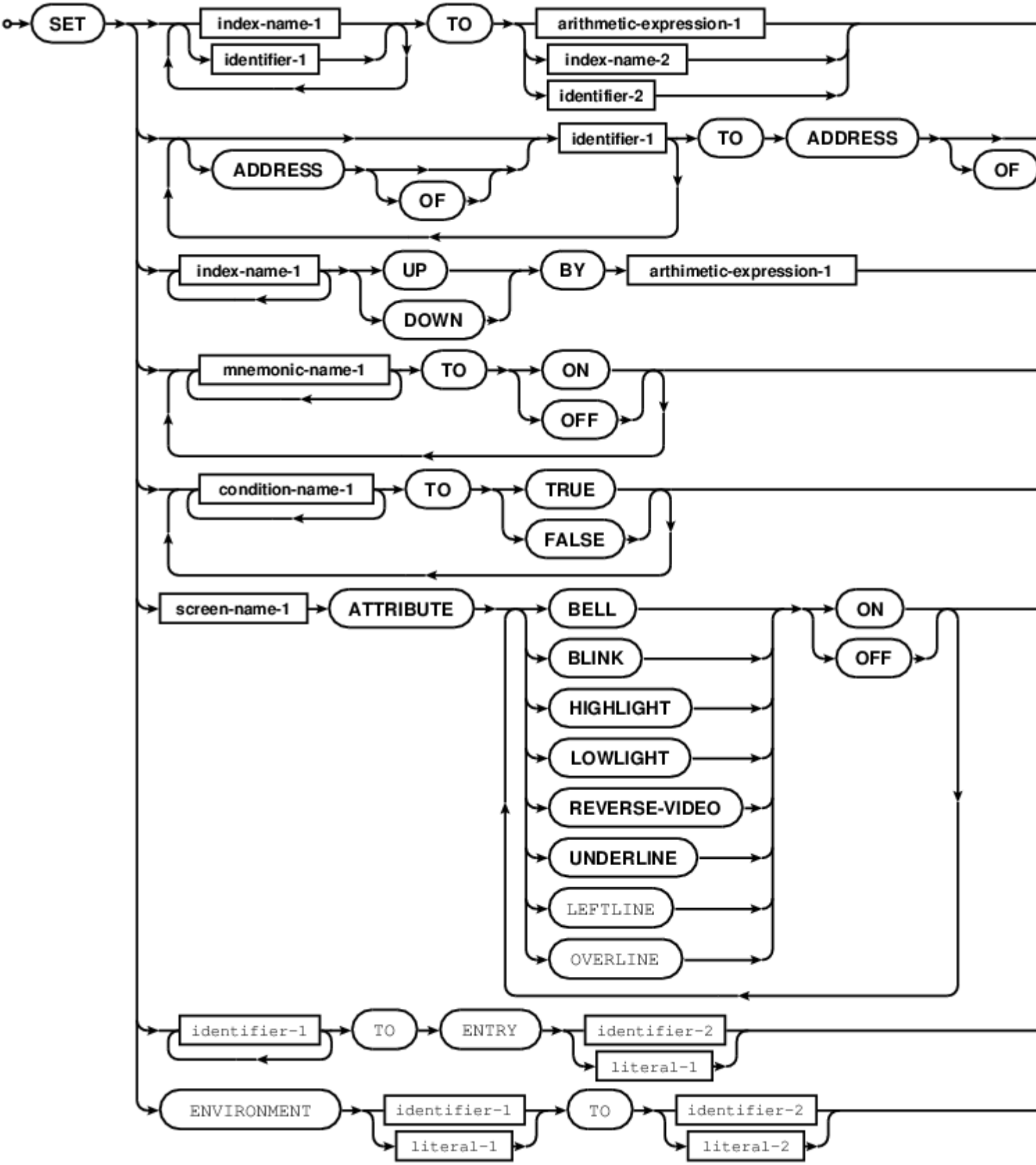
```
       identification division.
       program-id. searchbinary.

       environment division.
       input-output section.
       file-control.
           select optional wordfile
           assign to infile
           organization is line sequential.

       data division.
       file section.
       fd wordfile.
           01 wordrec          pic x(20).

       working-storage section.
       01 infile               pic x(256) value spaces.
          88 defaultfile       value '/usr/share/dict/words'.
       01 arguments            pic x(256).

      *> Note the based clause, this memory is initially unallocated
       78 maxwords            value 500000.
       01 wordlist             based.
          05 word-table occurs maxwords times
              depending on wordcount
              descending key is wordstr
              indexed by wl-index.
             10 wordstr        pic x(20).
             10 wordline       usage binary-long.
       01 wordcount            usage binary-long.

       01 file-eof             pic 9 value low-value.
          88 at-eof            value high-values.

       01 word                 pic x(20).

      *> ****************************************************************
       procedure division.
       begin.

      *> Get the word file filename
       accept arguments from command-line end-accept
       if arguments not equal spaces
           move arguments to infile
       else
           set defaultfile to true
       end-if

      *> ****************************************************************
      *> Try playing with the words file and binary SEARCH ALL
      *>   requires KEY IS and INDEXED BY table description

      *> Point wordlist to valid memory
       allocate wordlist initialized

       open input wordfile

       move low-value to file-eof
       read wordfile
           at end set at-eof to true
       end-read
```

```
     perform
         with test before
         until at-eof or (wordcount >= maxwords)
             add 1 to wordcount
             move wordrec to wordstr(wordcount)
             move wordcount to wordline(wordcount)
             read wordfile
                 at end set at-eof to true
             end-read
     end-perform

     close wordfile

    *> ensure a non-zero length table when allowing optional file
     evaluate true                  also file-eof
         when wordcount = 0         also any
             move 1 to wordcount
             display "No words loaded" end-display
         when wordcount >= maxwords also low-value
             display "Word list truncated to " maxwords end-display
     end-evaluate

  >>D display "Count: " wordcount ": " wordstr(wordcount) end-display

    *> Sort the words from z to a
     sort word-table on descending key wordstr

    *> fetch a word to search for
     display "word to find: " with no advancing end-display
     accept word end-accept

    *> binary search the words for word typed in and display
    *> the original line number if/when a match is found
     set wl-index to 1
     search all word-table
         at end
             display
                 word " not a word of " function trim(infile)
             end-display
         when wordstr(wl-index) = word
             display
                 word " sorted to " wl-index ", originally "
                 wordline(wl-index) " of " function trim(infile)
             end-display
     end-search

    *> Release memory ownership
     free address of wordlist

     goback.
     end program searchbinary.
```

SEARCH : https://riptutorial.com/ko/cobol/topic/7462/search-

35: SET

COBOL `SET` . `SET` .



Examples

SET

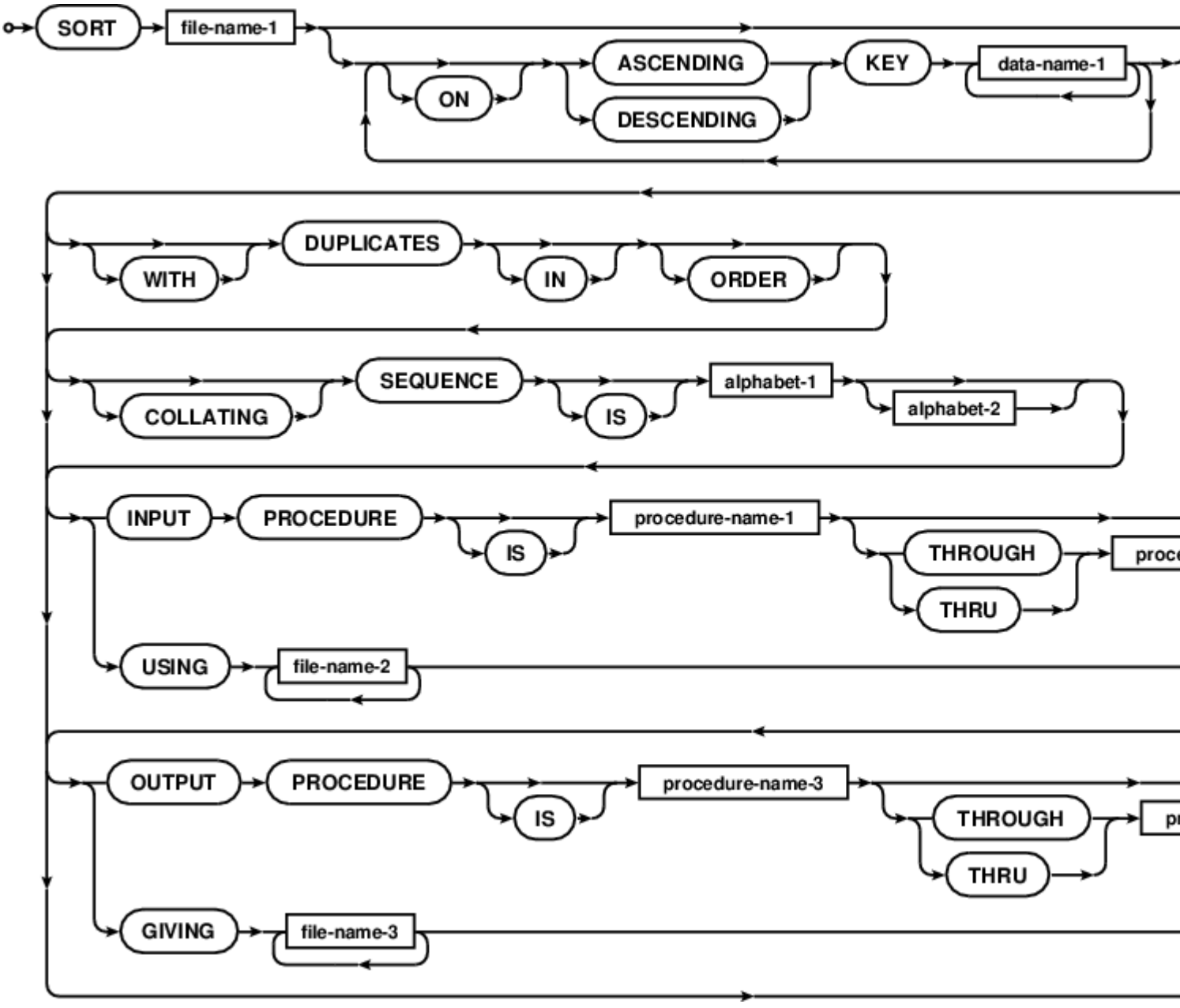
```
SET handle TO returned-pointer
SET handle UP BY LENGTH(returned-pointer)
SET ADDRESS OF buffer-space TO handle
MOVE buffer-space TO work-store
DISPLAY "Second element is " work-store
```

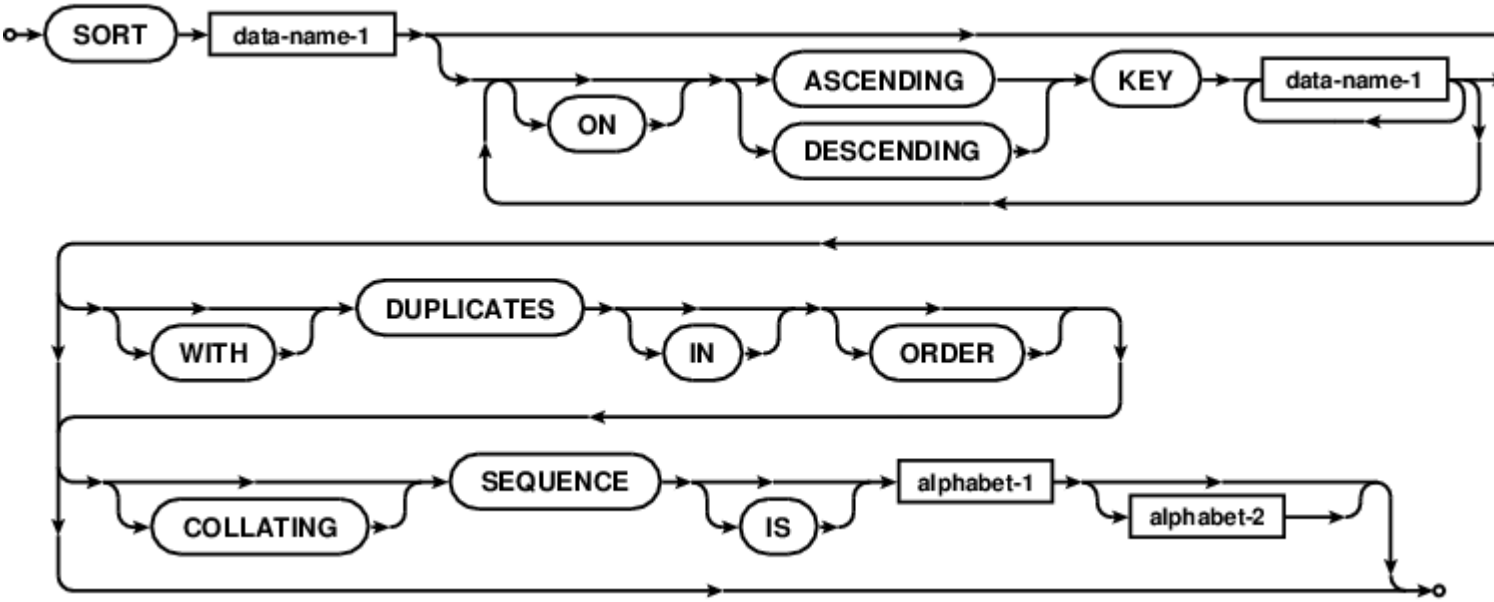
SET : https://riptutorial.com/ko/cobol/topic/7461/set-

36: SORT

COBOL SORT . W .

SORT





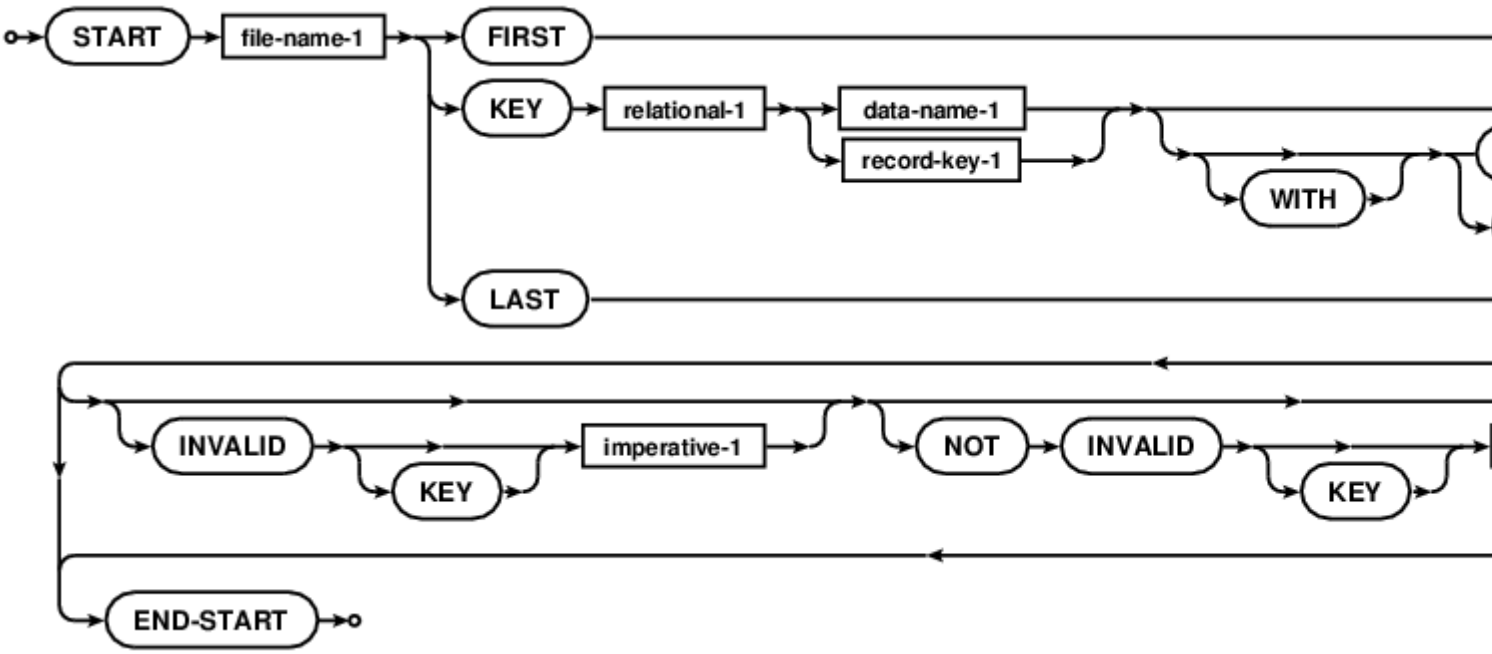
Examples

```
GCobol* GnuCOBOL SORT verb example using standard in and standard out
       identification division.
       program-id. sorting.

       environment division.
       input-output section.
       file-control.
           select sort-in
               assign keyboard
               organization line sequential.
           select sort-out
               assign display
               organization line sequential.
           select sort-work
               assign "sortwork".

       data division.
       file section.
       fd sort-in.
          01 in-rec        pic x(255).
       fd sort-out.
          01 out-rec       pic x(255).
       sd sort-work.
          01 work-rec      pic x(255).

       procedure division.
       sort sort-work
           ascending key work-rec
           using  sort-in
           giving sort-out.

       goback.
       exit program.
       end program sorting.
```

SORT : https://riptutorial.com/ko/cobol/topic/7463/sort-

37: START

START ( ) .



( ).

- .
- >
- .
- <
- .
- =
- .
- KEY IS NOT>
- .
- <
- .
- =
- <>
- .

- KEY IS = =
- .
- KEY <=

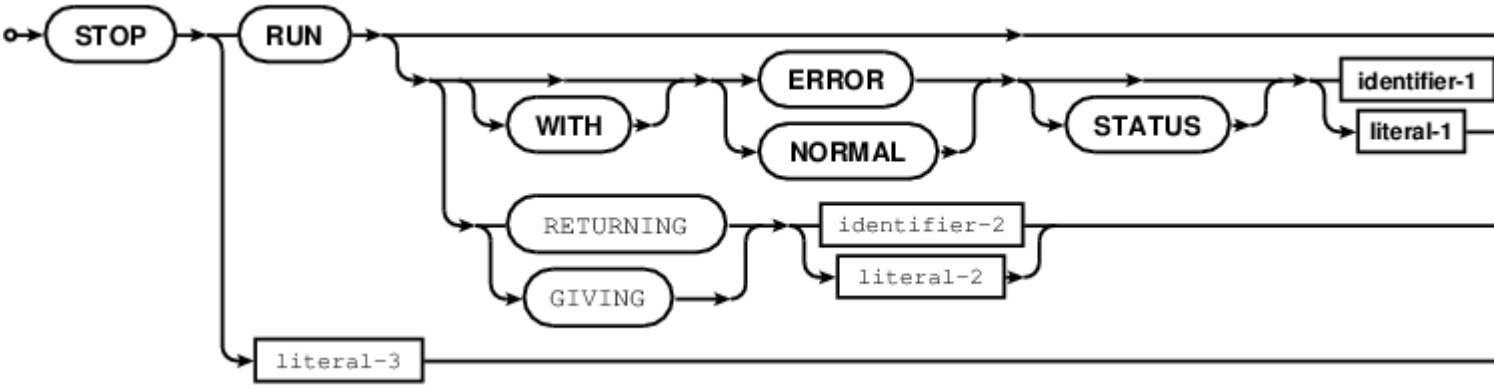
Examples

```
start indexing
   key is less than
       keyfield of indexing-record
   invalid key
       display "bad start: " keyfield of indexing-record
       set no-more-records to true
   not invalid key
       read indexing previous record
           at end set no-more-records to true
       end-read
end-start
```

START : https://riptutorial.com/ko/cobol/topic/7464/start-

38: STOP

```
STOP    .

STOP RUN   STOP literal        . "    "    .

STOP   . GOBACK        .    .
```



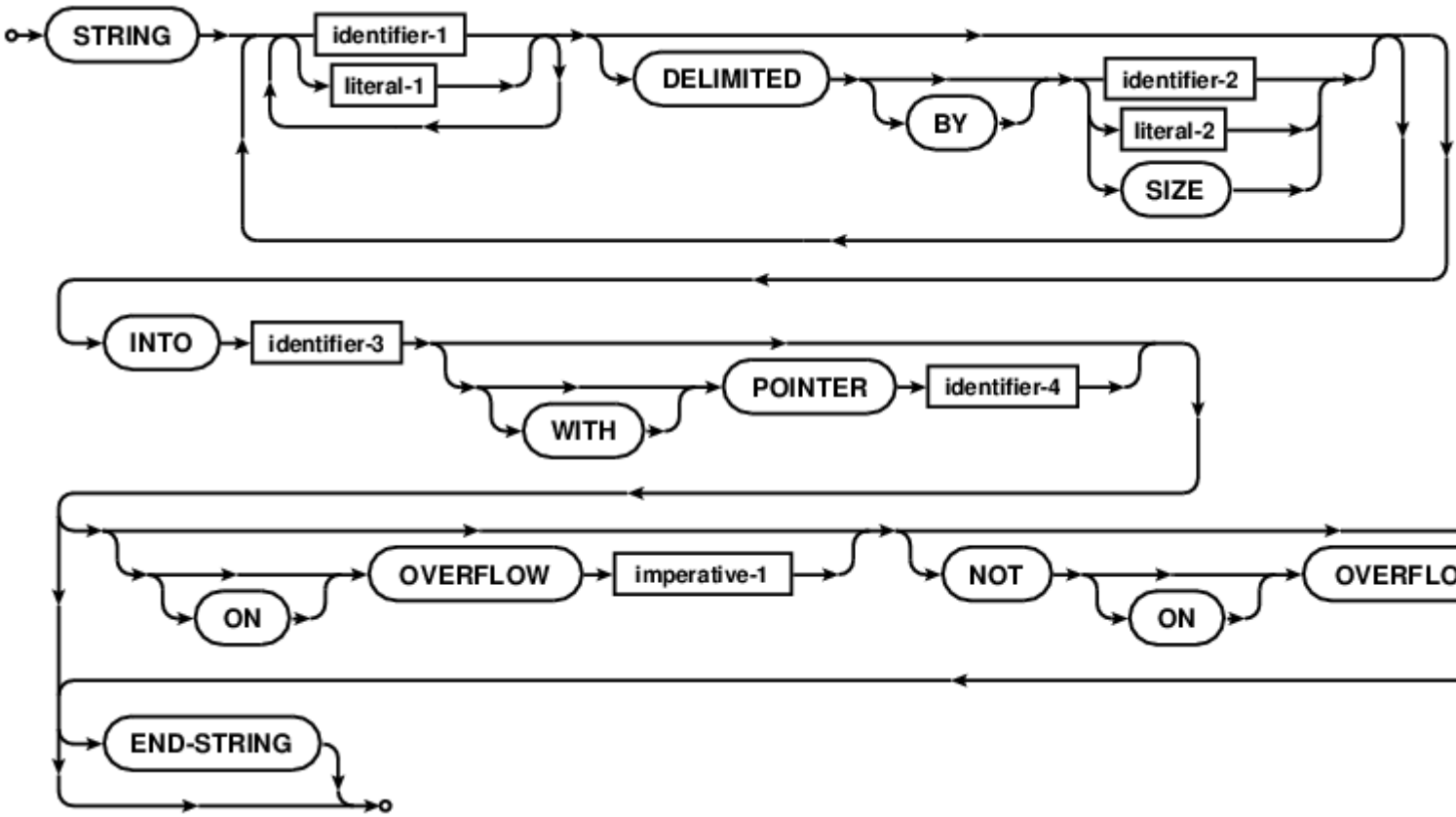
Examples

```
STOP RUN
```

STOP : https://riptutorial.com/ko/cobol/topic/7466/stop-

39: STRING

STRING .



Examples

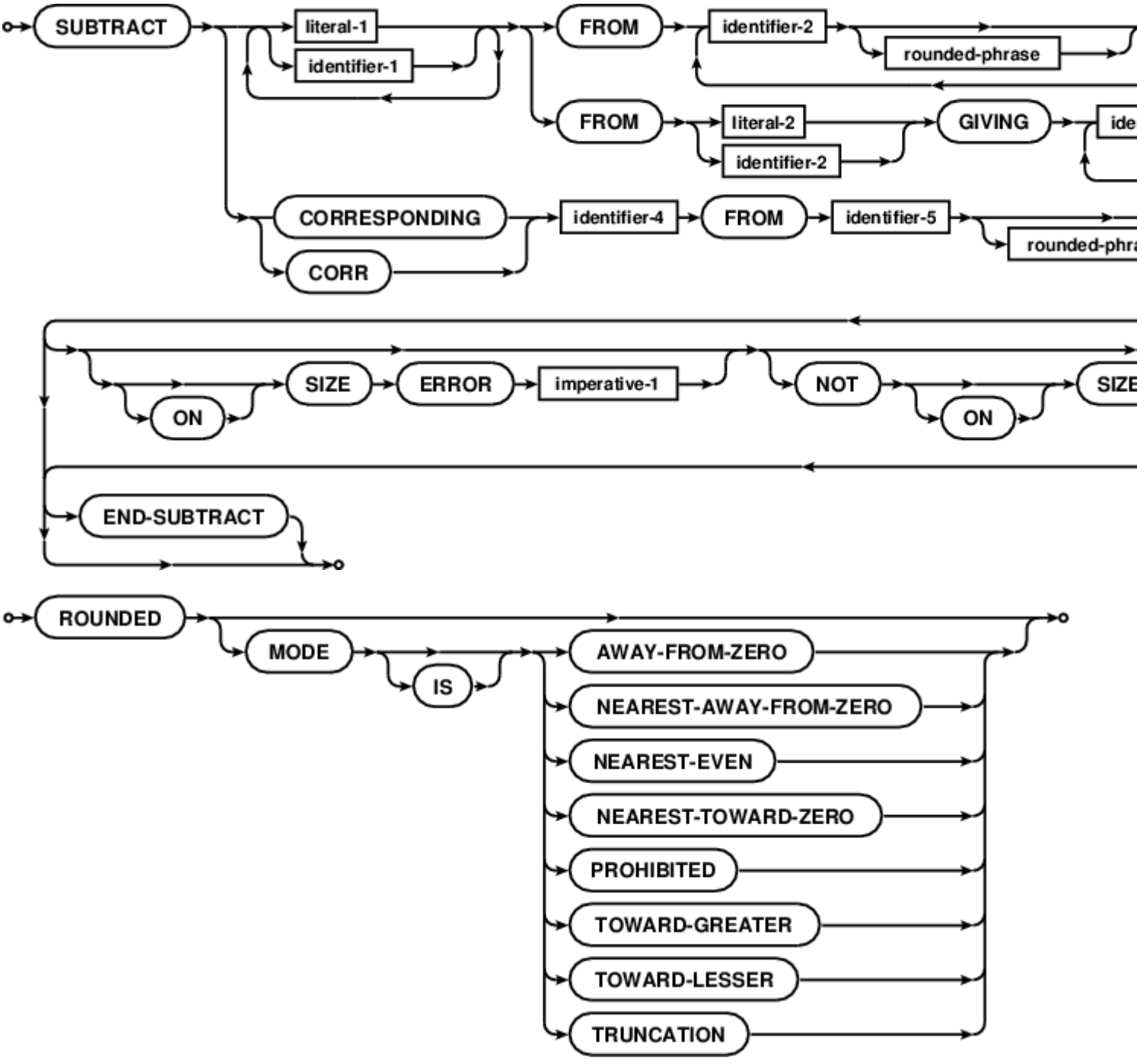
C STRING

```
*> Strip off trailing zero bytes
STRING c-string DELIMITED BY LOW-VALUE INTO working-store
```

STRING : https://riptutorial.com/ko/cobol/topic/7468/string-

40: SUBTRACT

SUBTRACT .



Examples

SUBTRACT

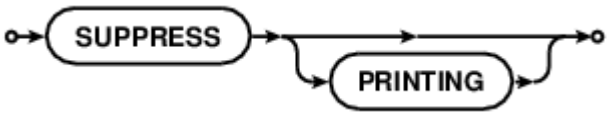
```
SUBTRACT item-a item-b item-c FROM account-z ROUNDED MODE IS NEAREST-EVEN
    ON SIZE ERROR
        DISPLAY "CALL THE BOSS, Account `Z` is OUT OF MONEY" END-DISPLAY
        PERFORM promisary-processing
```

```
    NOT ON SIZE ERROR
        PERFORM normal-processing
END-SUBTRACT
```

SUBTRACT : https://riptutorial.com/ko/cobol/topic/7465/subtract-

41: SUPPRESS

`SUPPRESS` . COBOL .



Examples

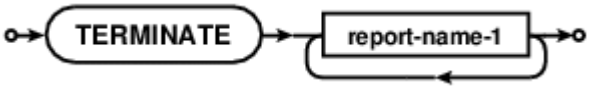
SUPPRESS

```
SUPPRESS PRINTING
```

SUPPRESS : https://riptutorial.com/ko/cobol/topic/7470/suppress-

42: TERMINATE

`TERMINATE` COBOL . .



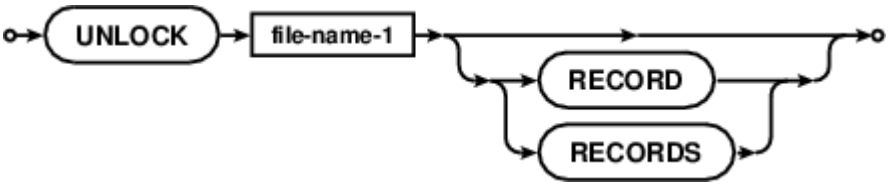
Examples

```
TERMINATE report-1 report-2 report-summary
```

TERMINATE : https://riptutorial.com/ko/cobol/topic/7467/terminate-

43: UNLOCK

UNLOCK .



Examples

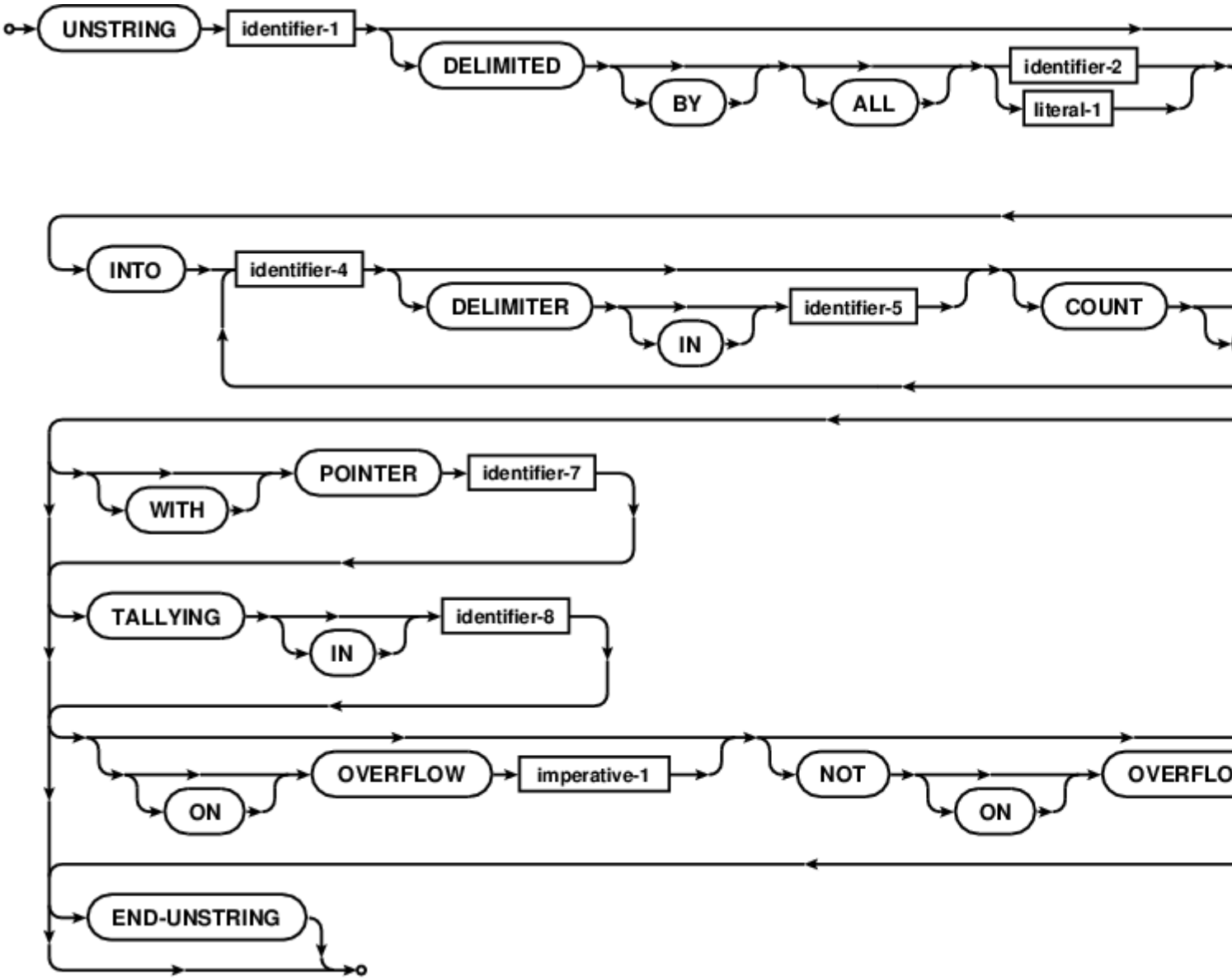
UNLOCK

```
UNLOCK filename-1 RECORDS
```

UNLOCK : https://riptutorial.com/ko/cobol/topic/7471/unlock-

44: UNSTRING

UNSTRING .



Examples

UNSTRING

```
UNSTRING Input-Address
    DELIMITED BY "," OR "/"
    INTO
        Street-Address DELIMITER D1 COUNT C1
        Apt-Number DELIMITER D2 COUNT C2
        City DELIMITER D3 COUNT C3
        State DELIMITER D4 COUNT C4
        Zip-Code DELIMITER D5 COUNT C5
    WITH POINTER ptr-1
    ON OVERFLOW
```

```
        SET more-fields TO TRUE
END-UNSTRING
```

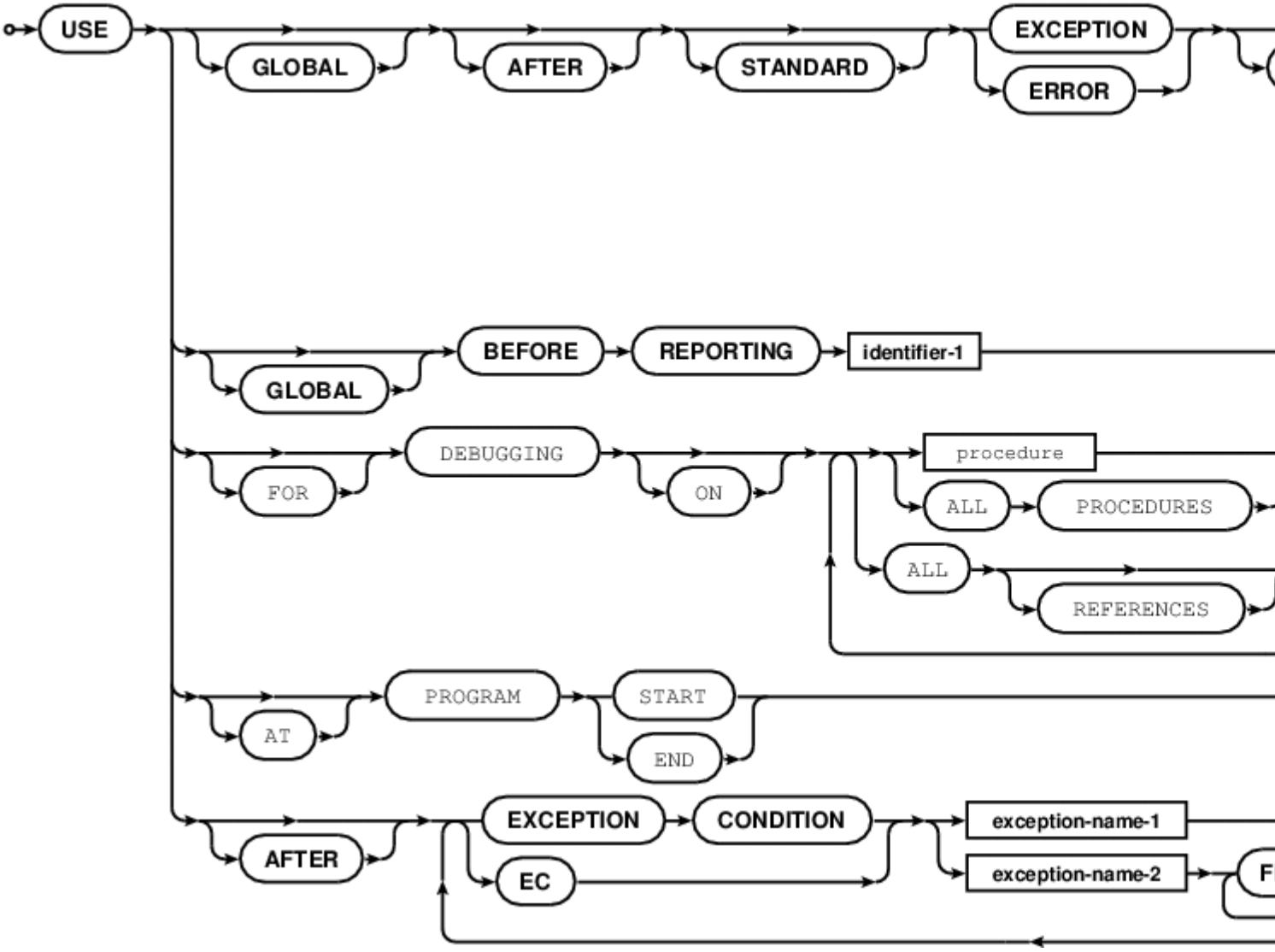
UNSTRING : https://riptutorial.com/ko/cobol/topic/7581/unstring-

45: USE

USE .

-
-
-

DEBUGGING , .



Examples

USE

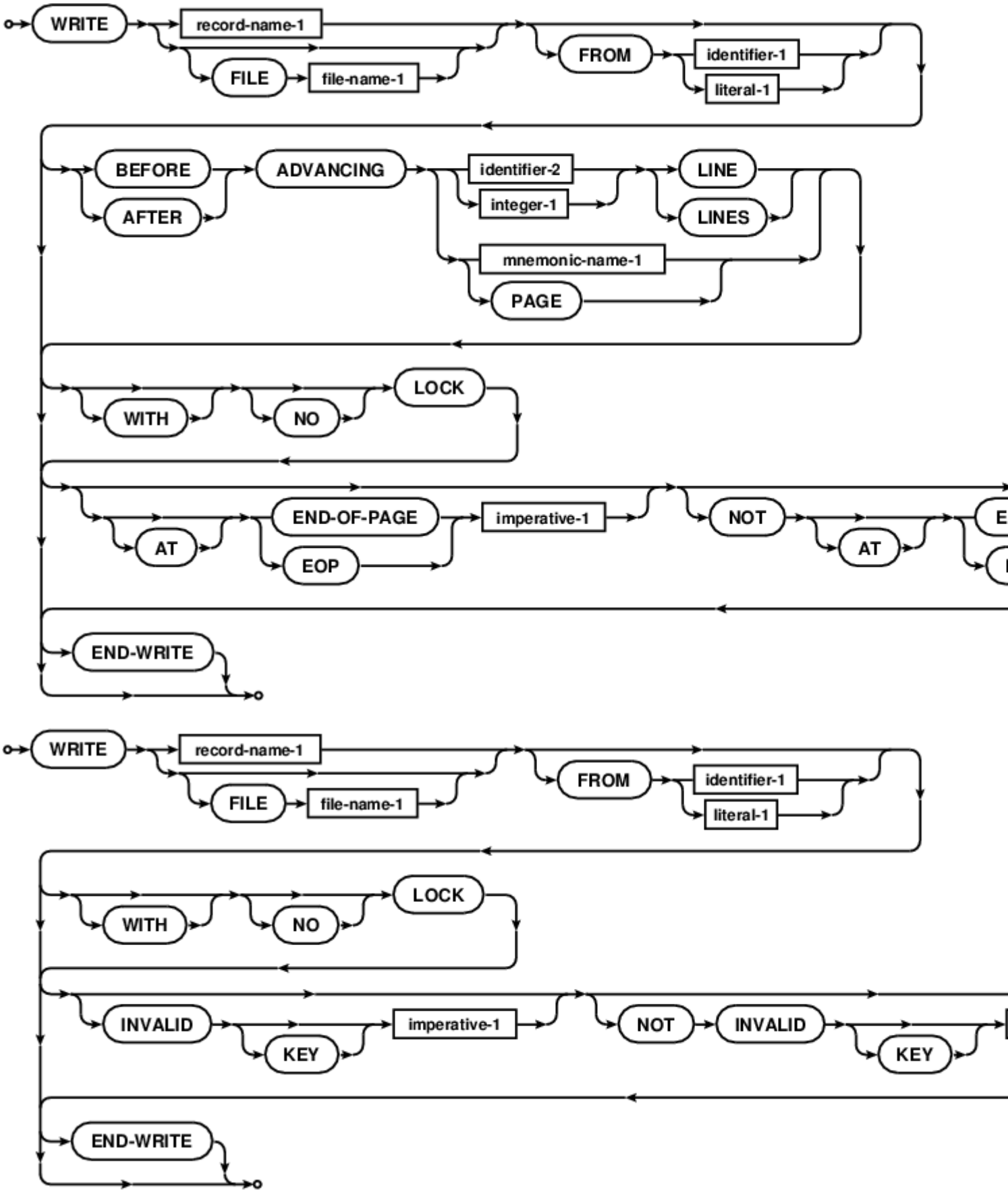
```
035700 PROCEDURE DIVISION.
035800
035900 DECLARATIVES.
036000
036100 DEPT-HEAD-USE SECTION. USE BEFORE REPORTING DEPT-HEAD.
036200 DEPT-HEAD-PROC.
```

```
036300     SET DE-IX TO +1.
036400     SEARCH DEPARTMENT-ENTRY
036500         WHEN DE-NUMBER (DE-IX) = PRR-DEPARTMENT-NUMBER
036600             MOVE ZEROS TO DE-GROSS (DE-IX), DE-FICA (DE-IX),
036700                           DE-FWT (DE-IX), DE-MISC (DE-IX),
036800                           DE-NET (DE-IX).
036900
037000 DEPT-HEAD-EXIT.
037100     EXIT.
037200
037300 EMPL-FOOT-USE SECTION. USE BEFORE REPORTING EMPL-FOOT.
037400 EMPL-FOOT-PROC.
037500     MOVE PRR-EMPLOYEE-KEY TO WS-EMPLOYEE-KEY.
037600
037700 EMPL-FOOT-EXIT.
037800     EXIT.
037900
038000 DEPT-FOOT-USE SECTION. USE BEFORE REPORTING DEPT-FOOT.
038100 DEPT-FOOT-PROC.
038200     MOVE DEPT-FOOT-GROSS TO DE-GROSS (DE-IX).
038300     MOVE DEPT-FOOT-FICA TO DE-FICA (DE-IX).
038400     MOVE DEPT-FOOT-FWT TO DE-FWT (DE-IX).
038500     MOVE DEPT-FOOT-MISC TO DE-MISC (DE-IX).
038600     MOVE DEPT-FOOT-NET TO DE-NET (DE-IX).
      *     SUPPRESS PRINTING.
038700
038800 DEPT-FOOT-EXIT.
038900     EXIT.
039000
039100 COMP-FOOT-USE SECTION. USE BEFORE REPORTING COMP-FOOT.
039200 COMP-FOOT-PROC.
039300     PERFORM COMP-FOOT-CALC
039400         VARYING WPCD-IX FROM +1 BY +1
039500         UNTIL WPCD-IX > +6.
039600     GO TO COMP-FOOT-EXIT.
039700
039800 COMP-FOOT-CALC.
039900     SET DE-IX TO WPCD-IX.
040000     SET WPCC-IX TO +1.
040100     COMPUTE WPC-PERCENT (WPCD-IX WPCC-IX) ROUNDED =
040200         ((DE-GROSS (DE-IX) / CO-GROSS) * 100) + .5.
040300     SET WPCC-IX TO +2.
040400     COMPUTE WPC-PERCENT (WPCD-IX WPCC-IX) ROUNDED =
040500         ((DE-FICA (DE-IX) / CO-FICA) * 100) + .5.
040600     SET WPCC-IX TO +3.
040700     COMPUTE WPC-PERCENT (WPCD-IX WPCC-IX) ROUNDED =
040800         ((DE-FWT (DE-IX) / CO-FWT) * 100) + .5.
040900     SET WPCC-IX TO +4.
041000     COMPUTE WPC-PERCENT (WPCD-IX WPCC-IX) ROUNDED =
041100         ((DE-MISC (DE-IX) / CO-MISC) * 100) + .5.
041200     SET WPCC-IX TO +5.
041300     COMPUTE WPC-PERCENT (WPCD-IX WPCC-IX) ROUNDED =
041400         ((DE-NET (DE-IX) / CO-NET) * 100) + .5.
041500
041600 COMP-FOOT-EXIT.
041700     EXIT.
041800
041900 END DECLARATIVES.
```

USE : https://riptutorial.com/ko/cobol/topic/7582/use-

46: WRITE

```
WRITE   output  input-output       .
```



Examples

```
WRITE record-buff

WRITE indexed-record
    WITH LOCK
    ON INVALID KEY
        DISPLAY "Key exists, REWRITING..." END-DISPLAY
        PERFORM rewrite-key
END-WRITE
IF indexed-file-status NOT EQUAL ZERO THEN
    DISPLAY "Write problem: " indexed-file-status UPON SYSERR
    END-DISPLAY
    PERFORM evasive-manoeuvres
END-IF

WRITE record-name-1 AFTER ADVANCING PAGE

WRITE record-name-1 FROM header-record-1
    AFTER ADVANCING 2 LINES
    AT END-OF-PAGE
        PERFORM write-page-header
        PERFORM write-last-detail-reminder
END-WRITE
```

WRITE : https://riptutorial.com/ko/cobol/topic/7583/write-

47:

Examples

STRINGVAL ... - - STRING

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  STRINGVAL.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.

01  WORK-AREAS.
    05 I-STRING        PIC X(08) VALUE   'STRNGVAL'.

    05 O-STRING        PIC XBXBXBXBXBXBXBX.
       88  O-STRING-IS-EMPTY     VALUE   SPACES.

PROCEDURE DIVISION.
GENESIS.

    PERFORM MAINLINE

    PERFORM FINALIZATION

    GOBACK

     .


MAINLINE.

    DISPLAY 'STRINGVAL EXAMPLE IS STARTING !!!!!!!!!!!!!!'

    DISPLAY '=== USING MOVE STATEMENT ==='
    MOVE I-STRING TO O-STRING
    DISPLAY 'O STRING= ' O-STRING

    DISPLAY '=== USING STRING STATEMENT ==='
    SET O-STRING-IS-EMPTY    TO  TRUE
    STRING I-STRING ( 1 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 2 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 3 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 4 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 5 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 6 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 7 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        I-STRING ( 8 : 1 ) DELIMITED BY SIZE
        ' ' DELIMITED BY SIZE
        INTO O-STRING
```

```
       DISPLAY 'O STRING= ' O-STRING

        .


   FINALIZATION.

       DISPLAY 'STRINGVAL EXAMPLE IS COMPLETE !!!!!!!!!!!!!!'

        .

   END PROGRAM STRINGVAL.
```

. (o) "" .

, o- " " "fred & Bert" o- "fred & Bertontains this data" ( ).

, .

: https://riptutorial.com/ko/cobol/topic/7039/

48:

COBOL 0 . . COBOL , .

, , , .

COBOL 2014 .

```
======================================== ==========
Intrinsic Function                       Parameters
======================================== ==========
FUNCTION ABS                             1
FUNCTION ACOS                            1
FUNCTION ANNUITY                         2
FUNCTION ASIN                            1
FUNCTION ATAN                            1
FUNCTION BOOLEAN-OF-INTEGER              2
FUNCTION BYTE-LENGTH                     1
FUNCTION CHAR                            1
FUNCTION CHAR-NATIONAL                   1
FUNCTION COMBINED-DATETIME               2
FUNCTION COS                             1
FUNCTION CURRENCY-SYMBOL                 0
FUNCTION CURRENT-DATE                    0
FUNCTION DATE-OF-INTEGER                 1
FUNCTION DATE-TO-YYYYMMDD                Variable
FUNCTION DAY-OF-INTEGER                  1
FUNCTION DAY-TO-YYYYDDD                  Variable
FUNCTION DISPLAY-OF                      Variable
FUNCTION E                               0
FUNCTION EXCEPTION-FILE                  0
FUNCTION EXCEPTION-FILE-N                0
FUNCTION EXCEPTION-LOCATION              0
FUNCTION EXCEPTION-LOCATION-N            0
FUNCTION EXCEPTION-STATEMENT             0
FUNCTION EXCEPTION-STATUS                0
FUNCTION EXP                             1
FUNCTION EXP10                           1
FUNCTION FACTORIAL                       1
FUNCTION FORMATTED-CURRENT-DATE          1
FUNCTION FORMATTED-DATE                  2
FUNCTION FORMATTED-DATETIME              Variable
FUNCTION FORMATTED-TIME                  Variable
FUNCTION FRACTION-PART                   1
FUNCTION HIGHEST-ALGEBRAIC               1
FUNCTION INTEGER                         1
FUNCTION INTEGER-OF-BOOLEAN              1
FUNCTION INTEGER-OF-DATE                 1
FUNCTION INTEGER-OF-DAY                  1
FUNCTION INTEGER-OF-FORMATTED-DATE       2
FUNCTION INTEGER-PART                    1
FUNCTION LENGTH                          1
FUNCTION LENGTH-AN                       1
FUNCTION LOCALE-COMPARE                  Variable
FUNCTION LOCALE-DATE                     2
FUNCTION LOCALE-TIME                     2
FUNCTION LOCALE-TIME-FROM-SECONDS        2
```

```
FUNCTION LOG                            1
FUNCTION LOG10                          1
FUNCTION LOWER-CASE                     1
FUNCTION LOWEST-ALGEBRAIC               1
FUNCTION MAX                            Variable
FUNCTION MEAN                           Variable
FUNCTION MEDIAN                         Variable
FUNCTION MIDRANGE                       Variable
FUNCTION MIN                            Variable
FUNCTION MOD                            2
FUNCTION MODULE-CALLER-ID               0
FUNCTION MODULE-DATE                    0
FUNCTION MODULE-FORMATTED-DATE          0
FUNCTION MODULE-ID                      0
FUNCTION MODULE-PATH                    0
FUNCTION MODULE-SOURCE                  0
FUNCTION MODULE-TIME                    0
FUNCTION MONETARY-DECIMAL-POINT         0
FUNCTION MONETARY-THOUSANDS-SEPARATOR   0
FUNCTION NATIONAL-OF                    Variable
FUNCTION NUMERIC-DECIMAL-POINT          0
FUNCTION NUMERIC-THOUSANDS-SEPARATOR    0
FUNCTION NUMVAL                         1
FUNCTION NUMVAL-C                       2
FUNCTION NUMVAL-F                       1
FUNCTION ORD                            1
FUNCTION ORD-MAX                        Variable
FUNCTION ORD-MIN                        Variable
FUNCTION PI                             0
FUNCTION PRESENT-VALUE                  Variable
FUNCTION RANDOM                         Variable
FUNCTION RANGE                          Variable
FUNCTION REM                            2
FUNCTION REVERSE                        1
FUNCTION SECONDS-FROM-FORMATTED-TIME    2
FUNCTION SECONDS-PAST-MIDNIGHT          0
FUNCTION SIGN                           1
FUNCTION SIN                            1
FUNCTION SQRT                           1
FUNCTION STANDARD-COMPARE               Variable
FUNCTION STANDARD-DEVIATION             Variable
FUNCTION STORED-CHAR-LENGTH             1
FUNCTION SUM                            Variable
FUNCTION TAN                            1
FUNCTION TEST-DATE-YYYYMMDD             1
FUNCTION TEST-DAY-YYYYDDD               1
FUNCTION TEST-FORMATTED-DATETIME        2
FUNCTION TEST-NUMVAL                    1
FUNCTION TEST-NUMVAL-C                  2
FUNCTION TEST-NUMVAL-F                  1
FUNCTION TRIM                           2
FUNCTION UPPER-CASE                     1
FUNCTION VARIANCE                       Variable
FUNCTION WHEN-COMPILED                  0
FUNCTION YEAR-TO-YYYY                   Variable
======================================== ==========
```

GnuCOBOL

```
======================================== ==========
```

```
FUNCTION CONCATENATE                    Variable
FUNCTION SUBSTITUTE                     Variable
FUNCTION SUBSTITUTE-CASE                Variable
======================================= ==========
```

( ) FUNCTION .

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
REPOSITORY.
    FUNCTION ALL INTRINSIC.
```

ALL INTRINSIC PROCEDURE DIVISION FUNCTION .

LENGTH . LENGTH . GnuCOBOL LENGTH OF , OF .

Examples

```
01 some-string PIC X(32).

...

MOVE "    a string literal" TO some-string

DISPLAY ":" some-string ":"
DISPLAY ":" FUNCTION TRIM(some-string) ":"
DISPLAY ":" FUNCTION TRIM(some-string LEADING) ":"
DISPLAY ":" FUNCTION TRIM(some-string TRAILING) ":"
```

```
:    a string literal           :
:a string literal:
:a string literal           :
:    a string literal:
```

```
MOVE FUNCTION UPPER-CASE("Hello World!") TO SOME-FIELD
DISPLAY SOME-FIELD
```

```
HELLO WORLD!
```

LOWER-CASE

```
MOVE FUNCTION LOWER-CASE("HELLO WORLD!") TO SOME-FIELD
DISPLAY SOME-FIELD
```

```
hello world!
```

: https://riptutorial.com/ko/cobol/topic/7580/-

49:

DATA DIVISION COBOL . . , , .

Examples

COBOL .

```
DATA DIVISION.
FILE SECTION.
FD SAMPLE-FILE
01 FILE-NAME PIC X(20).
WORKING-STORAGE SECTION.
01 WS-STUDENT PIC A(10).
01 WS-ID PIC 9(5).
LOCAL-STORAGE SECTION.
01 LS-CLASS PIC 9(3).
LINKAGE SECTION.
01 LS-ID PIC 9(5).
```

01 .

. . .

- 01 : . 01.

```
DATA DIVISION.
WORKING-STORAGE SECTION.
01 WS-NAME               PIC X(25).   ---> ELEMENTARY ITEM
01 WS-SURNAME            PIC X(25).   ---> ELEMENTARY ITEM
01 WS-ADDRESS.                        ---> GROUP ITEM
   05 WS-HOUSE-NUMBER    PIC 9(3).    ---> ELEMENTARY ITEM
   05 WS-STREET          PIC X(15).   ---> ELEMENTARY ITEM
```

- 02 ~ 49 :
- 66 :
- 77 : .
- 88 : 88 COBOL IF . 88 . PICTURE . 88 .

```
 01 YES-NO PIC X.
 88 ANSWER-IS-YES VALUE "Y".
```

YES-NO "Y" .

```
 IF YES-NO = "Y"
 IF ANSWER-IS-YES
```

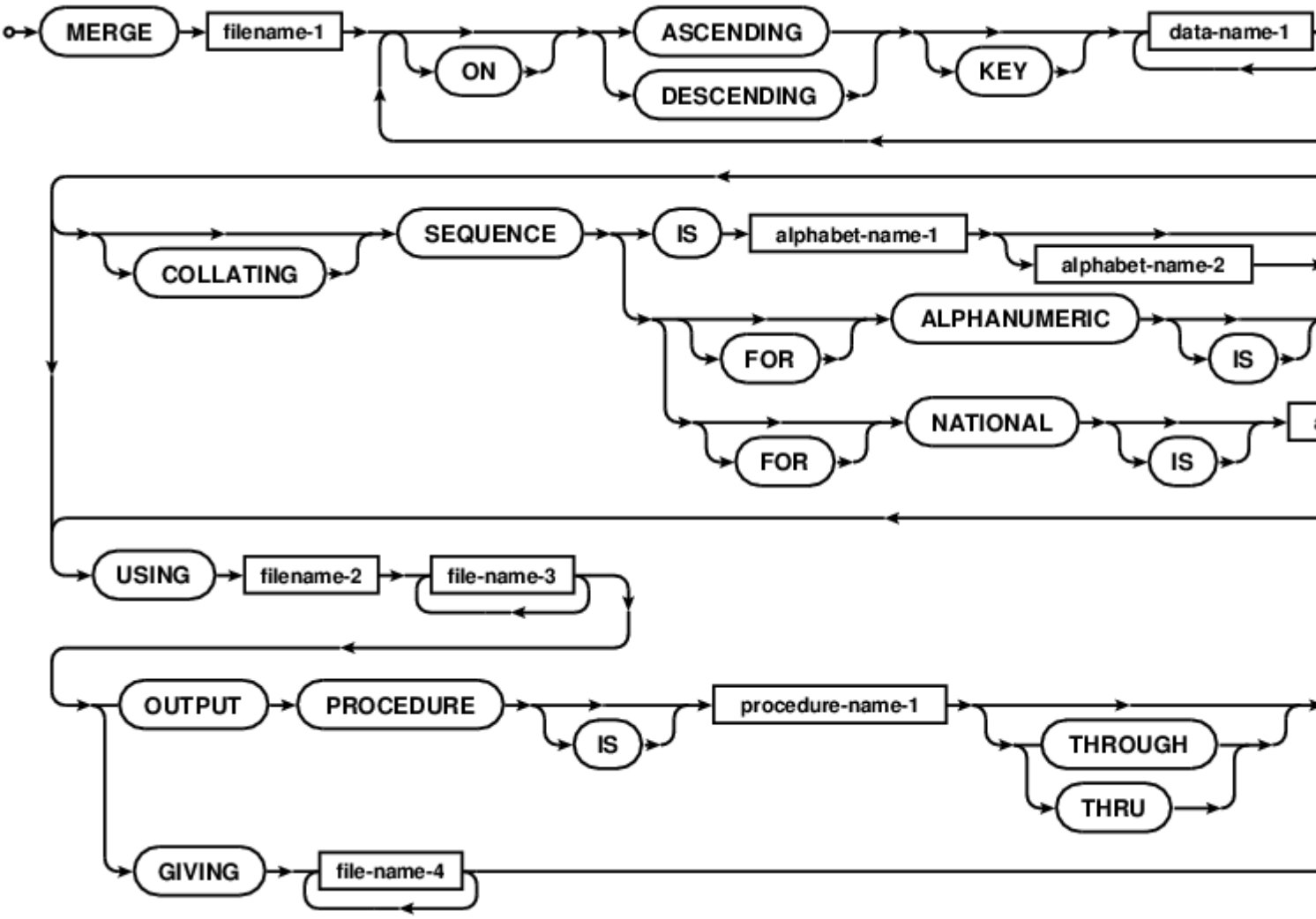
88 .

PICTURE CLAUSE . ( ) .

: https://riptutorial.com/ko/cobol/topic/10859/-

50:

MERGE COBOL . RELEASE OUTPUT PROCEDURE GIVING COBOL .



Examples

MERGE

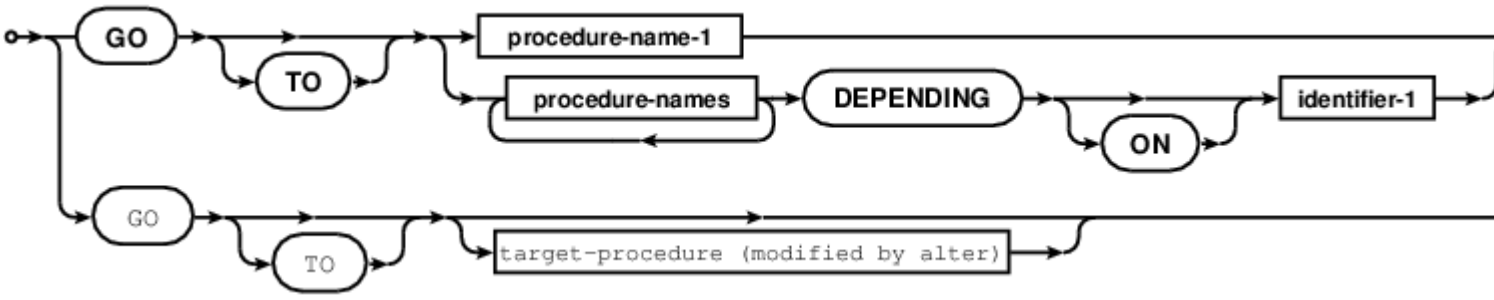
```
GCobol >>SOURCE FORMAT IS FIXED
      *> ****************************************************************
      *> Purpose:   Demonstrate a merge pass
      *> Tectonics: cobc -x gnucobol-merge-sample.cob
      *> ****************************************************************
       identification division.
       program-id. gnucobol-merge-sample.

       environment division.
       configuration section.
       repository.
           function all intrinsic.

files  input-output section.
       file-control.
```

```
           select master-file
               assign to "master-sample.dat"
               organization is line sequential.

           select eastern-transaction-file
               assign to "east-transact-sample.dat"
               organization is line sequential.

           select western-transaction-file
               assign to "west-transact-sample.dat"
               organization is line sequential.

           select merged-transactions
               assign to "merged-transactions.dat"
               organization is line sequential.

           select working-merge
               assign to "merge.tmp".

data   data division.
       file section.
       fd master-file.
          01 master-record     pic x(64).

       fd eastern-transaction-file.
          01 transact-rec      pic x(64).

       fd western-transaction-file.
          01 transact-rec      pic x(64).

       fd merged-transactions.
          01 new-rec           pic x(64).

       sd working-merge.
          01 merge-rec.
             02 master-key     pic 9(8).
             02 filler         pic x.
             02 action         pic xxx.
             02 filler         PIC x(52).

      *> *****************************************************************
      *> not much code
      *>     trick.  DEP, CHQ, BAL are action keywords.  They sort
      *>     descending as DEP, CHQ, BAL, so main can do all deposits,
      *>     then all withdrawals, then balance reports, for each id.
      *> *****************************************************************
code   procedure division.
       merge working-merge
           on ascending key master-key
              descending key action
           using eastern-transaction-file,
                 western-transaction-file,
                 master-file
           giving merged-transactions
done   goback.
       end program gnucobol-merge-sample.
```

: https://riptutorial.com/ko/cobol/topic/7183/-

51: GO

`GO TO` . COBOL , `GO` .



Examples

GO

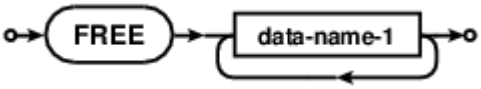
```
GO TO label

GO TO label-1 label-2 label-3 DEPENDING ON identifier-1

GO TO label OF section

GO.
```

`ALTER` `label` .

GO : https://riptutorial.com/ko/cobol/topic/7163/-go

52:

`FREE` POINTER BASED . FREE .



Examples

```
01 field-1 PIC X(80) BASED.

ALLOCATE field-1

*> use field-1

FREE field-1

*> further use of field-1 will cause memory corruption
```

: https://riptutorial.com/ko/cobol/topic/7162/-

S. No		Contributors
1	cobol	4444, Abhishek Jain, Bharat Anand, Brian Tiffin, Community, Joe Zitzelberger, ncmathsadist
2	ACCEPT	Brian Tiffin
3	ADD	Brian Tiffin
4	ALLOCATE	Brian Tiffin
5	ALTER	Brian Tiffin
6	CALL	4444, Bill Woodger, Brian Tiffin, infoRene, Jeffrey Ranney, Joe Zitzelberger, Simon Sobisch
7	CANCEL	Brian Tiffin
8	cobol ?	Bruce Martin, Bulut Colak
9	COMMIT	Brian Tiffin
10	COMPUTE	Brian Tiffin
11	CONTINUE	Brian Tiffin
12	COPY	Brian Tiffin
13	DELETE	Brian Tiffin
14	DISPLAY	Brian Tiffin
15	DIVIDE	Brian Tiffin
16	EVALUATE	Brian Tiffin
17	EXIT	Brian Tiffin
18	GENERATE	Brian Tiffin
19	GNU / Linux GnuCOBOL	Brian Tiffin
20	GOBACK	Brian Tiffin
21	IF	Brian Tiffin
22	INITIALIZE	Brian Tiffin

23	INITIATE	Brian Tiffin
24	INSPECT	Brian Tiffin
25	MOVE	Brian Tiffin
26	MULTIPLY	Brian Tiffin
27	OPEN	Brian Tiffin
28	PERFORM	Brian Tiffin
29	READ	Brian Tiffin
30	RELEASE	Brian Tiffin
31	REPLACE	Brian Tiffin
32	RETURN	Brian Tiffin
33	REWRITE	Brian Tiffin
34	SEARCH	Brian Tiffin
35	SET	Brian Tiffin
36	SORT	Brian Tiffin
37	START	Brian Tiffin
38	STOP	Brian Tiffin
39	STRING	Brian Tiffin
40	SUBTRACT	Brian Tiffin
41	SUPPRESS	Brian Tiffin
42	TERMINATE	Brian Tiffin
43	UNLOCK	Brian Tiffin
44	UNSTRING	Brian Tiffin
45	USE	Brian Tiffin
46	WRITE	Brian Tiffin
47		Jeffrey Ranney, Michael Simpson
48		Brian Tiffin, MC Emperor

49		Bulut Colak
50		Brian Tiffin
51	GO	Brian Tiffin
52		Brian Tiffin