



















```
>>DISPLAY string ...
    Write string to standard error as a warning message.

>>SOURCE format
    format may be one of:
    FIXED
        Source conforms to COBOL fixed-form reference format with unlimited line length.
    FREE
        Source conforms to COBOL free-form reference format. ‘*’ at the beginning of a
        line is recognized as a comment.

>>FLAG-02
    Not implemented.
>>FLAG-85
    Not implemented.
>>FLAG-NATIVE-ARITHMETIC
    Not implemented.
>>LEAP-SECOND
    Not implemented.
>>LISTING
    Not implemented.
>>PAGE
    Not implemented.
>>PROPAGATE
    Not implemented.
>>PUSH directive
>>POP directive
    With PUSH, push CDF state onto a stack. With POP, return to the prior pushed state.
    directive may be one of
    CALL-CONVENTION
COBOL-WORDS
DEFINE
SOURCE FORMAT
TURN
>>TURN [ec [file ...] ...]CHECKING {[ON] [[WITH] LOCATION] | OFF}
    Enable (or, with OFF, disable) exception condition ec optionally associated with the file
    connectors file. If LOCATION is specified, gcobol reports at runtime the source
    filename and line number of the statement that triggered the exception condition.
```

### Feature-set Variables

Some command-line options affect CDF *feature-set* variables that are special to **gcobol**. They can be set and tested using **>>DEFINE** and **>>IF**, and are distinguished by a leading ‘%’ in the name, which is otherwise invalid in a COBOL identifier:

#### %EBCDIC-MODE

is set by `-finternal-ebcdic`.

#### %64-BIT-POINTER

is implied by `-dialect ibm`.

To set a feature-set variable, use

```
>>SET feature [AS] {ON | OFF}
```

If *feature* is **%EBCDIC-MODE**, the directive must appear before **PROGRAM-ID**.

To test a feature-set variable, use

```
>>IF feature DEFINED
```

### Intrinsic functions

**gcobol** implements all intrinsic functions defined by ISO/IEC 1989:2023, plus a few others. They are listed alphabetically below.

ABS ACOS ANNUITY ASIN ATAN

BASECONVERT BIT-OF BIT-TO-CHAR BOOLEAN-OF-INTEGER BYTE-LENGTH









### Beyond COBOL/85

**gcobol** increasingly implements ISO/IEC 1989:2023. For example, **DECLARATIVES** is not tested by CCVS-85, but are implemented by **gcobol**. Similarly, Exception Conditions were not defined in 1985, and **gcobol** contains a growing number of them.

The authors are well aware that a complete, pure COBOL-85 compiler won't compile most existing COBOL code. Every vendor offered (and offers) extensions, and most environments rely on a variety of preprocessors and ancillary systems defined outside the standard. The express goal of adding an ISO COBOL front-end to GCC is to establish a foundation on which any needed extensions can be built.

### HISTORY

COBOL, the language, may well be older than the reader. To the author's knowledge, free COBOL compilers first began to appear in 2000. Around that time an earlier COBOL for GCC project *cobolforgcc*: <https://cobolforgcc.sourceforge.net/> met with some success, but was never officially merged into GCC.

This compiler, **gcobol**, was begun by *COBOLworx*: <https://www.cobolworx.com/> in the fall of 2021. The project announced a complete implementation of the core language features in December 2022.

### AUTHORS

James K. Lowden

([jklowden@cobolworx.com](mailto:jklowden@cobolworx.com)) is responsible for the parser.

Robert Dubner

([rdubner@cobolworx.com](mailto:rdubner@cobolworx.com)) is responsible for producing the GIMPLE tree, which is input to the GCC back-end.

### CAVEATS

- **gcobol** has been tested only on x64 and Apple M1 processors running Linux in 64-bit mode.
- The I/O support has not been extensively tested, and does not implement or emulate many features related to VSAM and other mainframe subsystems. While LINE-SEQUENTIAL files are ordinary text files that can be manipulated with standard utilities, INDEXED and RELATIVE files produced by **gcobol** are not compatible with that of any other COBOL compiler. Enhancements to the I/O support will be readily available to the paying customer.