By Carson Crockett

I. COBOL History

A. COBOL is a domain specific language, specifically for the business domain (1)

Domain specific languages are targeted to a specific problem rather than being built to serve a robust purpose.

a) Examples include SQL, CSS, make, etc. (2)

This means it was built to handle heterogeneous information and perform accurate arithmetic operations on the data

B. COBOL is not object oriented like more modern languages such as Java but instead functions more like SQL to operate on the data. COBOL is usually run on a server or mainframe.

C. At the time it was built for readability so “non-technical” employees could use it. Although it seems hard to read at first once you understand the structure it becomes simpler.

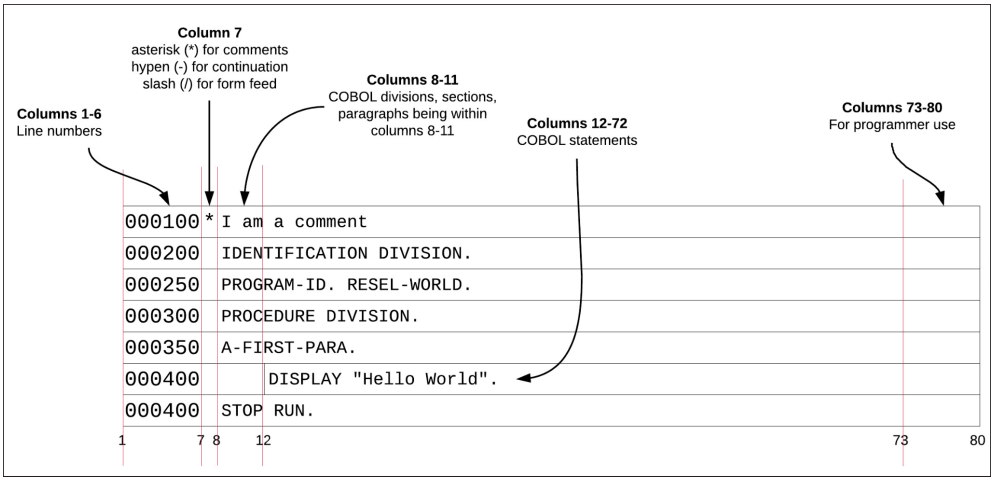
It was also built in 1959 by the DOD when the hardware available was extremely limited and needed to be written on punch cards (1)

II. Reading COBOL and understanding it’s structure

A. Columns

COBOL’s structure is dependent on the column in which a line starts

a) This is an artifact of needing to write COBOL on punch cards



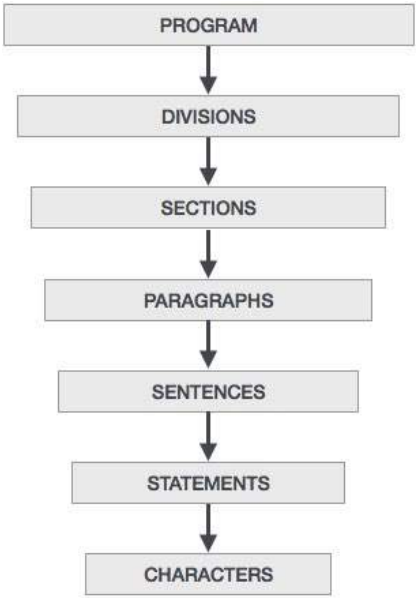
b) The picture above shows what columns do what

(1) 1-6 store the line number, 7 holds an asterisk to indicate comments, 8-11 is where divisions, sections, and paragraphs are written, 12-72 are statements of code, and the rest is ignored

c) This structure allows COBOL to be easily written in read in any basic text editor. Many editors like VScode will add similar lines to the one above to make it simple (1)

B. Divisions, sections, paragraphs, sentences, statements and characters.

1. A COBOL code is divided into divisions, sections, paragraphs, sentences, statements, and characters. The definition is decided by both the written text and the column



2. (5)
    a) This is also an artifact from the hardware available at the time. It was required for the computers at the time to understand what it was reading as it began reading the card.
    b) Divisions and sections were used for the computer to know what resources were required
3. Here's an example

```
IDENTIFICATION DIVISION.
 PROGRAM-ID. HELLO.
PROCEDURE DIVISION.
    DISPLAY "Hello, world".
 END PROGRAM HELLO.
```

    a) Here we see two divisions, the identification and procedure. The identification division is where the computer would read the name of the program. The procedure division is where the program would execute. In this instance it would print “Hello, world”
4. Defining Variables

```
01 Customer.
    02 Ident    PIC 9(3).
    02 CustName PIC X(20).
    02 DateOfBirth.
       03 MOB  PIC 99.
```

03 DOB PIC 99.
03 YOB PIC 9(4).

a) Here we define a record called Customer
    (1) The record has several attributes, starting with Ident, for identifier. We see it is identified as PIC 9(3) (1)
    (2) PIC is the abbreviation for PICTURE Clause. All you need to know at the moment is that this is defining a numeric variable.
        (a) A PIC can have a 9, V, X, or S to indicate what type of variable it is. 9 is a whole number, V indicates decimal places, and S is the sign. An X indicates an alphanumeric field
        (b) So PIC 9(3) means that the Ident attribute is 3 whole numbers (3)
    (3) We also see MOB which is defined as 99, this indicates two whole numbers.
        (a) This shows too different ways of defining the size of the variable. For the previous example we could've also said PIC 999 instead of PIC 9(3) (1)

5. Divisions
    a) There are four divisions in COBOL which make up a program
        (1) Identification, Environment, Data, and Procedure
    b) Identification Division
        (1) This is the first division in a COBOL file and required for execution. This division defines information like "author name, date of execution, date of writing the code, installation, etc"
        (2) The program name and id are actual statements the rest are comments
    c) Environment Division
        (1) Used to specify features of the program and the environment the program is written and executed in
        (2) Divided into the configuration and Input-Output Section
            (a) Configuration - "Identifies the computer used for compiling programs"
            (b) Input-Output Section - "Identifies the files and the I/O resources used by the program"
    d) Data Division
        (1) Used to declare variables and parameters of the program

(2) Optional unless the program manipulates any constant user-defined data
(3) Can be broken down into the File Section, Working Storage Section, and Linkage Section
(a) File Section - describes data that is sent to or received from the peripherals
(b) Working Storage - Defines general variables
(c) Linkage Section - Establishes a link between two programs, an optional section mainly used in subprograms

e) Procedure Division
(1) Where business logic is actually written
(2) Contains user defined sections, paragraphs, sentences, statements, clauses, and verbs
(3) Needs at least one paragraph, sentence, and statements. Section(s) is optional (4)

6. Paragraphs, Statements, Sentences, and Characters
a) A section is a division of the program logic. It defines a group of code working towards a single goal within the overall goal of the program
b) Sections consist of paragraphs which are further divisions of sections. A block of code consisting of one or more sentences or statements
(1) Similar to a function, “a block of logical code which can be called by other parts of the program”
c) Sentences are a sequence of two or more statements which are terminated with a period.
(1) It is a grouping of statements into one logical block
d) Statements is a line of COBOL operation. For example DISPLAY “Hello World”.
e) Characters are what they sound like. Strings of letters or numbers which define variables or keywords.
(1) In the above example DISPLAY would be an example of Characters (6)

Works Cited

(1) https://monadical.com/posts/cobol.html
(2) https://martinfowler.com/bliki/DomainSpecificLanguage.html
(3) https://www.ibm.com/docs/en/cobol-zos/6.4?topic=entry-picture-clause
(4) https://www.geeksforgeeks.org/cobol-divisions/
(5) https://www.tutorialspoint.com/cobol/cobol_program_structure.htm
(6) https://www.mainframestechhelp.com/tutorials/cobol/program-structure.htm