COBOL Cheatsheet

A quick reference guide to COBOL syntax, data types, control structures, and other essential elements.

COBOL Fundamentals

Program Structure

COBOL programs are divided into four divisions:

1. **IDENTIFICATION DIVISION:** Identifies the program.
2. **ENVIRONMENT DIVISION:** Describes the computer environment.
3. **DATA DIVISION:** Defines the data used by the program.
4. **PROCEDURE DIVISION:** Contains the program logic.

Example:

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  SAMPLE-PROGRAM.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-370.
OBJECT-COMPUTER. IBM-370.
DATA DIVISION.
FILE SECTION.
WORKING-STORAGE SECTION.
PROCEDURE DIVISION.
   DISPLAY 'Hello, World'.
   STOP RUN.
```

Data Types

`PIC X(n)`	Alphanumeric data, `n` is the length.
`PIC 9(n)`	Numeric data, `n` is the number of digits.
`PIC A(n)`	Alphabetic data, `n` is the length.
`PIC S9(n)`	Signed numeric data, `n` is the number of digits.
`PIC V`	Implied decimal point.

Data Definition

Data is defined in the `DATA DIVISION` within the `WORKING-STORAGE SECTION` or `FILE SECTION`.

Example:

```
WORKING-STORAGE SECTION.
01  CUSTOMER-NAME PIC X(30).
01  CUSTOMER-AGE  PIC 9(02).
01  PI  PIC 9V99 VALUE 3.14.
```

Control Structures

IF Statement

Conditional execution based on a condition.

Syntax:

```
IF condition THEN
    statements
ELSE
    statements
END-IF.
```

Example:

```
IF CUSTOMER-AGE > 18 THEN
    DISPLAY 'Customer is an adult'
ELSE
    DISPLAY 'Customer is a minor'
END-IF.
```

EVALUATE Statement

Multi-way branch based on the value of a variable.

Syntax:

```
EVALUATE variable
    WHEN value1
        statements
    WHEN value2
        statements
    WHEN OTHER
        statements
END-EVALUATE.
```

Example:

```
EVALUATE CUSTOMER-CODE
    WHEN 1
        DISPLAY 'Premium Customer'
    WHEN 2
        DISPLAY 'Gold Customer'
    WHEN OTHER
        DISPLAY 'Standard Customer'
END-EVALUATE.
```

PERFORM Statement

`PERFORM paragraph-name.`	Executes a paragraph once.
`PERFORM paragraph-name n TIMES.`	Executes a paragraph `n` times.
`PERFORM paragraph-name UNTIL condition.`	Executes a paragraph until the condition is true.
`PERFORM paragraph-name VARYING identifier FROM initial BY increment UNTIL condition.`	Executes a paragraph, varying a counter from an initial value by an increment until a condition is met.

File Handling

File Section

Describes the structure and organization of data files.

Example:

```
FILE SECTION.
FD  CUSTOMER-FILE.
01  CUSTOMER-RECORD.
    05  CUSTOMER-ID   PIC 9(5).
    05  CUSTOMER-NAME PIC X(30).
```

File Operations

`OPEN INPUT file-name.`	Opens a file for reading.
`OPEN OUTPUT file-name.`	Opens a file for writing.
`READ file-name INTO record-name AT END statements.`	Reads a record from a file.
`WRITE record-name FROM variable-name.`	Writes a record to a file.
`CLOSE file-name.`	Closes a file.

Example: Reading a File

```
PROCEDURE DIVISION.
    OPEN INPUT CUSTOMER-FILE.
    PERFORM UNTIL END-OF-FILE
        READ CUSTOMER-FILE INTO
CUSTOMER-RECORD
            AT END
                MOVE 'Y' TO END-OF-FILE
            NOT AT END
                DISPLAY CUSTOMER-ID,
CUSTOMER-NAME
        END-READ.
    END-PERFORM.
    CLOSE CUSTOMER-FILE.
    STOP RUN.
```

String Handling & Tables

String Manipulation

`STRING ... DELIMITED BY ... INTO ...`	Concatenates strings into a single string.
`UNSTRING ... DELIMITED BY ... INTO ...`	Splits a string into multiple strings based on delimiters.
`INSPECT ... REPLACING ...`	Replaces characters or substrings within a string.

Tables (Arrays)

Tables are defined using the `OCCURS` clause.

Example:

```
01  EMPLOYEE-TABLE.
    05  EMPLOYEE-RECORD OCCURS 10 TIMES.
        10  EMPLOYEE-ID   PIC 9(5).
        10  EMPLOYEE-NAME PIC X(30).
```

Accessing table elements:

```
DISPLAY EMPLOYEE-NAME(5).
```

Example of Table Processing:

```
PROCEDURE DIVISION.
    PERFORM VARYING I FROM 1 BY 1 UNTIL
I > 10
        DISPLAY EMPLOYEE-NAME(I)
    END-PERFORM.
```

Search Statement

The `SEARCH` statement is used to find a specific element in a table.

Example:

```
SEARCH EMPLOYEE-TABLE
    AT END
        DISPLAY 'Employee not found'
    WHEN EMPLOYEE-ID(I) = SEARCH-ID
        DISPLAY EMPLOYEE-NAME(I)
END-SEARCH.
```