**LANGUAGE OVERVIEW**

LexiCom Compiler’s Lexis Programming Language is derived from C, VB.net and Swift Language's where the advantages involve its concise and simple syntax.

*Lexis* is a synonym of the word *Lexicon* which means the words used in a language or by a person or group of people. LexiCom comes from the English Word *Lexicon* and *Compiler* which means a program that reads a program written in one language and translates it into an equivalent program in another language.

**GENERAL RULES**

1. The program must have a *Lead* function (Main Function). The word *Lead* is followed by a colon “**:**”.
2. The *Lead* function must begin with the word *Start* and end with the word *End.*
3. *LexiCom* allows any number of whitespaces between two words. Spaces, indentations, and newlines are considered as whitespace.
4. *LexiCom* is a case sensitive Programming Language. Thus, *Automata* and *AuToMaTa* are two different identifiers.
5. Declarations, such as variable, object, and array, can be made locally or globally. Declaring constants and *Task*s (sub-function) can only be made globally.
6. Declaring variable, constant, object, task, and array can be done in any order.
7. *Var* is used for declaring a variable and *Let* for constants.
8. *Task* definitions (Sub function) are placed after the *Lead* function. To use these tasks they must first be declared before the *Lead* function.
9. *Tilde* “~” is used as a negative sign for negative integer and double values.
10. Reserved words must not be used a s an identifier.
11. Identifiers must start with a capital letter. It must not begin with a special character, a small letter or a number.
12. Identifiers must have a minimum of 1 character, and a maximum of 19 characters (Alphanumeric and underscore)
13. Underscores “**\_**” are allowed in naming an Identifier as long as it is not the first character.
14. *Read* (input) and *Say* (output)are used as I/O statements.
15. *If*, *If – Otherwise*, *If – Or*, *If – Or – Otherwise*, and *Option – State* are used as conditional statements.
16. *Do*, *Do – Until*, *Until,* and *For* are used as looping statements.
17. Statements such as, I/O, conditional, looping, assignment, and mathematical statements are terminated with a period “**.**”.
18. The whole program must be terminated by the number sign “#”.
19. Statements after the number sign “#” will not be read.

**STRUCTURE OF THE PROGRAM**

Before we introduce the basic building blocks of LexiCom programming language, let us look a bare minimum LexiCom program structure so that we can fully understand the Language in the next pages

LexiCom Hello World Example

A LexiCom program basically consists of the following parts:

* Declarations:
  + Task Declarations
  + Object Declarations
  + Global Variable Declarations
* Lead Function
* Statements
* Task Definition
* Statements

|  |
| --- |
| ***[<Task declaration>]***  ***[<Object declaration>] [<Global variable declaration>]***  ***Lead: Start  [Statement\_1.]***  ***[Statement\_2.]  …***  ***[Statement\_n.] End. [<Task definition>***  ***Start  [Statement\_1.]***  ***[Statement\_2.]  …***  ***[Statement\_n.] End.] #*** |

Let us look at a simple code that would print the words *“Hello World”:*

**Lead:**

**Start**

**Say “Hello World!”.**

**End.#**

Let us look at the various parts of the program:

1. The first line of the program, *Lead:* is the main function where the program execution begins.
2. The next line *Start,* it serves as the open curly brace “{” in C.
3. The next line *Say “…”* which displays *“Hello World!”* on the screen while execution of the program.
4. The last line *End.#* terminates the whole program.