JOHN MCCARTHY - CREATOR OF LISP

Biography Of John McCarthy

John McCarthy was born in 1927 in Boston, Massachusetts. His parents, John Patrick and IDA Glatt McCarthy, were immigrants, his father was Lithuanian, and his mother was Jewish. Young John showed early success in mathematics – so, as a schoolboy, he independently mastered mathematics from University textbooks. Later, as a student at the California Institute of technology, McCarthy was able to jump right through the first two years of study, but McCarthy's most important achievement is the invention of the Lisp programming language in 1958.

Lisp creation process

* In 1956-1959, John McCarthy, along with students at the Massachusetts Institute of technology (MIT), developed a new elegant language for working with lists, which he gave the name LISP. And many students have made a significant contribution to this development. In particular, D. Edwards programmed the process of clearing memory of unnecessary arrays, the so-called garbage collection, which is a fundamental concept in LISP.

The name of the language is derived from the abbreviation of the English words LISP Processing, which means "list processing". LISP programs and data exist in the form of symbolic expressions that are stored as list structures. LISP deals with two types of objects: atoms and lists. Atoms are symbols used to identify objects that can be numeric or symbolic (concepts, materials, people, etc.). A list is a sequence of zero or more elements enclosed in parentheses, each of which is either an atom or a list.

Advantages Of Lisp

- In Lisp, programmers usually work in a special incremental interactive programming environment called the eval-print read cycle (REPL). This means that the system continuously reads expressions entered by the user, evaluates them, and prints the results.

- Common Lisp was developed from the ground up to create large, complex, and long-running applications and thus supports hot-swapping software - the code of an executable program can be changed without having to interrupt it.

- The Lisp programming language invented exception handling and has an error handling tool called a "condition system" that goes far beyond the limits of most other languages.

- Common Lisp implementations usually come with a complex "external function interface" (FFI) that allows direct access to Lisp code written in C or C++, and sometimes to Java code. This allows programmers to use libraries written in other languages, which makes these libraries dependent on Lisp. Lisp gives programmers the ability to implement complex data operations, and mathematical constructions in an expressive and natural idiom.

Basic Lisp concept
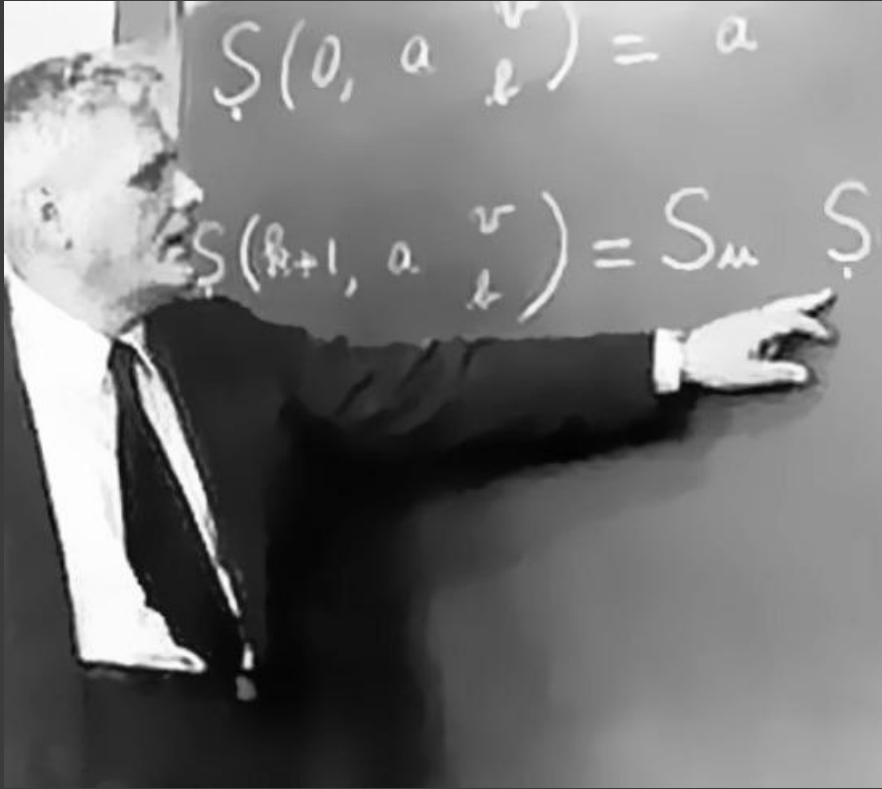
Lisp is the earliest representative of the functional programming language paradigm from scratch. Unlike procedural and object-oriented languages, whose theoretical model is a Turing machine, the theoretical model for calculating Lisp is the Lambda calculus developed by the Alonso Church. This difference can be understood in the following ways:

- In a procedural language the code is working with data.
- In object-oriented languages, objects encapsulate code and data by interacting with each other.
- In functional languages, data is passed through functions, but does not have a separate existence of its own.
- In pure functional languages, this is ephemeral.
- However, most functional languages (including Lisp) provide a mechanism for storing certain data even when it does not pass through functions. This takes the form of what Common Lisp calls "special variables", and is equivalent to "global variables" in other programming languages. That is, although it is a functional language, it is not purely functional, although it can be used as one, avoiding imperative constructs.

The interactivity and flexibility of   Lisp

The interactivity and flexibility of LispThe speed of Lisp languages is usually equivalent to the interpreted languages. This is also the source of the old myth that if it is interpreted, it must therefore be slow, but it is not. Compilers for Lisp have been around since 1959, and now all major common Lisp implementations can be compiled directly with machine code, which is often the same as C code.

- For example, the CL-PPCRE expression library written in Common Lisp is faster than the Perl regular expression engine on some tests in C, and learning Lisp is much easier. Programmers who use interpreted languages such as Python or Perl should resort to writing in C /C ++ for critical parts of their code for convenience and flexibility, and have unique opportunities to do so. This was shown with direct benchmarking by the Creator of the R programming language, Ross Ihaka, who provided tests demonstrating that the optional Lisp type Declaration and machine code compiler allow them to be used 380 times faster than R and 150 times faster than Python.