Turiba University

Robin Singh

History of Programming Languages

PROFESSIONAL BACHELOR THESIS

Computer Systems

Author: **Robin Singh**
***Thesis* Advisor:** **Jānis Pekša**

Riga, 2025

Foundations & Early Ideas

Programming Language is indeed the fundamental unit of today's tech world. It is considered as the set of commands and instructions that we give to the machines to perform a particular task. For example, if you give some set of instructions to add two numbers, then the machine will do it for you and tell you the correct answer accordingly.

In the computer world, we have about 500+ programming languages with having their own syntax and features. And if you type who's the father of the computer, then the search engine will show you the result as to Charles Babbage but the father of the computer didn't write the first code. It was Ada Lovelace who has written the first-ever computer programming language and the year was 1883.

Programming Languages

Ada Lovelace: the first programmer

It all started with the first programmer - Ada Lovelace. She was a prominent figure in computer science history who saw the true potential of machines. In the mid-1800s, Lovelace worked alongside Charles Babbage on the first computer, the Analytical Engine.

While Babbage designed the machine, Lovelace recognized that it could do more than simple calculations. Hence, she theorized that it could follow a sequence of instructions, much like modern programming.

Lovelace did not just predict the future; she also wrote the first algorithm for machine execution, making her the world's first programmer. This laid the groundwork for the concept of a computer language, proving that machines could be programmed to perform tasks beyond pure arithmetic.

The rise of high-level languages (1950s–1970s)

While punch cards and early machine code were the start, programmers needed to simplify the instructions. It led to the rise of high-level programming languages, a notable shift from the low-level assembly language.

From the 1950s to the 1970s, several groundbreaking languages were invented, laying the foundation for modern software development. Despite these languages being decades old, Many are still used today in various fields. Thus, let's explore the key languages of this era.

Fortran (1957) — The first high-level language

The first programming language designed for high-level scientific computing was Fortran (Formula Translation). Developed by John Backus and his team at IBM in 1957, it made computer programming more accessible to scientists and engineers.

It enabled programmers to write mathematical equations in a structured way, making it a game-changer for applications in physics, weather modeling, and engineering simulations.

Its efficiency and numerical computing capabilities made it a staple in the scientific community.

Lisp and functional programming (1958)

A year later, John McCarthy brought Lisp (List Processing) to the market as the first programming language designed for artificial intelligence. Unlike Fortran, which focused on number crunching, Lisp introduced powerful concepts like recursion and symbolic processing, which made it ideal for AI research.

Lisp pioneered functional programming and influenced many modern languages, including Python and JavaScript. Despite being over six decades old, Lisp has remained relevant, especially in AI and machine learning research.

COBOL (1959) — The business language

While Fortran and Lisp catered to science and AI, the business world needed a programming language optimized for handling large amounts of data and financial transactions. That is where COBOL (Common Business-Oriented Language) came in. COBOL was created in 1959 by a committee known as the Conference on Data Systems Languages (CODASYL).

Its goal was to standardize programming for government and financial institutions. Its simple, English-like syntax made it easier for non-technical professionals to understand and use. It remains critical in banking, insurance, and legacy systems.

C (1972) — The universal language

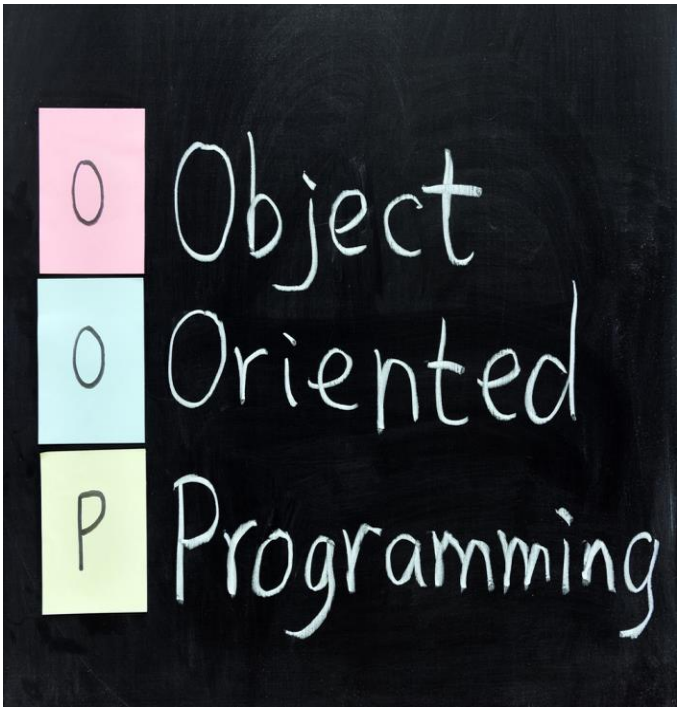
C was developed by Dennis Ritchie in 1972 at Bell Labs. It provides low-level memory access, like assembly language, but with the ease and structure of high-level programming. Its most significant impact was on operating systems, particularly UNIX.

Modern languages like C++, Objective C, and Java are rooted in C. C's efficiency, speed, and portability, making it one of the most widely used programming languages in system software, embedded programming, and modern applications.

The era of object-oriented programming

Over time, computers have become more powerful, and software systems have become more complex, leading to the need for better ways to manage code. This resulted in the development of Object-Oriented Programming (OOP).

This approach structured programs around "objects" rather than just functions and procedures, making designed software more reusable, scalable, and easier to maintain.



It led to creating some of the most influential high-level programming languages, still widely used today. Let’s examine the key languages that shaped this era.

Conclusion

- Programming languages have evolved from simple mechanical instructions to powerful tools that shape modern technology.

- The mid-1900s introduced high-level languages such as FORTRAN, COBOL, and LISP, each solving different problems and expanding what computers could do.

- Languages like C pushed computing forward by enabling faster, more efficient software and influencing many modern languages.

- The shift toward object-oriented programming helped developers manage growing software complexity through reusable and structured design.

- Understanding this history not only highlights how far technology has come but also shows how each generation of languages inspired the next.

Sources

- https://www.ramotion.com/blog/first-programming-language/
- https://www.geeksforgeeks.org/c/the-evolution-of-programming-languages/
- https://www.almabetter.com/bytes/articles/evolution-of-programming-languages