Turiba University

Bachelor's Degree in computer system

Bismaya Aneesh

Group : CSA1D1

History of Programming

Institution: Turiba University

Introduction

Programming is the process of writing instructions that a computer can understand and execute. The history of programming is a fascinating journey from simple mechanical calculations to the advanced artificial intelligence we use today. This report explains how programming languages and techniques have evolved over time.

Early Beginnings (1800s–1940s)

- Charles Babbage designed the *Analytical Engine* in the 1830s, considered the first mechanical computer.
- Ada Lovelace, often called the *first programmer*, wrote instructions for this machine to perform calculations the first algorithm ever designed for a computer.
- In the 1930s–1940s, scientists like Alan Turing developed the concept of a "universal machine" that could perform any computation the foundation of modern computers.

Early Beginnings (1800s–1940s)

- Charles Babbage designed the *Analytical Engine* in the 1830s, considered the first mechanical computer.
- Ada Lovelace, often called the *first programmer*, wrote instructions for this machine to perform calculations the first algorithm ever designed for a computer.
- In the 1930s–1940s, scientists like Alan Turing developed the concept of a "universal machine" that could perform any computation the foundation of modern computers.

The Second Generation (1950s–1960s): Assembly and Early High-Level Languages

- To make coding easier, **assembly language** was developed. It used simple words (like ADD, MOV) instead of binary numbers.
- Then came **high-level languages**, which looked more like English.
    - **FORTRAN (1957):** for scientific and engineering calculations.
    - **COBOL (1959):** for business and data processing.
    - **LISP (1958):** for artificial intelligence research.

The Third Generation (1960s–1970s): Structured Programming

- Programming languages became more powerful and readable.
- **C** (developed in 1972 by Dennis Ritchie) was one of the most influential languages.
- Structured programming introduced concepts like **loops**, **functions**, and **modularity**, making large programs easier to manage.
- **Pascal** and **BASIC** also became popular for education and general use.

The Fourth Generation (1980s–1990s): Object-Oriented Programming

- New languages focused on **objects**, which combine data and behavior.
- **C++** and **Java** were major innovations in this era.
- They allowed better reuse of code, which improved software development.
- At the same time, **databases** and **query languages** (like SQL) emerged.

The Fifth Generation (2000s–Present): Modern and AI Programming

- Internet growth led to new languages such as Python, JavaScript, PHP, and Ruby.
- These languages focus on simplicity, flexibility, and web development.
- Artificial intelligence and machine learning use modern languages like Python, R, and frameworks such as TensorFlow.
- Today, programmers use integrated development environments (IDEs) and AI tools to write and test code more efficiently.

Conclusion

From Ada Lovelace's early algorithms to the AI-driven coding of today, programming has evolved dramatically. Each generation of programming languages has made it easier, faster, and more powerful to communicate with computers. The history of programming shows how human creativity and technology together continue to shape our digital world.

References

1. Computer History Museum. (n.d.). *A Brief History of Programming Languages.* Retrieved from https://computerhistory.org
2. Britannica. (2024). *Computer Programming.* In *Encyclopaedia Britannica.* Retrieved from https://www.britannica.com/technology/computer-programming
3. Wexler, J. (2020). *History of Computing and Programming.* New York: TechWorld Press.
4. Turing, A. M. (1936). *On Computable Numbers, with an Application to the Entscheidungsproblem.* Proceedings of the London Mathematical Society.
5. Ritchie, D. M. (1993). *The Development of the C Language.* Bell Labs Technical Journal.

6. McCarthy, J. (1960). *Recursive Functions of Symbolic Expressions and Their Computation by Machine.* Communications of the ACM.