



**INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**
Bahia

**MINISTÉRIO DA EDUCAÇÃO INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E
TECNOLOGIA DA BAHIA - CAMPUS VALENÇA**

Aleff Yuri Santos Bulcão

Ronald Santos Belarmino

Hugo Silva dos Santos

Evolução das Linguagens de Programação

VALENÇA-BA

2023

Aleff Yuri Santos Bulcão
Ronald Santos Belarmino
Hugo Silva dos Santos

Evolução das Linguagens de Programação

Trabalho de Conclusão de Curso do Curso Integrado de Informática, do Instituto Federal de Educação, Ciência e Tecnologia da Bahia, como requisito parcial para a obtenção do título de Técnico em Informática.
Orientador: João Paulo Just Peixoto

VALENÇA
2023

Aleff Yuri Santos Bulcão
Ronald Santos Belarmino
Hugo Silva dos Santos

Evolução das Linguagens de Programação

Trabalho de Conclusão de Curso do Curso Integrado de Informática, do Instituto Federal de Educação, Ciência e Tecnologia da Bahia, como requisito parcial para a obtenção do título de Técnico em Informática.

RESULTADO: _____ NOTA: _____

Valença, _____ de _____ de _____.

Prof. João Paulo Just Peixoto (orientador)
Instituto Federal de Educação, Ciência e Tecnologia da Bahia - Campus Valença

Evolução das Linguagens de Programação

RESUMO

Este trabalho tem como objetivo contribuir para o conhecimento e a apreciação das linguagens de programação, bem como para a reflexão sobre os seus desafios e oportunidades futuras. A evolução das linguagens de programação representa um campo de estudo crucial para compreender a constante transformação da tecnologia da informação. Este artigo explora a trajetória histórica e as mudanças paradigmáticas ao longo das décadas. Começando com as primitivas linguagens de máquina da década de 1940, caracterizadas por representações binárias e dificuldades significativas, avançou-se para as linguagens de montagem na década de 1950, introduzindo mnemônicos e melhorando a legibilidade. A década seguinte viu o surgimento das linguagens de alto nível, como Fortran e COBOL, que abstraíram a programação, tornando-a mais acessível e portátil. As linguagens de consulta, como SQL, na década de 1970, impulsionaram o gerenciamento de bancos de dados relacionais. O período de 1980 a 1990 trouxe linguagens de programação lógica, exemplificadas pelo Prolog, que focaram em relações e regras. As linguagens orientadas a objetos, como C++ e Java, dominaram a década de 1990, introduzindo conceitos de objetos e reutilização de código. A partir dos anos 2000, linguagens modernas como Python, Ruby e JavaScript trouxeram inovações, como tipagem estática e programação funcional. Em conclusão, essa evolução reflete o constante esforço para simplificar a programação, tornando-a mais poderosa e acessível, e promete um futuro contínuo de inovação à medida que as demandas tecnológicas evoluem.

Palavras-chave: linguagens de programação, evolução, mudanças paradigmáticas.

ABSTRACT

This work aims to contribute to the knowledge and appreciation of programming languages, as well as to reflection on their future challenges and opportunities. The evolution of programming languages represents a crucial field of study to understand the constant transformation of information technology. This article explores the historical trajectory and paradigmatic changes over the decades. Starting with the primitive machine languages of the 1940s, characterized by binary representations and significant difficulties, it moved to assembly languages in the 1950s, introducing mnemonics and improving readability. The following decade saw the emergence of high-level languages such as Fortran and COBOL, which abstracted programming, making it more accessible and portable. Query languages such as SQL in the 1970s drove the management of relational databases. The period from 1980 to 1990 brought logical programming languages, exemplified by Prolog, which focused on relationships and rules. Object-oriented languages such as C++ and Java dominated the 1990s, introducing concepts of objects and code reuse. Starting in the 2000s, modern languages such as Python, Ruby and JavaScript brought innovations such as static typing and functional programming. In conclusion, this evolution reflects the constant effort to simplify programming, making it more powerful and accessible, and promises a continued future of innovation as technological demands evolve.

Key words: programming languages, evolution, paradigmatic changes.

SUMÁRIO

| | |
|--|----|
| RESUMO..... | 4 |
| LISTA DE QUADROS E TABELAS..... | 7 |
| LISTA DE FIGURAS E ILUSTRAÇÕES..... | 7 |
| 1. INTRODUÇÃO..... | 8 |
| 2. FUNDAMENTAÇÃO TEÓRICA..... | 8 |
| 3. OBJETIVOS GERAIS E ESPECÍFICOS..... | 10 |
| 4. METODOLOGIA..... | 10 |
| 5. ANÁLISE DA PESQUISA..... | 11 |
| 6. CONSIDERAÇÕES FINAIS..... | 15 |
| REFERÊNCIAS..... | 16 |

LISTA DE QUADROS E TABELAS

| | |
|--|----|
| Tabela 1 — Tipos de variável..... | 13 |
| Tabela 2 — Estruturas e sintaxe..... | 13 |
| Tabela 3 — Vantagens e desvantagens..... | 14 |

LISTA DE FIGURAS E ILUSTRAÇÕES

| | |
|---|----|
| Imagem 1 — Código “Hello, World!” em C++, Java e Python, respectivamente..... | 14 |
|---|----|

Evolução das Linguagens de Programação

1. INTRODUÇÃO

A evolução das linguagens de programação (LP) é um tema fascinante e de grande importância para a história da computação. Desde a criação da primeira linguagem de programação de alto nível, Plankalkül, em 1944–45, tem se testemunhado avanços significativos na evolução das linguagens de programação. Atualmente, existem centenas de linguagens de programação distintas, cada uma com suas próprias características e aplicações. Neste trabalho, será explorada a evolução das linguagens de programação, desde suas origens até as linguagens modernas, a partir de uma análise de como essas linguagens se desenvolveram ao longo do tempo e como foram influenciadas por avanços tecnológicos, mudanças na indústria de software e necessidades dos usuários. Além disso, abordar uma discussão sobre o uso atual das linguagens de programação e suas possíveis decorrências futuras. As linguagens de programação refletem uma evolução contínua e dinâmica, acompanhando as demandas e desafios da tecnologia e da computação. Este trabalho de conclusão de curso apresenta um panorama histórico das principais linguagens de programação, destacando suas características, paradigmas e aplicações. que este estudo contribua para o conhecimento e apreciação das linguagens de programação, bem como para a reflexão sobre seus desafios e oportunidades futuras.

2. FUNDAMENTAÇÃO TEÓRICA

A evolução das linguagens de programação é um tópico central na história da ciência da computação e da tecnologia da informação. Para compreender essa evolução, é fundamental mergulhar em sua fundamentação teórica, que engloba diversos aspectos, desde os princípios fundamentais das linguagens de programação até os fatores socioeconômicos que moldaram seu desenvolvimento ao longo do tempo. Esta fundamentação teórica ajuda a contextualizar a importância das linguagens de programação na resolução de problemas computacionais e na criação de sistemas complexos.

- 2.1. Teoria da Computação:** A evolução das linguagens de programação está intrinsecamente ligada à Teoria da Computação. Os princípios da computabilidade, tais como a máquina de Turing e a noção de algoritmo, fornecem as bases teóricas para a criação de linguagens que descrevem processos computacionais.
- 2.2. Abstração e Expressividade:** As linguagens de programação permitem a abstração de conceitos complexos em termos compreensíveis e a expressividade de ideias em código. A teoria das linguagens formais e gramáticas contextuais ajuda a compreender como as linguagens são definidas e interpretadas.
- 2.3. Compiladores e Interpretadores:** A teoria dos compiladores e interpretadores é essencial para entender como as linguagens de programação são traduzidas em código de máquina ou interpretadas por um ambiente de execução. Isso inclui a análise léxica e sintática, bem como a geração de código intermediário.
- 2.4. Paradigmas de Programação:** A evolução das linguagens de programação envolve a adoção de diferentes paradigmas, como programação procedural, orientada a objetos, funcional e lógica. Cada paradigma tem sua própria base teórica e influencia a forma como os programas são estruturados e concebidos.
- 2.5. Semântica e Tipagem:** A teoria da semântica formal e a teoria dos tipos desempenham um papel importante na definição do comportamento das linguagens e na prevenção de erros de programação. A segurança de tipos e a coerência semântica são conceitos centrais.
- 2.6. História e Contexto Socioeconômico:** Além dos aspectos técnicos, a evolução das linguagens de programação está inserida em um contexto socioeconômico. Fatores como a competição entre empresas de

tecnologia, a demanda por eficiência e a necessidade de resolver problemas complexos impulsionam a criação e a evolução das linguagens.

2.7. Padrões e Comunidades: As linguagens de programação são frequentemente definidas e mantidas por organizações de padronização, como a ISO e a IEEE. Compreender a influência dessas organizações na evolução das linguagens é essencial.

2.8. Ensino e Aprendizado: A teoria da educação em ciência da computação desempenha um papel importante na evolução das linguagens de programação, à medida que influencia a forma como as linguagens são ensinadas e aprendidas.

Em resumo, a fundamentação teórica por trás da evolução das linguagens de programação abrange uma ampla gama de tópicos, desde a teoria da computação até fatores sociais e econômicos. Compreender esses aspectos é fundamental para acompanhar e contribuir para o desenvolvimento contínuo das linguagens de programação e, assim, para a resolução de problemas computacionais cada vez mais complexos.

3. OBJETIVOS GERAIS E ESPECÍFICOS

O objetivo geral desta pesquisa é investigar e analisar a evolução das linguagens de programação ao longo do tempo, com o propósito de compreender não apenas o desenvolvimento técnico, mas também as influências conceituais e sociais que moldaram esse processo. A intenção é fornecer uma visão abrangente e contextualizada da trajetória das linguagens de programação, destacando seu papel crucial na evolução da computação e sua relevância contínua na resolução de desafios contemporâneos.

3.1. Objetivos Específicos:

- Identificar as principais linguagens de programação em diferentes épocas.

- Analisar as características técnicas e conceituais que influenciaram a evolução das linguagens.
- Compreender o papel das linguagens na evolução da computação e na solução de problemas contemporâneos.

4. METODOLOGIA

A condução desta pesquisa sobre a evolução das linguagens de programação envolverá uma abordagem multifacetada, empregando métodos qualitativos e quantitativos para proporcionar uma compreensão abrangente e aprofundada do tema. A metodologia escolhida visa realizar uma análise histórica e comparativa das linguagens de programação, destacando marcos importantes, inovações e mudanças de paradigmas ao longo do tempo.

Foi realizada uma revisão bibliográfica em bases de dados como Scopus, Web of Science e Google Scholar, utilizando palavras-chave como "linguagens de programação", "evolução", "história", "paradigmas" e "aplicações", além de sites diversos destinados à educação e pesquisa, assim analisando artigos, postagens, filmes e vídeos. A escolha dessa metodologia se justifica pela necessidade de compreender a evolução das linguagens de programação e seu impacto na indústria de software, permitindo uma análise aprofundada das mudanças e inovações ao longo do tempo.

5. ANÁLISE DA PESQUISA

O primeiro algoritmo foi criado por Ada Lovelace em 1843. Ela usou papel para escrever o primeiro algoritmo para um computador, pois não havia computadores eletrônicos na época. No entanto, a primeira linguagem de programação considerada de alto nível (linguagens com sintaxe mais simples para humanos) foi a Plankalkül, criada por Konrad Zuse entre 1942 e 1945. A Plankalkül foi desenvolvida para ser uma linguagem de programação de propósito geral, mas nunca foi implementada completamente. A primeira linguagem comercializada foi a FORTRAN, desenvolvida pela IBM em 1957. Desde então, muitas outras linguagens

de programação foram criadas, cada uma com suas próprias características e finalidades específicas.

A história das linguagens de programação remonta à criação dos primeiros computadores mecânicos. No início, as linguagens eram altamente especializadas e totalmente dependentes do hardware, as chamadas linguagem de máquina, mais próximas da linguagem entendida diretamente pelo computador, em código binário. Na década de 1940, nasceram os primeiros computadores eletrônicos modernos, mas a velocidade limitada e a pequena capacidade de memória forçaram os programadores a escrever programas em linguagem de máquina ou de montagem, o que exigia muito trabalho intelectual e era passível de erros. Nos anos 80, houve progressos significativos na implementação de linguagens de programação, com o movimento RISC na arquitetura de computadores defendendo que o hardware deveria ser projetado para os compiladores, e não para os programadores em linguagem assembly. A história das linguagens de programação é fascinante e inclui muitas linguagens criadas desde a década de 1940 até o presente, como FORTRAN, LISP, COBOL, ALGOL 60, C, Prolog, Pascal, C++, Java, JavaScript, PHP, Perl e Python.

Estudos como o realizado em 2018 implementaram problemas clássicos e realizaram análises comparativas, evidenciando a importância de compreender o desempenho das linguagens em diferentes contextos de aplicação. A evolução das linguagens de programação reflete a constante busca por suprir necessidades e apresentar novos conceitos que são inerentes a qualquer linguagem de alto nível. A diversidade de linguagens, paradigmas e propósitos demonstra a riqueza e complexidade desse campo, que continua a evoluir em resposta às demandas da indústria e às inovações tecnológicas.

As linguagens a seguir foram escolhidas com base em sua popularidade e relevância em cada década, atendendo às necessidades e demandas impostas pela computação em cada período.

- Década de 1980: C++ foi a linguagem mais popular, pois trouxe melhorias significativas em relação à linguagem C, tornando-se mais poderosa e flexível, sendo amplamente adotada para o desenvolvimento de sistemas e aplicações de grande porte.

- Década de 1990: Java foi a linguagem mais popular, pois atendeu à necessidade de uma linguagem portátil e capaz de criar aplicações robustas, especialmente para a web e ambientes corporativos.
- Década de 2000: PHP foi a linguagem mais popular, pois atendeu à demanda por uma linguagem amplamente utilizada para o desenvolvimento web, especialmente em sistemas de gerenciamento de conteúdo e aplicações web dinâmicas.
- Década de 2010: Python foi a linguagem mais popular, consolidando-se como uma das linguagens mais populares devido à sua legibilidade, vasta gama de bibliotecas e aplicabilidade em diversas áreas, incluindo ciência de dados e desenvolvimento web.
- Década de 2020: Python continua a ser a linguagem mais popular, devido à sua aplicabilidade em inteligência artificial, aprendizado de máquina, ciência de dados e desenvolvimento web.

C++ trouxe melhorias significativas em relação à linguagem C, Java atendeu à necessidade de uma linguagem portátil e capaz de criar aplicações robustas, PHP atendeu à demanda por uma linguagem amplamente utilizada para o desenvolvimento web, e Python consolidou-se como uma das linguagens mais populares devido à sua legibilidade, vasta gama de bibliotecas e aplicabilidade em diversas áreas. A partir dessas, foi realizada uma nova seleção visando a comparação entre as linguagens com o propósito de entender suas semelhanças e diferenças, percebendo o que as tornam tão singulares a ponto de se tornarem as mais populares em suas épocas e serem ampla e ativamente utilizadas até os dias de hoje. A seguir, serão apresentados quadros comparativos entre estas três linguagens: C++, Java e Python.

Tabela 1 — Tipos de variável

| Tipo de variável | C++ | JAVA | PYTHON |
|-------------------------|----------------------|----------------------|----------------------|
| <i>Inteiro</i> | <i>int</i> | <i>int</i> | <i>int</i> |
| <i>Decimal</i> | <i>float, double</i> | <i>float, double</i> | <i>float, double</i> |
| <i>Caractere</i> | <i>char</i> | <i>char</i> | <i>str</i> |
| <i>Texto</i> | <i>string</i> | <i>String</i> | <i>str</i> |
| <i>Booleano</i> | <i>bool</i> | <i>boolean</i> | <i>bool</i> |

Variáveis em programação são estruturas básicas que armazenam dados temporariamente durante a execução do programa. Conforme visto na Tabela 1, em C++, os tipos de variáveis são *int*, *float*, *double*, *char*, *string* e *bool*. Em Java, os tipos de variáveis são semelhantes: *int*, *float*, *double*, *char*, *String* e *Boolean*. Em Python, os tipos de variáveis são: *int*, *float*, *str* e *bool*. É importante notar que não há diferença entre um caractere e uma string em Python, pois ambos são representados por *str*.

Tabela 2 — Estruturas e sintaxe

| SINTAXE | C++ | JAVA | PYTHON |
|---------------------------------------|--|--|--|
| <i>Declaração de variáveis</i> | <i>int x = 5;</i> | <i>int x = 5;</i> | <i>x = 5</i> |
| <i>Estrutura de repetição</i> | <i>for(int i=0; i<10; i++) { }</i> | <i>for(int i=0; i<10; i++) { }</i> | <i>for i in range(10):</i> |
| <i>Estrutura condicional</i> | <i>if (x > 5) { } else { }</i> | <i>if (x > 5) { } else { }</i> | <i>if x > 5: else:</i> |
| <i>Funções</i> | <i>int sum(int a, int b) { return a + b; }</i> | <i>int sum(int a, int b) { return a + b; }</i> | <i>def sum(a, b): return a + b</i> |

Em geral, C++, Java e Python têm diferenças significativas em termos de sintaxe, como a declaração de variáveis, estruturas de repetição, estruturas condicionais e definição de funções. É notável na Tabela 2 uma maior semelhança entre Java e C++, enquanto a linguagem Python se destaca por ser mais curta e intuitiva.

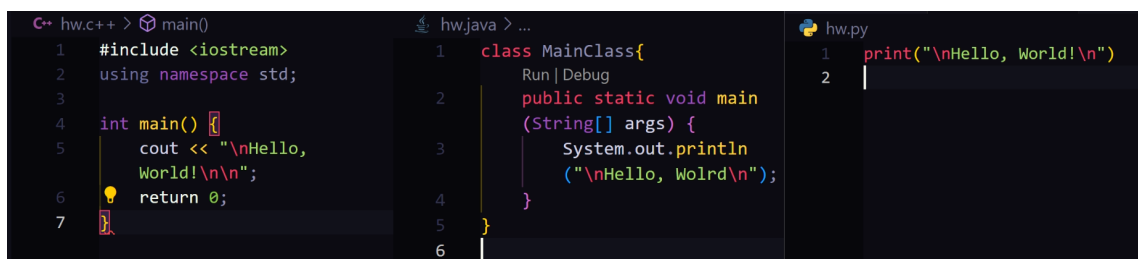
Tabela 3 — Vantagens e desvantagens

| Linguagem | Vantagens | Desvantagens |
|---------------|--|---|
| C++ | <i>Alta performance, ampla utilização em sistemas operacionais e jogos</i> | <i>Complexidade, dificuldade de aprendizado</i> |
| Java | <i>Portabilidade, segurança, facilidade de aprendizado</i> | <i>Performance inferior a C++, limitações em programação de sistemas operacionais</i> |
| Python | <i>Fácil aprendizado, grande quantidade de bibliotecas, alta produtividade</i> | <i>Performance inferior a C++ e Java em aplicações de alta demanda</i> |

Em geral, C++ é conhecida por sua alta performance, mas é mais complexa e difícil de aprender. Java é portátil e segura, mas tem performance inferior a C++ e limitações na programação de sistemas operacionais. Python é fácil de aprender, tem uma grande quantidade de bibliotecas e alta produtividade, mas tem performance inferior a C++ e Java em aplicações de alta demanda.

Como ilustração final de uma análise comparativa entre essas três linguagens de programação, as imagens a seguir mostram um exemplo simples e conhecido de código que quase todo programador faz ao iniciar seus estudos em algoritmo e logica de programacao, “Hello, World!”, notando-se mais uma vez a diferença entre as sintaxes e estruturas de cada uma:

Imagem 1 — Código “Hello, World!” em C++, Java e Python, respectivamente



The image displays three code snippets side-by-side, each for a different programming language. The first snippet is for C++ (hw.cpp), showing the use of `<iostream>`, `using namespace std;`, and `cout` to print the message. The second snippet is for Java (hw.java), showing a `MainClass` with a `main` method that uses `System.out.println`. The third snippet is for Python (hw.py), showing a single line using `print` to output the message. Each snippet is numbered 1 and 2, corresponding to the line numbers in the code.

6. CONSIDERAÇÕES FINAIS

Ao final deste estudo sobre o desenvolvimento de linguagens de programação, pode-se ressaltar que o caminho percorrido mostra muito mais do que apenas uma linha cronológica de avanços tecnológicos. Uma análise detalhada das línguas ao

longo do tempo proporcionou uma compreensão profunda das complexas interações de fatores técnicos, conceituais e sociais que moldaram esta paisagem dinâmica.

O estudo das propriedades técnicas e conceituais das linguagens enfatizou a busca constante pela eficiência, expressividade e adaptabilidade. Das linguagens pioneiras às linguagens mais recentes, o progresso é visível não só na complexidade das instruções, mas também na flexibilidade oferecida aos desenvolvedores. A diversidade de paradigmas de programação reflete a busca constante por abordagens mais eficientes para expressar a lógica computacional.

Compreender o papel das linguagens no desenvolvimento da computação revela uma história contínua de adaptação às novas necessidades. As línguas não só facilitaram a comunicação entre humanos e máquinas, mas também influenciaram profundamente a forma como pensamos e resolvemos problemas computacionais. A importância contínua destas línguas na resolução dos desafios atuais reforça a sua natureza fundamental na era digital. O exame das implicações sociais e éticas do desenvolvimento de linguagens de programação destaca a crescente necessidade de considerar não apenas os aspectos técnicos, mas também o impacto humano. As questões de inclusão, acessibilidade e igualdade tornam-se cada vez mais urgentes e exigem uma reflexão ética constante à medida que as línguas evoluem.

REFERÊNCIAS

<https://www.atrainformatica.com.br/2023/04/05/historia-das-linguagens-de-programacao/>

<https://www.dio.me/articles/evolucao-das-linguagens-de-programacao-um-panorama-historico-ate-os-dias-atuais>

https://www.academia.edu/15520004/A_Hist%C3%B3ria_das_Linguagens_de_Programa%C3%A7%C3%A3o

<https://repositorio.ufsc.br/bitstream/handle/123456789/197462/TCC%20Gabriel%20Jurask%20%281%29.pdf?sequence=1>

<https://revistas.unifacs.br/index.php/rsc/article/download/5133/3488>

<http://www.ime.unicamp.br/~apmat/ada-lovelace/#:~:text=As%20notas%20de%20Lovelace%20foram,de%20primeira%20programadora%20da%20hist%C3%B3ria.>

<https://blog.eseg.edu.br/linguagens-de-programacao/>