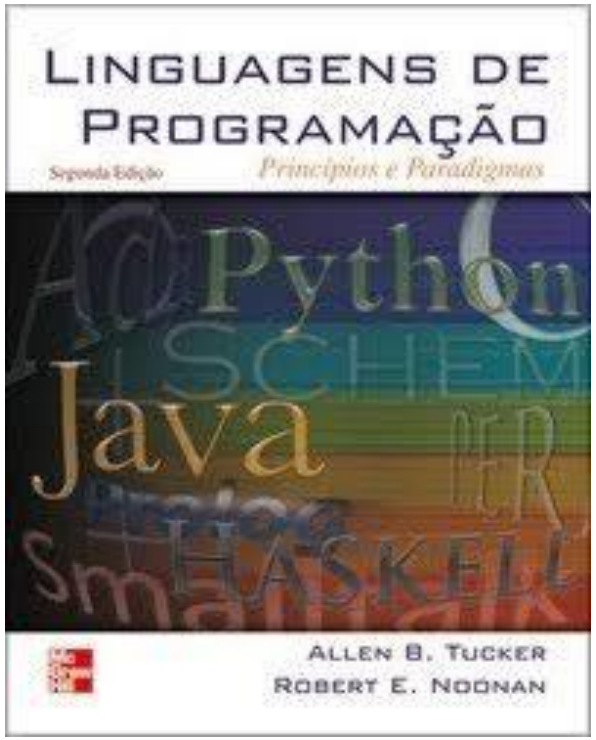
Paradigmas de Linguagem de Programação

Fabio Lubacheski
fabio.lubacheski@mackenzie.br
http://lattes.cnpq.br/9894811024725114

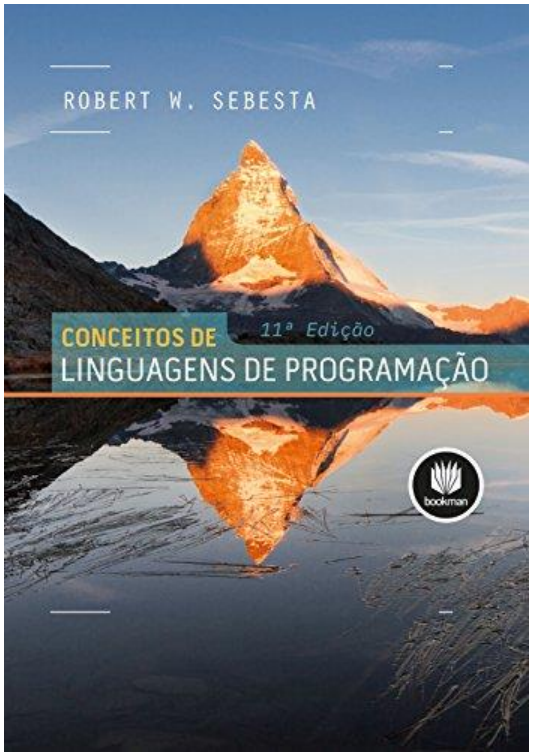
Plano de Ensino

- Conteúdo Programático
- Metodologia
- Critérios de Avaliação
- Bibliografia básica e complementar

Bibliografia



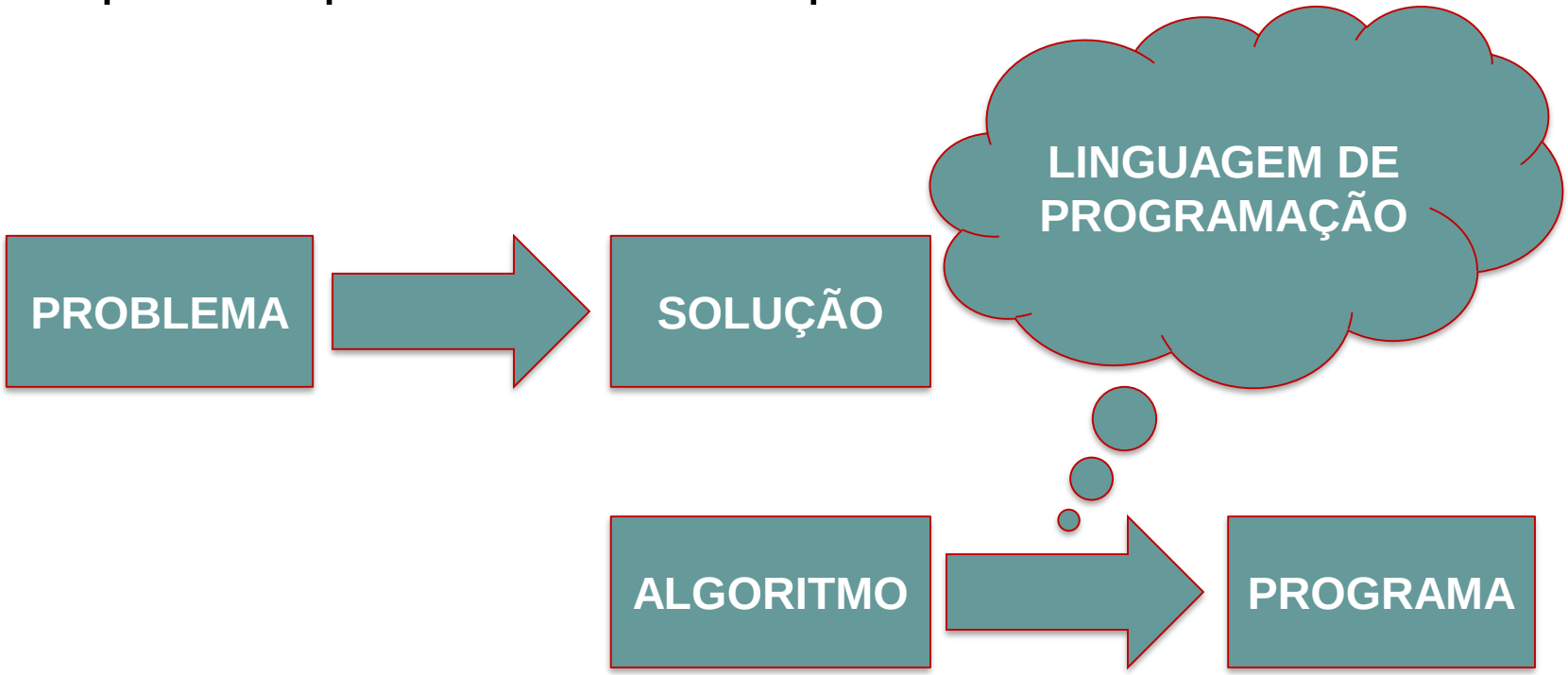
TUCKER, A. B.; NOONAN, R. E. **Linguagens de programação: Princípios e Paradigmas**.



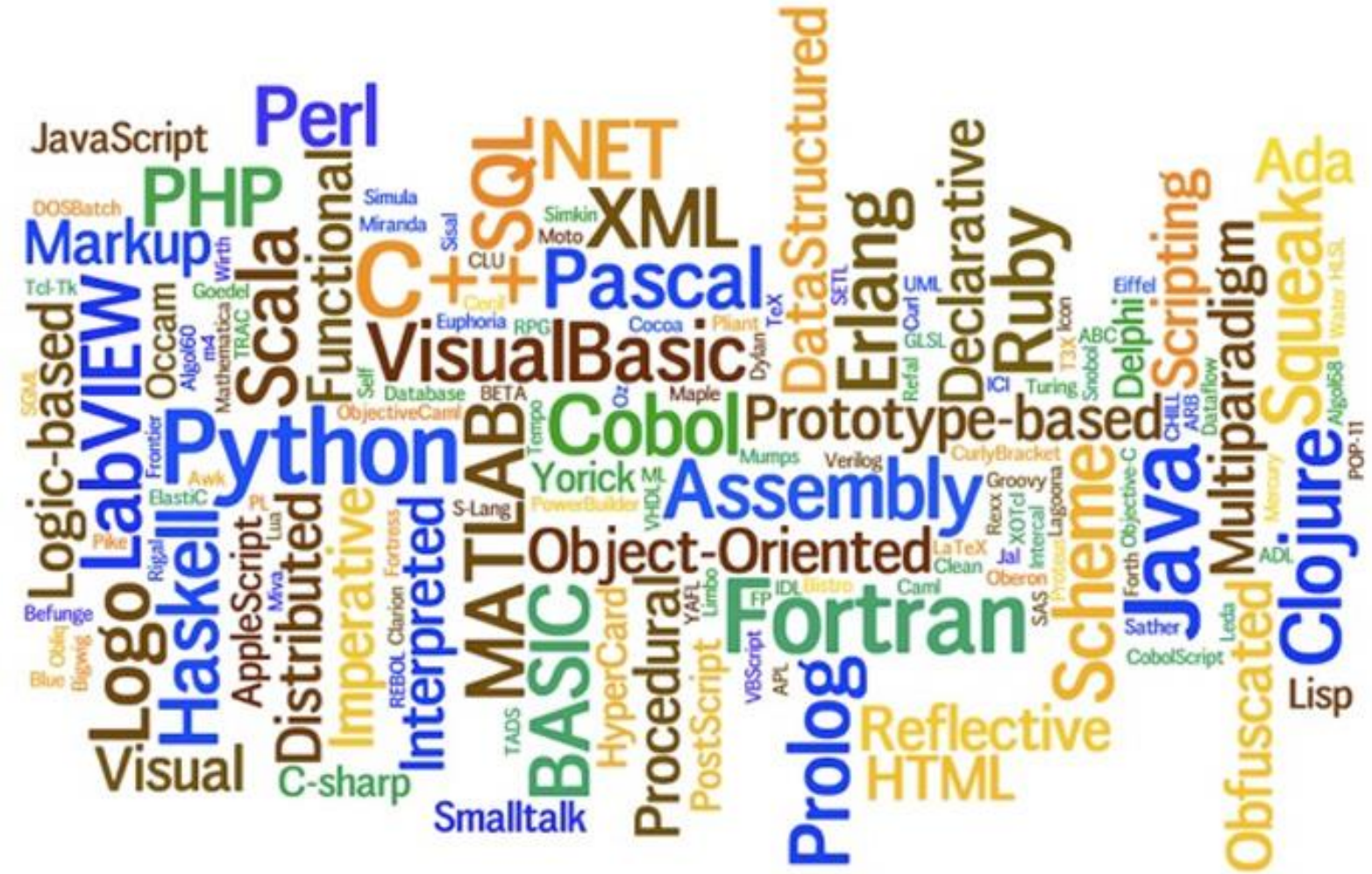
SEBESTA, R.W. **Conceitos de Linguagens de programação.**

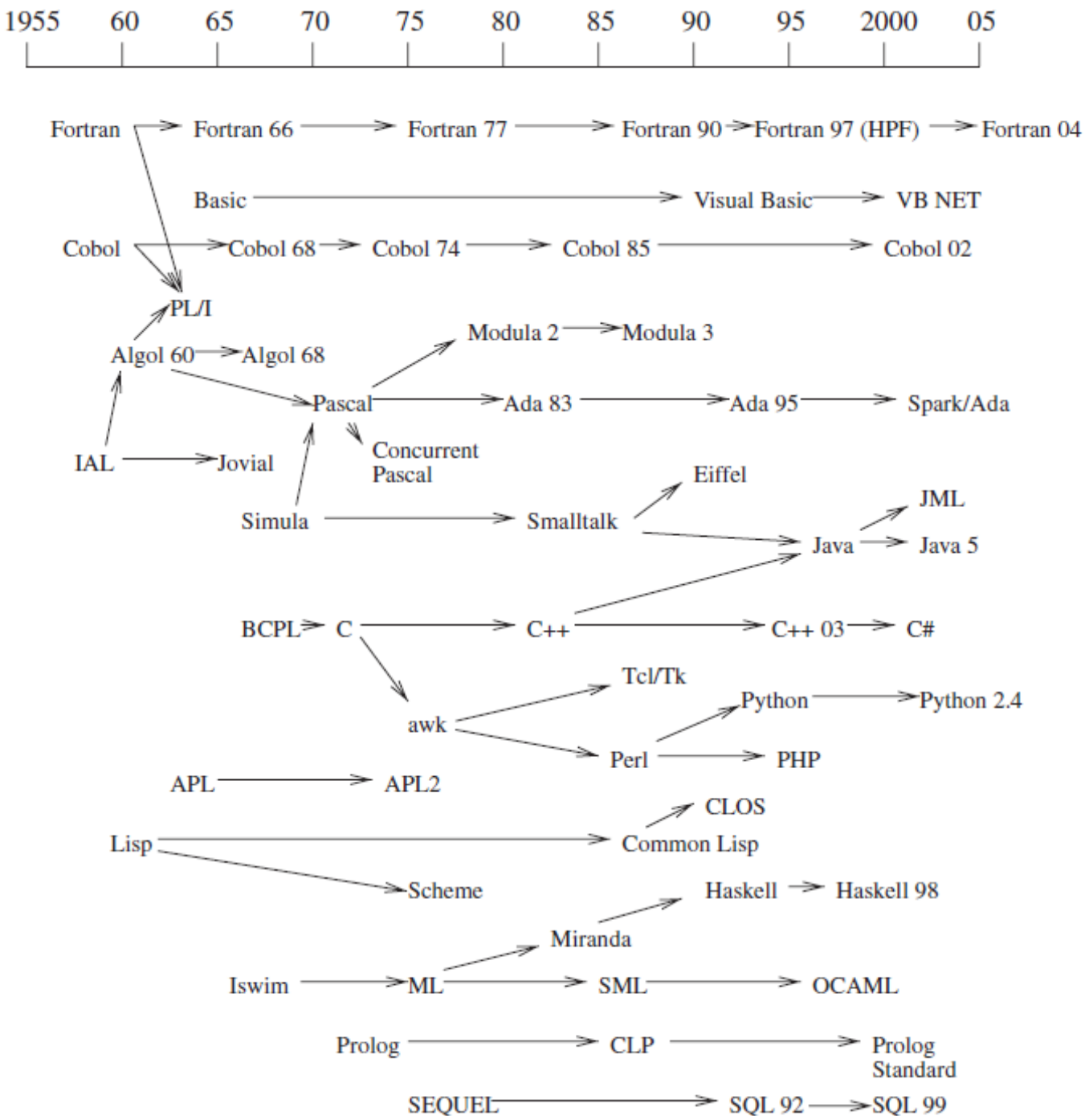
O que é uma linguagem de programação ?

- Uma **linguagem de programação** é usada por uma pessoa para expressar um processo através do qual um computador pode resolver um problema.



Quais linguagens de programação existe ?




1955
60
65
70
75
80
85
90
95
2000
05
Fortran
Fortran 66
Fortran 77
Fortran 90
Fortran 97 (HPF)
Fortran 04
Basic
Visual Basic
VB NET
Cobol
Cobol 68
Cobol 74
Cobol 85
Cobol 02
PL/I
Algol 60
Algol 68
Modula 2
Modula 3
Pascal
Ada 83
Ada 95
Spark/Ada
IAL
Jovial
Concurrent
Pascal
Simula
Smalltalk
Eiffel
JML
Java
Java 5
BCPL
C
C++
C++ 03
C#
Tcl/Tk
awk
Python
Python 2.4
Perl
PHP
APL
APL2
CLOS
Lisp
Common Lisp
Scheme
Haskell
Haskell 98
Miranda
Iswim
ML
SML
OCAML
Prolog
CLP
Prolog
Standard
SEQUEL
SQL 92
SQL 99

Ranking das linguagens de programação

https://www.tiobe.com/tiobe-index/

Jan 2020	Jan 2019	Change	Programming Language	Ratings	Change
1	1		Java	16.896%	-0.01%
2	2		C	15.773%	+2.44%
3	3		Python	9.704%	+1.41%
4	4		C++	5.574%	-2.58%
5	7	↑	C#	5.349%	+2.07%
6	5	↓	Visual Basic .NET	5.287%	-1.17%
7	6	↓	JavaScript	2.451%	-0.85%
8	8		PHP	2.405%	-0.28%
9	15	⇈	Swift	1.795%	+0.61%
10	9	↓	SQL	1.504%	-0.77%
11	18	⇈	Ruby	1.063%	-0.03%
12	17	⇈	Delphi/Object Pascal	0.997%	-0.10%
13	10	↓	Objective-C	0.929%	-0.85%
14	16	↑	Go	0.900%	-0.22%
15	14	↓	Assembly language	0.877%	-0.32%
16	20	⇈	Visual Basic	0.831%	-0.20%
17	25	⇈	D	0.825%	+0.25%
18	12	⇊	R	0.808%	-0.52%
19	13	⇊	Perl	0.746%	-0.48%
20	11	⇊	MATLAB	0.737%	-0.76%

Um pouco de história...

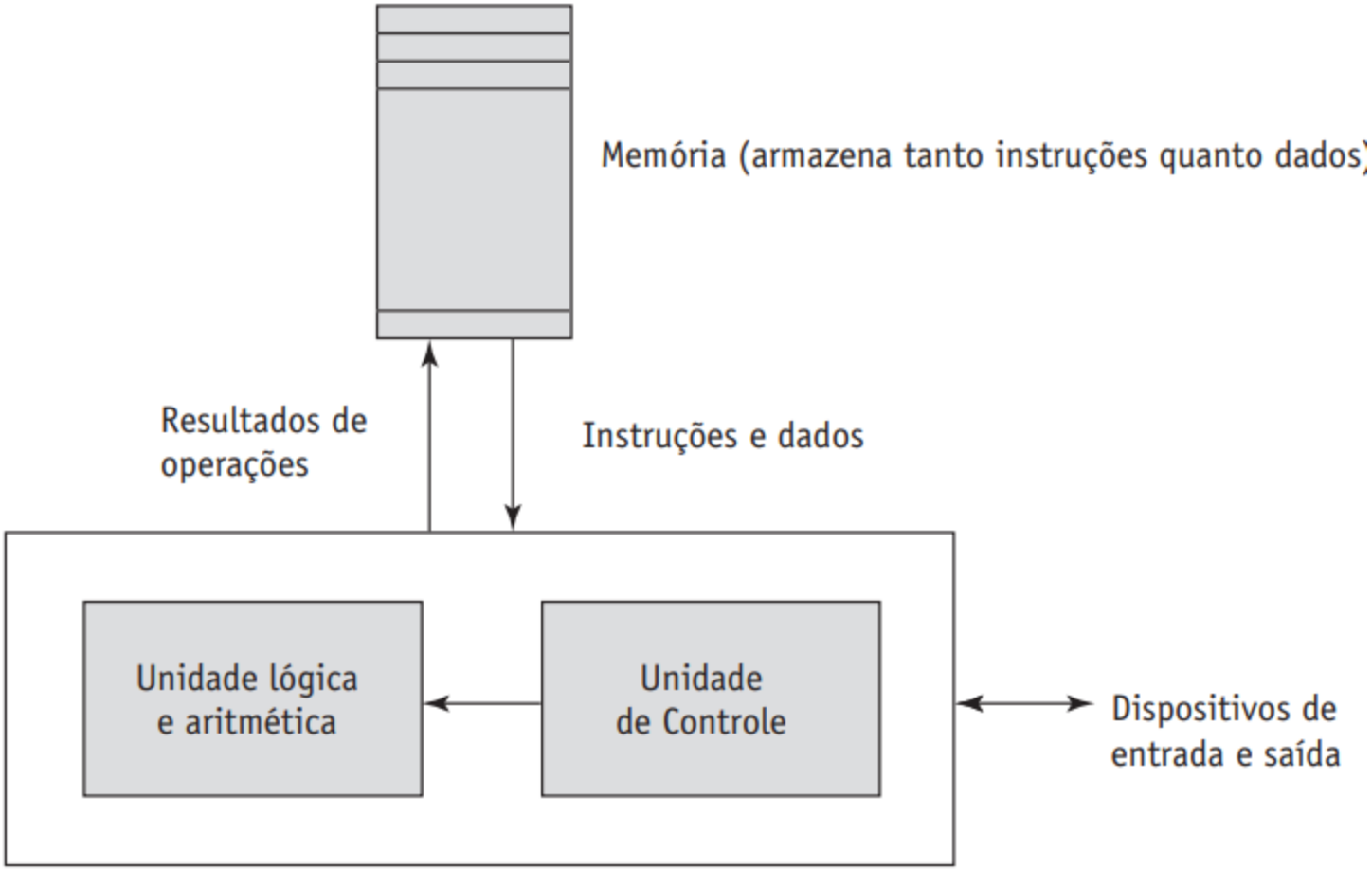
- O primeiro paradigma de programação (**imperativo**) foi fortemente influenciado pela arquitetura de Von-Neumann, onde tanto o programa como os dados residem na memória do computador (Máquina de Turing).

- Nessa arquitetura a memória da máquina contém tanto instruções de programas (***armazenamento de programa***) quanto valores de dados (***armazenamento de dados***).

- No coração dessa arquitetura está a ***atribuição*** – alterar o valor de um **local de memória** e **destruir seu valor anterior**.

Um pouco de história...

A arquitetura de computadores de von Neumann



Um pouco de história...

- O **Fortran** foi primeira linguagem de programação imperativa, desenvolvida para o IBM 704 em 1954.
- O compilador era optimizado, pois os autores acreditavam que ninguém iria usar essa linguagem se a sua rapidez não fosse comparável com a da **linguagem assembly**.

Um pouco de história...

- Os programas escritos nas linguagens imperativas suportam declarações de variáveis, expressões, comandos ramificação **condicionais** e **incondicionais** (**goto**), laços e **abstração procedural** (utilização de subprogramas ou funções).

- No início do desenvolvimento programas utilizando as linguagens imperativas, o **comando incondicional goto** era muito utilizado.

- Alguns pesquisadores diziam que o código usando goto ficavam ilegível, outros afirmavam que uso de goto era viável em alguns casos.

Um pouco de história...

- Em 1968 **Dijkstra** publicou um artigo "**Go To Statement Considered Harmful**“ na ACM **contra o uso excessivo de comandos de goto** era prejudicial ao processo de desenvolvimento de programas confiáveis.

  https://pt.wikipedia.org/wiki/Edsger_Dijkstra

  https://pt.wikipedia.org/wiki/Considered_harmful

- **Donald Knuth advogou a favor do goto** em algumas circunstâncias, mesmo depois do estabelecimento da programação estruturada.

  https://pt.wikipedia.org/wiki/Donald_Knuth

  https://dl.acm.org/citation.cfm?id=356635.356640

Um pouco de história...

- Nos anos 70 (1970) o **software** começa se tornar mais caro que **hardware**, além disso os problemas que serão resolvidos pelos comutadores se tornam maiores e mais complexos.

- Novas metodologias de desenvolvimento de software começam a surgir, como desenvolvimento (*top-down*) e o refinamento passo a passo.

- Os projetos de programas começam a utilizar a abstração com o propósito é simplificar o processo de programação.

- A ideia da **abstração** é focar nos aspectos essenciais do problema ignorando características menos importantes.

Conceito de abstração

- Os dois tipos de abstração nas linguagens de programação são a **abstração procedural** e **abstração de dados**.

- Um programa pode ser decomposto em diversos subprogramas, cada subprograma pode ser considerado uma **abstração procedural**, e para utilizar um subprograma (função) o programador não precisa saber detalhes de como o subprograma funciona, basta conhecer a declaração do subprograma (*interface*) .

- Por exemplo, em programa, caso o programador necessite ordenar um vetor, o programar considera somente a declaração abaixo, sem se preocupar como é feita a ordenação ou qual método é implementado.
  sortInt(list, listLen)

Conceito de abstração

- A **abstração de dados** é uma especificação de um conjunto de dados e operações que podem ser executadas sobre esses dado.

- Uma das motivações para a abstração de dados é similar à motivação para a abstração procedural: é uma arma contra a complexidade; uma forma de tornar programas grandes e/ou complicados mais gerenciáveis.

Continuando a história ...

- No início dos anos 1980 surgem os **projetos orientado a objetos** usando metodologias que utilizam a **abstração de dados** que encapsulam e controlam o acesso aos dados de um objeto e também adiciona mecanismos de herança e vinculação dinâmica de dados.

- A **herança** é um conceito poderoso que melhora o potencial reuso de software existente, fornecendo a possibilidade de melhorias significativas na produtividade no contexto de desenvolvimento de software.

- A **vinculação dinâmica** de métodos (em tempo de execução) permite um uso mais flexível da herança.

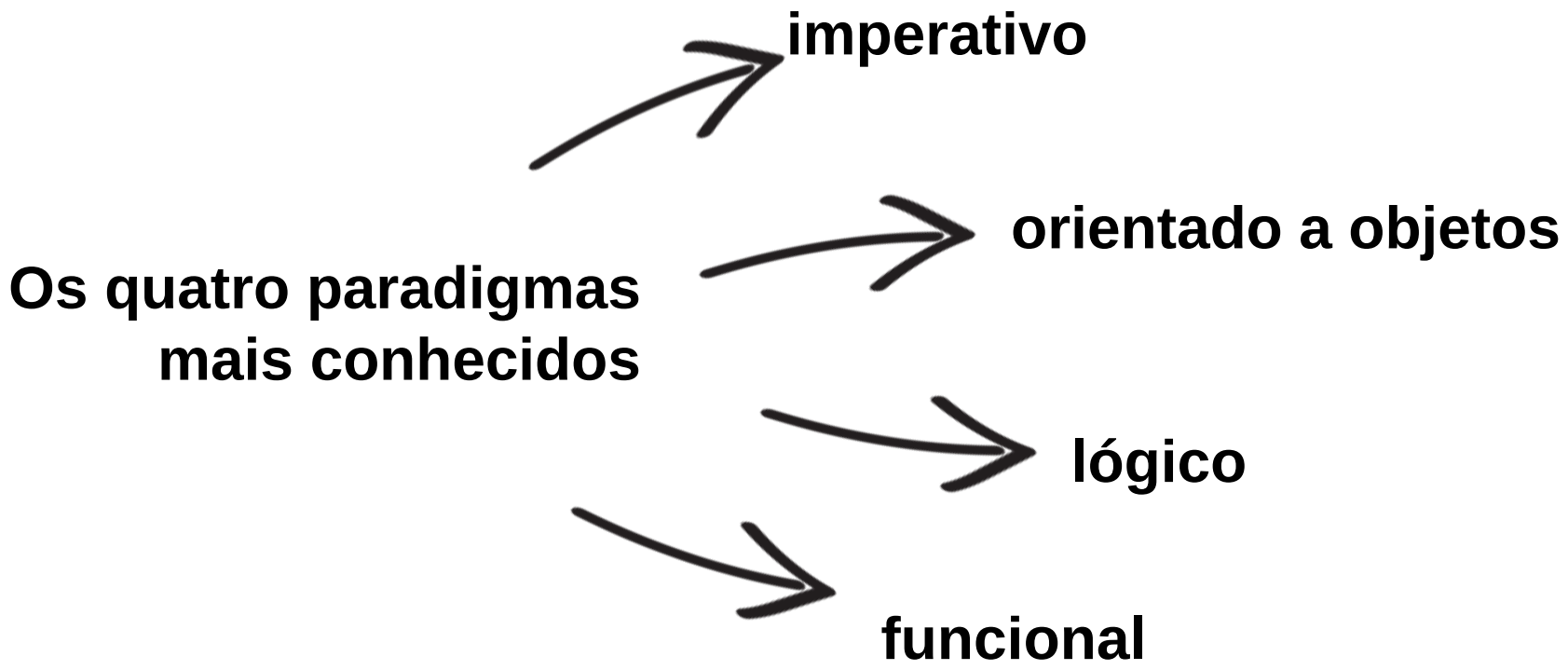
Paradigmas de programação

- Todos esses passos evolutivos nas metodologias de desenvolvimento de software levaram a novas construções de linguagens para suportar as metodologias propostas.

- Linguagens de programação são normalmente divididas em quatro categorias (paradigmas): **imperativas**, **orientadas a objetos**, **lógicas** e **funcionais**.

- Um **paradigma de programação** é um padrão de resolução de problemas que se relaciona a **determinado gênero de programas e linguagens.**

Paradigmas de programação



Paradigma imperativo

- É o paradigma mais antigo (1940), fundamentado no **modelo computacional de Von-Neumann**. Nesse paradigma um programa contém uma série de comandos para executar cálculos, atribuir valores a variáveis, obter entrada e produzir saídas ou redirecionar o **controle para outro ponto** nessa série de comandos.
- **Desestímulo** ao uso de desvio incondicional (**goto**)
- A abstração procedural (funções) é um componente utilizada nesse paradigma, assim como os laços, estrutura condicionais.
- Exemplos de linguagens: Cobal, Fortran, C, Ada e Pascal

  https://pt.wikipedia.org/wiki/Programação_imperativa

Paradigma orientado a objetos – POO

- A POO foi uma evolução do paradigma imperativo e fornece um modelo no qual um programa é uma coleção de objetos que interagem entre si, passando mensagens que transformam seu estado. Essa característica ajuda distinguir melhor a POO da programação imperativa.
- A **herança, polimorfismo** e a **passagem de mensagens** são componentes fundamentais da programação POO.
- Baseia no conceito de classe= atributos + métodos e na **abstração de dados** (TAD).
- Exemplos de linguagens: Smaltalk, C++, Java e C#

  https://pt.wikipedia.org/wiki/Orientação_a_objetos

Paradigma lógico

- A programação lógica (declarativa) surgiu em 1970 e é diferente dos outros paradigmas porque ela requer que o programador declare os objetivos da computação, em vez de algoritmos detalhados.

- As declarações do programação são baseadas em regras ou restrições sobre o problema, em vez de uma sequência de comandos a serem executados.

- Exemplos de linguagens: Prolog (Inteligência artificial) e Structured Query Language - SQL (banco de dados).

  https://pt.wikipedia.org/wiki/Programação_lógica

Paradigma funcional

- A criação da programação funcional (1960) foi motivada pela necessidade dos pesquisadores no desenvolvimento de inteligência artificial, computação simbólica provas de teoremas e processamento de linguagem natural.
- A programação funcional modela um problema como uma coleção de funções matemáticas, isso separa a programação funcional das linguagens que possuem o comando de atribuição (x = x + 1), isso não faz sentido.
- Uma linguagem funcional ***pura*** não possui variáveis.
- Exemplos de linguagens: Lisp, Scheme e Haskell.

  https://pt.wikipedia.org/wiki/Programação_funcional

Tarefa: Leiam os capítulos dos livros abaixo.

- Leia o capítulo 1 do livro TUCKER, A. B.; NOONAN, R. E. **Linguagens de programação: Princípios e Paradigmas**.

- Leia os capítulo 1 seções 1.3 e 1.4 e capítulo 11 seções 11.1 e 11.2 do livro SEBESTA, R.W. **Conceitos de Linguagens de programação.**

Fim