Revisão: linguagens de programação

Definição, histórico, paradigmas
Sintaxe e semântica

Prof. Thiago A. S. Pardo
taspardo@icmc.usp.br

Definição

- Uma linguagem de programação é uma linguagem destinada a ser usada por uma pessoa para expressar um processo através do qual um computador pode resolver um problema
- Dependendo da perspectiva, têm-se
  - Pessoa = paradigma lógico
  - Processo = paradigma funcional
  - Computador = paradigma imperativo
  - Problema = paradigma orientado a objetos

2

Paradigma lógico

- Perspectiva da pessoa
- Um programa lógico é equivalente à descrição do problema expressa de maneira formal, similar à maneira que o ser humano raciocinaria sobre ele
- Exemplo de linguagem: PROLOG

3

Paradigma funcional

- Perspectiva do processo
- A visão funcional resulta num programa que descreve as operações que devem ser efetuadas (processos) para resolver o problema
- Exemplo de linguagem: LISP

4

Paradigma imperativo

- Perspectiva do computador
- Baseado na execução seqüencial de comandos e na manipulação de estruturas de dados
- Exemplos de linguagens: FORTRAN, COBOL, ALGOL 60, APL, BASIC, PL/I, SIMULA 67, ALGOL 68, PASCAL, C, MODULA 2, ADA

5

Paradigma orientado a objetos

- Perspectiva do problema
- Modelagem das entidades envolvidas como objetos que se comunicam e sofrem operações
- Exemplos de linguagens: SIMULA, SMALLTALK
  - C++: linguagem híbrida (paradigmas imperativo e orientado a objetos)

6

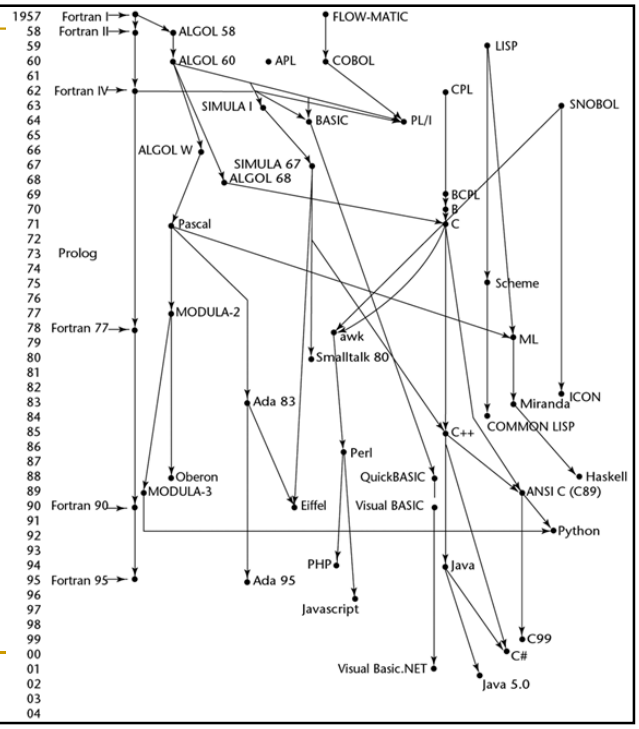
Um pouco de história

- Linguagens que introduziram conceitos importantes e que ainda estão em uso
  - 1955-1965: FORTRAN, COBOL, ALGOL 60, LISP, APL, BASIC
  - 1965-1971 (com base em ALGOL): PL/I, SIMULA 67, ALGOL 68, PASCAL
  - Anos 70 e 80: PROLOG, SMALL TALK, C, MODULA 2, ADA

7

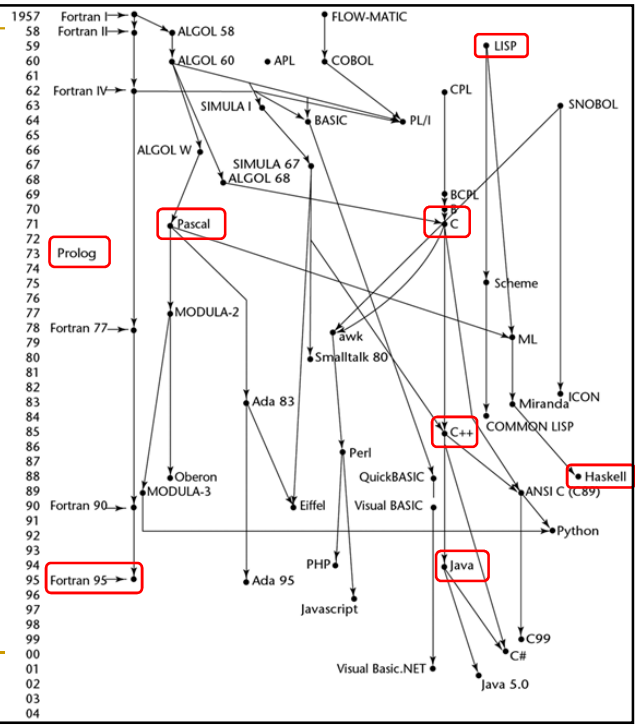
Um pouco de história

- Sebesta, 2009



Um pouco de história

- Sebesta, 2009



Um pouco de história

- Domínios de programação
  - Aplicações científicas: FORTRAN
    - Grandes números, cálculos com ponto flutuante, arranjos
  - Comércio, negócios: COBOL
    - Relatórios, números decimais e caracteres
  - IA: LISP
    - Símbolos, listas
  - Sistemas: C
    - Eficiência e velocidade
  - Web: XHTML, PHP, Java
    - Marcação/estilo, velocidade

10

Sintaxe e semântica

- A descrição de uma linguagem de programação envolve dois aspectos principais
  - Sintaxe: conjunto de regras que determinam quais construções são corretas
  - Semântica: descrição de como as construções da linguagem devem ser interpretadas e executadas
- Em Pascal: a:=b
  - Sintaxe: comando de atribuição correto
  - Semântica: substituir o valor de a pelo valor de b

11

Sintaxe

- A sintaxe de uma linguagem é descrita por uma gramática com os seguintes elementos
  - Símbolos terminais: cadeias que estão no programa
    - while, do, for, id
  - Símbolos não-terminais: não aparecem no programa
    - <cmd_while>, <programa>
  - Produções: como produzir cadeias que formam o programa
    - <cmd_while> ::= while (<expressão>) <comandos>
  - Símbolo inicial: não-terminal a partir do qual se inicia a produção do programa
    - <programa>

12

Sintaxe

- BNF (*Backus Naur Form*): uso dos símbolos <> e ::=
- Exemplo de gramática

<cálculo> ::= <expressão> = <expressão>
<expressão> ::= <valor> | <valor><operador><expressão>
<valor> ::= <número> | <sinal><número>
<número> ::= <dígito> | <dígito><número>
<operador> ::= + | - | * | /
<sinal> ::= + | -
<dígito> ::= 0 |1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

13

Sintaxe

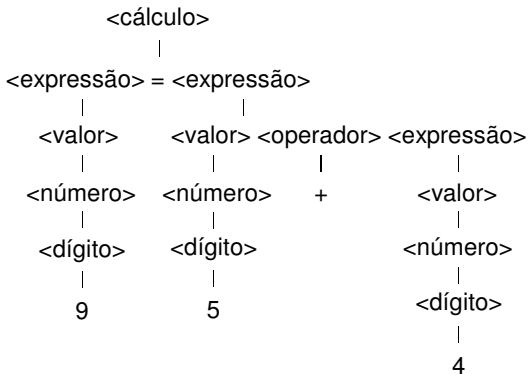
- EBNF (*Extended BNF*)
  - Opcionalidade [ ] e repetição { }
- Re-escrita da gramática anterior

<cálculo> ::= <expressão> = <expressão>
<expressão> ::= <valor> [<operador><expressão>]
<valor> ::= [<sinal>] <número>
<número> ::= <dígito> {<dígito>}
<operador> ::= + | - | * | /
<sinal> ::= + | -
<dígito> ::= 0 |1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

14

Sintaxe

- A gramática pode ser usada para produzir ou reconhecer programas sintaticamente corretos
  - 9=5+4



15

Sintaxe

- A gramática pode ser usada para produzir ou reconhecer programas sintaticamente corretos
  - 9=5+4

9 = 5 + 4

<dígito> = <dígito> + <dígito>

<número> = <número> + <número>

<valor> = <valor> + <valor>

<valor> = <valor> <operador> <expressão>

<expressão> = <expressão>

<cálculo>

16

Sintaxe

- A gramática pode ser usada para produzir ou reconhecer programas sintaticamente corretos
  - 9=5+

9 = 5 +

<dígito> = <dígito> <operador>

<número> = <número> <operador>

<valor> = <valor> <operador>

<valor> = <valor> <operador>

<expressão> = <expressão> <operador>

<cálculo> <operador>

CADEIA INVÁLIDA

17

Sintaxe

- As gramáticas de linguagens de programação são utilizadas para produzir ou reconhecer cadeias?

18

Sintaxe

- Descrição de linguagens de programação por meio de gramáticas livres de contexto
- A maioria das linguagens não são livres de contexto, mas sensíveis ao contexto
  - Por exemplo, variável deve ser declarada antes de ser usada
- Métodos para reconhecer gramáticas sensíveis ao contexto são complexos. Na prática, especifica-se uma gramática livre de contexto para a linguagem de programação e trata-se a sensibilidade ao contexto de maneira informal
  - Tabela de símbolos

19

Gramáticas e reconhecedores

Gramáticas	Reconhecedores
Irrestrita	Máquina de Turing
Sensível ao contexto	Máquina de Turing com memória limitada
Livre de contexto	Autômato a pilha
Regular	Autômato finito

20