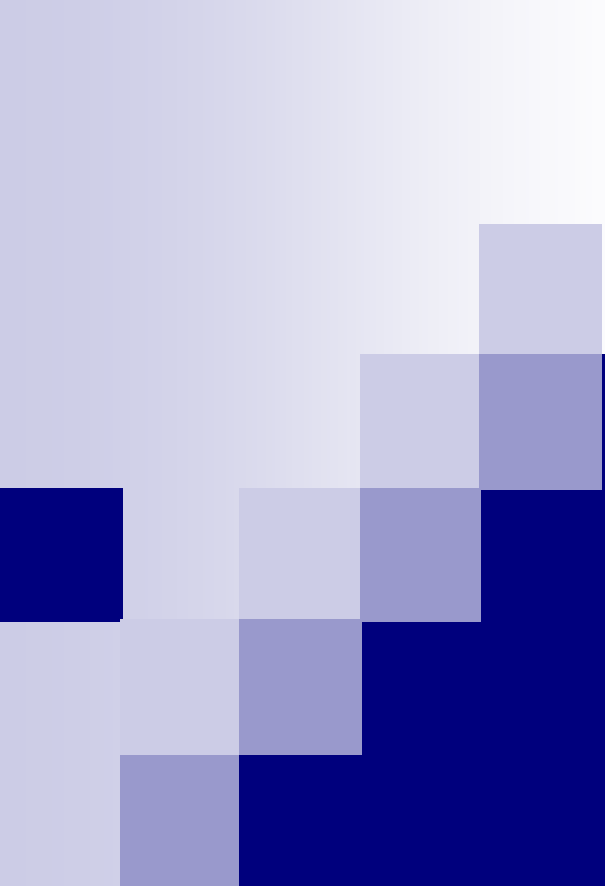




Introdução Paradigmas de Linguagens de Programação

Prof. Maurício Braga



Linguagens de programação

- Linguagem de programação: meio de comunicação entre o usuário e o computador, através da qual problemas podem ser resolvidos.
- Elemento fundamental no processo de construção de software;



Linguagens de programação e engenharia de software

- A escolha da linguagem afeta diretamente diferentes aspectos do software, tais como:
 - Performance;
 - Tamanho do código / software;
 - Facilidade (dificuldade) de programação;
 - Produtividade dos programadores;



Cenário atual

- Software cada vez mais complexo;
- Usuários cada vez mais exigentes em relação aos atributos de qualidade do software (Ex: usabilidade, confiabilidade, desempenho etc);
- Diversidade de ambientes (web, mobile, desktop, linux, windows, mac os, etc.) nos quais o software deve estar disponível / funcionar.
- Cenário atual requer o conhecimento sobre quais paradigmas de programação e quais LPs são mais adequados para um determinado software com necessidades particulares.



Linguagem de Programação

- Uma linguagem de programação (LP) pode ser definida como uma notação formal que pode ser usada para representar o conhecimento bem como definir problemas / algoritmos que possam ser executados em um computador.



Linguagens de baixo nível

- São aquelas nas quais o programador manipula diretamente o hardware, e escreve o programa usando as instruções reconhecidas / suportadas pela CPU do computador.
- As linguagens de montagem (assembly) e as linguagens de máquina são exemplos de linguagens de baixo nível.
- Foram muito utilizadas no início da computação, e embora tenham caído em desuso, ainda são utilizadas em situações bem específicas ou em sistemas legados.



Linguagens de baixo nível

- Apesar da velocidade de execução, otimização do tamanho do código, a programação em assembly é difícil, o código produzido não é estruturado, não é portátil e muito sujeito a bugs.
- Com o passar do tempo, linguagens de alto nível foram criadas, para facilitar o trabalho de programação;



Linguagens de alto nível

- São linguagens que são mais fáceis de entender / trabalhar, sendo mais próximas do ser humano.
- Costumam ser independentes da arquitetura do computador, o que facilita a sua migração entre diferentes plataformas;
- Ao utilizá-las, o programador pode se concentrar no problema sendo resolvido, deixando de lado aspectos sobre o funcionamento do hardware onde o programa será executado.

Linguagens de Programação

- Como a CPU só consegue executar instruções em linguagem de máquina, um processo de tradução se torna necessário para permitir a criação de softwares utilizando uma linguagem de alto nível;

PASCAL	JAVA
<pre>program Hello; var mensagem : string; begin mensagem := 'Hello World!'; write(mensagem); End.</pre>	<pre>public class Main { public Main(){ System.out.println("Hello World"); } public static void main(String [] args){ Main m =new Main(); } }</pre>
C	COBOL
<pre>#include <stdio.h> #include <stdlib.h> int main() { printf("HELLO WORLD!!!"); return(0); }</pre>	<pre>function Hello (){ alert("Hello World!") }</pre>



Endereço	Código	Assembly
1B8D:0100	01D8	ADD AX,BX
1B8D:0102	C3	RET
1B8D:0103	16	PUSH SS
1B8D:0104	B03A	MOV AL,3A
1B8D:0106	380685D5	CMP [D585],AL
1B8D:010A	750E	JNZ 011A
1B8D:010C	804E0402	OR BYTE PTR [BP+04],02
1B8D:0110	BF86D5	MOV DI,D586
1B8D:0113	C6460000	MOV BYTE PTR [BP+00],00
1B8D:0117	E85F0B	CALL 0C79
1B8D:011A	8B7E34	MOV DI,[BP+34]
1B8D:011D	007C1B	ADD [SI+1B],BH



Conversão linguagem alto nível → linguagem máquina

- Conversão é realizada através de compiladores que recebem o programa escrito na linguagem de alto nível (código fonte) e produzem como saída uma representação do mesmo programa expresso em código objeto.
- O programa compilado pode então ser executado diretamente na máquina.
- Outra opção para executar programas de alto nível é através do uso de um interpretador (linguagens interpretadas).



Linguagens de alto nível

■ Vantagens:

- ☐ Maior portabilidade;
- ☐ Mais fácil de serem usadas que as linguagens de baixo nível;

■ Desvantagens:

- ☐ Ocupam mais memória;
- ☐ Performance inferior a programas equivalentes escritos diretamente em Assembly.



Componentes de uma linguagem

- **Léxico;**
- **Sintaxe;**
- **Semântica;**

Léxico

- Conjunto de palavras reservadas que compõe a linguagem;
- Palavras reservadas: são de uso da linguagem, não podem ser definidas como variáveis ou ter seu significado redefinido pelo programador;
- Ex: na linguagem Java são exemplos de palavras reservadas: **if, while, for, else, public static**



Sintaxe

- Conjunto de regras que definem a forma de uma linguagem (o formato como as sentenças podem ser escritas).
- A partir dessas regras, pode-se identificar se uma sentença está correta ou não.
- A sintaxe permite validar o formato das instruções que compõem um programa.



Sintaxe

- Caso o programador viole as regras de sintaxe da linguagem será emitido uma mensagem informando a existência de um erro de sintaxe no programa.
- Todos os erros de sintaxe precisam ser corrigidos para que o programa possa ser compilado.
- Por tratar apenas de regras de construção das sentenças, um programa pode estar sintaticamente correto e ainda assim estar semanticamente errado, contendo bugs ou não realizando aquilo que foi planejado (erros de lógica).



Sintaxe

- A sintaxe diz como as palavras devem ser usadas para construir comandos e programas corretamente.
- Palavras são formadas entretanto por caracteres que pertencem a um alfabeto.
- Assim, a sintaxe de uma linguagem é definida por dois conjuntos de regras: regras léxicas e regras sintáticas.

Regras léxicas

- Especificam o conjunto de caracteres que constituem o alfabeto da linguagem e a maneira como os caracteres podem ser combinados para formar palavras, sentenças, comandos e expressões válidas.
- Exemplo regra léxica:
 - a regra para formação de nomes de variáveis em uma linguagem pode especificar que o identificador de uma variável deve iniciar por uma letra, um cifrão ou um underline, e após a primeira letra pode ter conter letras e numeros apenas (caracteres especiais são vedados), tendo até um determinado número de caracteres.

Exemplos de regras léxicas

$\langle \text{digit} \rangle ::= 0 \mid 1 \mid \dots \mid 9$

$\langle \text{letter} \rangle ::= a \mid b \mid c \mid \dots \mid z$

$\langle \text{identifier} \rangle ::= \langle \text{letter} \rangle \mid \langle \text{identifier} \rangle \langle \text{letter} \rangle \mid \langle \text{identifier} \rangle \langle \text{digit} \rangle$

$\langle \text{operator} \rangle ::= + \mid - \mid * \mid /$

$\langle \text{relation} \rangle ::= \leq \mid \geq \mid == \mid > \mid < \mid !=$



Regras sintáticas

- Especificam as seqüências de símbolos que constituem estruturas sintáticas válidas;
- São verificadas pelo compilador através de uma análise (parsing) do código fonte.
- Nas IDEs modernas, isso é feito tanto durante a escrita do código como durante a compilação do programa.



Exemplos de regras sintáticas

`<type> ::= boolean | integer | real | string`

`<declaration> ::= var <variable list> : <type>;`

`if <boolean expr> then <command seq> end`

`while <boolean expr> <boolean expr> do <command seq> end`

Exemplo regras sintáticas

- Declaração variáveis:

- ☐ Linguagem Pascal: `var i: integer;`
- ☐ Linguagem Java / C e outras: `int i;`

- Atribuição entre variáveis:

- ☐ Linguagem Pascal: `a := b`
- ☐ Linguagem Java / C e outras: `a = b`



Gramática

- A sintaxe de uma linguagem é definida através de uma gramática (conjunto de regras que definem todos os elementos que podem ser aceitos na linguagem).
- A gramática BNF fornece uma definição compacta e clara para a sintaxe de linguagens em geral, e foi usada na definição de linguagens clássicas tais como Pascal e C.
- BNF é uma “metalinguagem”, pois consiste numa linguagem para descrição de outras linguagens, mais especificamente de gramáticas usadas por outras linguagens.

Semântica

- Define o significado dos programas sintaticamente corretos e tem a ver com a interpretação e o significado correto das instruções e estruturas de um programa.
- Exemplo: semântica da declaração **int vetor[10]** na linguagem C:
 - São reservados dez espaços para valores inteiros, acessados a partir de uma variável chamada vetor.
 - Os elementos do vetor podem ser referenciados por um índice i , $0 \leq i \leq 9$, sendo o primeiro deles `vetor[0]`.

Semântica

- Exemplo: semântica da declaração **if A > 10 THEN C = D** na linguagem Basic:
 - A expressão $a > 10$ deverá ser avaliada, e se for verdadeira, a variável C receberá uma cópia do conteúdo armazenado na variável D.
- Portanto, as regras sintáticas mostram como formar o comando e as regras semânticas informam qual é o efeito que o comando produz.



Paradigma

- Paradigma: termo que tem origem no grego paradeigma que significa modelo ou padrão, correspondendo a algo que vai servir de modelo ou exemplo a ser seguido em determinada situação.
- Pode ser pensado como um conjunto de regras que formam domínios e permitem a resolução de problemas dentro desses domínios;
- Permite representar e manipular o conhecimento;



Paradigmas de programação

- Meio de se classificar as linguagens de programação baseado em suas funcionalidades e no formato que os programas são escritos.
- Assim como um software pode ser construído usando diferentes metodologias (processos) de desenvolvimento, linguagens de programação podem se basear em diferentes paradigmas de programação.



Paradigmas de programação

- Podem impor restrições na organização do código, no agrupamento de código em unidades, afetando também o estilo de sintaxe ou a gramática da linguagem.
- Podem ser diferenciados também pelas técnicas de programação que proíbem ou permitem.
 - Ex: a programação estruturada não permite o uso de goto



Paradigmas de programação

- Imperativo;
- Orientado a objetos;
- Funcional;
- Lógico;



Paradigma imperativo

- Também chamado de paradigma procedural, por incluir subrotinas ou procedimentos como mecanismo de estruturação.
- Foi o primeiro paradigma a surgir e ainda é o dominante.



Paradigma imperativo

- Baseado na idéia de um estado que sofre modificações durante a computação.
- O elemento central de um programa imperativo é o comando, que pode ser primitivo (como atribuição, leitura e escrita) ou composto, como seqüência, condicional e iteração.
- Faz uso também de procedimentos e funções como elemento estruturador, que podem ser parametrizados, e possuem seu próprio corpo de comandos.



Paradigma imperativo

- Paradigma de programação baseado no conceito de chamadas de funções e procedimentos, e as instruções são estruturadas em sub-rotinas.
- Neste paradigma temos uma lista de instruções para dizer ao computador o que fazer passo a passo.
- As linguagens de programação procedurais compõe a maioria das linguagens criadas e são conhecidas como linguagens Top-down.



Paradigma funcional

- Paradigma de programação que trata a computação como uma avaliação de funções matemáticas e que evita estados ou dados mutáveis.
- Enfatiza a aplicação de funções, em contraste da programação imperativa, que enfatiza mudanças no estado do programa.
- Exemplos de linguagens funcionais: Scala, Lisp, Haskell, R, Ocaml.



Paradigma lógico

- Se fundamenta na lógica matemática, na qual as declarações do programa expressam fatos e regras sobre problemas dentro de um sistema.
- Em vez de um algoritmo, os elementos constituintes são os fatos e as regras.
- Seguem uma abordagem declarativa em vez de imperativa (baseada em um algoritmo).

Paradigma lógico

- Prolog: linguagem baseada no paradigma lógico.
- Prolog possui dois aspectos principais, um para expressar os dados e outro para consultá-los. Existem três elementos básicos:
 - Fatos → afirmações fundamentais sobre o domínio do problema (por exemplo, "Sócrates é um homem")
 - Regras → inferências sobre fatos no domínio (por exemplo, "Todos os homens são mortais.")
 - Consultas → questões sobre esse domínio (por exemplo, "É Sócrates mortal?")



Paradigma lógico

- O paradigma lógico tem sido muito usado nos domínios da prova de teoremas, sistemas especialistas, processamento de linguagem natural e no campo da inteligência artificial em geral.



Paradigma orientado a objetos

- Neste paradigma, os dados e as rotinas para manipulá-los são mantidos numa unidade chamada objeto.
- Objetos são criados de classes, e essas devem ser modeladas de acordo com os princípios:
 - Abstração;
 - Encapsulamento;
 - Herança;
 - Polimorfismo;
 - Alta coesão;
 - Baixo acoplamento.

Paradigmas de programação

- Fornece e determina a visão que o programador possui sobre a estruturação e execução do programa. Ex:
 - Na programação imperativa os programadores definem variáveis e algoritmos que manipulam essas variáveis.
 - Na programação orientada a objetos, os programadores podem abstrair um programa como uma coleção de objetos que interagem entre si;
 - Na programação funcional os programadores abstraem o programa como uma sequência de funções executadas de modo empilhado.
 - Na programação lógica, os programadores definem fatos e regras em vez de algoritmos.



Paradigmas de programação

- Paradigmas e exemplos de linguagens que os seguem:
 - Paradigma imperativo (ex: Basic, Pascal, C, Ada, Assembly...)
 - Paradigma OO → (Ex: Smalltalk, Java, C# ...)
 - Paradigma funcional: (ex: Haskell, Lisp, ML, Miranda).
 - Paradigma lógico: (ex: Prolog)

Paradigmas de programação

- Algumas linguagens suportam múltiplos paradigmas. Ex:
 - C++ → Paradigma imperativo + paradigma orientado a objetos;
 - Ocaml → Paradigma funcional + paradigma orientado a objetos.



Linguagens de Programação

- Desde os anos 50 até hoje, muitas LPs foram projetadas, para atender diferentes propósitos, fazendo uso de diferentes paradigmas de programação.
- Cada tipo de problema pode ser resolvido de forma mais fácil se utilizado a LP / paradigma correto.
- Veremos algumas dessas linguagens agora.

Exemplos de linguagens de programação

■ Linguagem de máquina

- Linguagem nativa de uma CPU.
- Cada CPU tem sua própria versão.
- Código enxuto e eficiente, porém sujeito a erros e difícil de programar.
- Código escrito fica preso a plataforma para o qual foi desenvolvido.
- Linguagem de baixo nível, suas instruções manipulam diretamente o hardware da máquina.



Exemplos de linguagens de programação

- Assembly (linguagem de montagem)
 - Equivalente da linguagem de máquina em uma forma mais fácil de utilizar, substituiu as instruções por mnemônicas, endereços de memória absolutos por identificadores e permitiam o uso de macros.
 - A linguagem de montagem ainda é considerada de baixo nível, ainda que seja um paradigma da "segunda geração" das linguagens.
 - Ainda usada em sistemas críticos ou embarcados.

Exemplo assembly

STROUT EQU 9

BDOS EQU 5

start:

LD DE,FRASE

LD C,STROUT

CALL BDOS

JP 0

FRASE: DB 'Testando...\$'

end

**Compilação desse programa
produz:**

**00 00 00 11 0E 01 0E 09 CD 05
00 C3 00 00 54 65 73 74 ...**

Memory

0100	00	00	00	11	0E	01	0E	09	
0108	CD	05	00	C3	00	00	54	65	Te
0110	73	74	61	6E	64	6F	2E	2E	stando..
0118	2E	24	67	43	09	B8	C8	6F	..gC...o

Exemplos de linguagens de programação

■ Fortran

- Formula Translator → Criada em 1954, projetada para aplicações matemáticas, envolvendo fórmulas e cálculos complexos.

■ Lisp

- Criada em 1958, *List Processor* foi projetada primariamente para o processamento de dados simbólicos, tendo sido muito utilizada na área inteligência artificial.

Exemplos de linguagens de programação

■ Cobol

- *Common Business Oriented Language*. → criada em 1959, projetada para aplicações comerciais e administrativas, orientada a resolver problemas de negócios.

■ Basic

- *Beginner's All-purpose Symbolic Instruction Code*, criada com fins didáticos em 1964 no Dartmouth College.

Exemplo Cobol

```
PROCEDURE DIVISION.  
*  
000-CALCULATE-SALES-TAX.  
*  
    PERFORM 100-CALCULATE-ONE-INTEREST  
        UNTIL END-OF-SESSION-SWITCH = "Y".  
    DISPLAY "END OF SESSION."  
    STOP RUN.  
*  
100-CALCULATE-ONE-INTEREST.  
*  
    DISPLAY "-----".  
    DISPLAY "TO END PROGRAM, ENTER 0."  
    DISPLAY "ENTER THE INVESTMENT AMOUNT."  
    ACCEPT PRESENT-VALUE.  
    IF PRESENT-VALUE = ZERO  
        MOVE "Y" TO END-OF-SESSION-SWITCH  
    ELSE  
        PERFORM 200-SHOW-FUTURE-VALUE  
    END-IF.
```

“The use of Cobol cripples the mind; its teaching should, therefore, be regarded as a criminal offense.” **(Edsger W. Dijkstra)**

Exemplo Basic

```
MSX BASIC versão 1.0 Br
G r a d i e n t e
23430 bytes livres
Basic de Disco V1.0
Ok
load"pares.bas
Ok
list
10 PRINT"Programa imprime os números pa
res"
20 PRINT"Digite o valor de A e B"
30 INPUT A,B
40 FOR I=A TO B STEP 1
50 IF I MOD 2 > 0 THEN 70
60 PRINT I
70 NEXT I
80 END
Ok
■
```

Exemplos de linguagens de programação

■ C / C++

- Criada em 1972 por Dennis Ritchie na empresa AT&T Bell Labs para desenvolvimento do sistema operacional Unix (que originalmente foi escrito em Assembly).
- Uma das linguagens de programação mais populares da computação, tendo influenciado muitas outras linguagens de programação (por exemplo, a linguagem Java).

Exemplos de linguagens de programação

■ Java

- a linguagem de programação orientada a objetos desenvolvida na década de 90 por uma equipe de programadores chefiada por James Gosling, na empresa Sun Microsystems.
- Diferente de outras linguagens de programação, que são compiladas para código nativo, a linguagem Java é compilada para um bytecode que é interpretado por uma máquina virtual (JVM).

Algumas propriedades desejáveis em uma linguagem

■ Legibilidade do código:

- ☐ Permite a criação de funções ?
- ☐ Permite variáveis locais a funções / blocos ?
- ☐ Não requer o uso de linhas de código numeradas ?
- ☐ Não requer o uso de comandos de desvio incondicional (Ex: GOTO) ?
- ☐ Permite a construção de código estruturado ?
- ☐ Não é verbosa ?



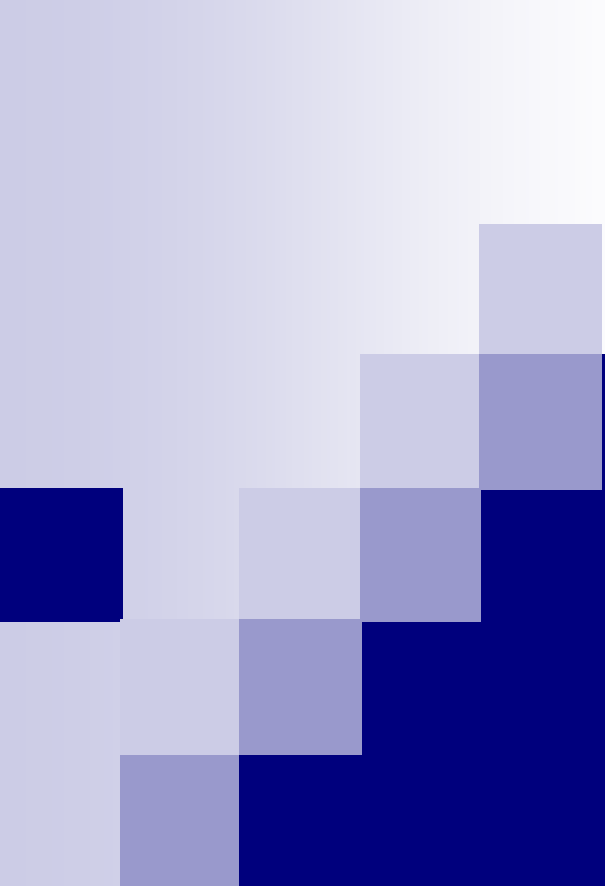
Algumas propriedades desejáveis em uma linguagem

- Fácil de aprender;
- Código simples;
- Eficiente;
- Confiável (produz a mesma saída para o mesmo conjunto de entradas);



Algumas propriedades desejáveis em uma linguagem

- Possibilidade de criação de tipos de dados estruturados por parte do usuário;
- Neutra em relação a arquitetura:
- Suporte a tratamento de erros (exceções);
- Multithread;
- Segurança;
- Portabilidade.



Introdução Paradigmas de Linguagens de Programação

Prof. Maurício Braga