

Trabalho de Estruturas de Linguagens

Linguagem Lisp: História, Classificação e Expressividade

Alunos:
Kleber Luiz
Luiz Henrique



História do Lisp

- Uma linguagem de programação chamada Lisp (LISt Processor, do inglês) foi desenvolvido para o computador IBM 704, por um grupo de Inteligência Artificial do MIT e John McCarthy, participante desse grupo, elaborou um artigo sobre o LISP.
- A linguagem Lisp nasceu como uma ferramenta matemática, independente de qualquer computador e só posteriormente se procedeu à sua adaptação a uma máquina.
- A linguagem LISP é interpretada, onde o usuário digita expressões em uma linguagem formal definida e recebe de volta a avaliação de sua expressão. Deste ponto de vista podemos pensar no LISP como uma calculadora, que ao invés de avaliar expressões aritméticas avalia expressões simbólicas, chamadas de expressões.



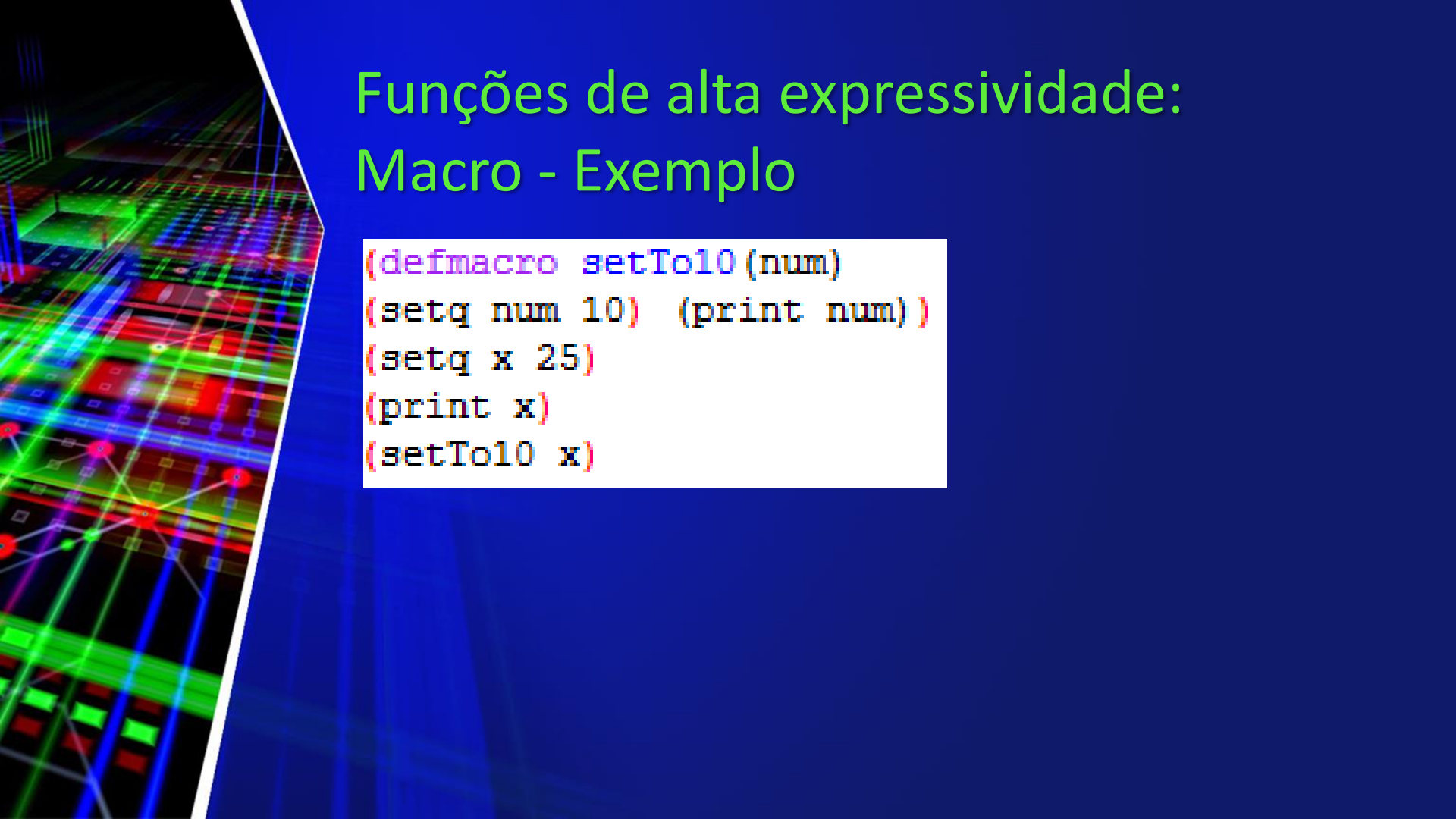
Classificação do Lisp

- Common Lisp é uma linguagem dinâmica, cujos programas são constituídos por pequenos módulos, de funcionalidade genérica e que cumprem um objetivo muito simples. É a sua combinação que produz um programa completo.
- A não tipificação de dados, a possibilidade de tratar dados e programas de um mesmo modo e a indistinção entre funções definidas pela linguagem e funções definidas pelo programador são algumas das razões da sua flexibilidade.




Funções de alta expressividade: Macro

- Common Lisp é uma linguagem dinâmica, cujos programas são constituídos por pequenos módulos, de funcionalidade genérica e que cumprem um objetivo muito simples. É a sua combinação que produz um programa completo.
- A não tipificação de dados, a possibilidade de tratar dados e programas de um mesmo modo e a indistinção entre funções definidas pela linguagem e funções definidas pelo programador são algumas das razões da sua flexibilidade.




Funções de alta expressividade: Macro - Exemplo

```
(defmacro setTo10 (num)
  (setq num 10) (print num))
(setq x 25)
(print x)
(setTo10 x)
```



Funções de alta expressividade: Orientação a Objeto

Como mencionado, Lisp é uma linguagem que permite o paradigma da OO. Vamos usar, como exemplo, o mesmo programa escrito em duas linguagens diferentes: Lisp e Java. Será notado, após os exemplos, o quão expressivo a linguagem Lisp é, sem deixar a legibilidade de lado.



Funções de alta expressividade: Orientação a Objeto - Comparação

- Em Lisp:

```
(defclass cubo()  
  ((comprimento :accessor cubo-comprimento)  
   (largura :accessor cubo-largura)  
   (altura :accessor cubo-altura)  
  ))  
  
(setf item (make-instance 'cubo))  
(setf (cubo-comprimento item) 10)  
(setf (cubo-largura item) 10)  
(setf (cubo-altura item) 5)  
  
(format t "Comprimento do cubo: ~d~%" (cubo-comprimento item))  
(format t "Largura do cubo: ~d~%" (cubo-largura item))  
(format t "Altura do cubo: ~d~%" (cubo-altura item))
```


Funções de alta expressividade: Orientação a Objeto - Comparação

- Em Java:

```
1 public class Cubo{
2     double comprimento;
3     double altura;
4     double largura;
5     public Cubo(double comprimento, double altura, double largura){
6         this.comprimento = comprimento;
7         this.altura = altura;
8         this.largura = largura;
9     }
10
11     public String getAltura(){
12         return this.altura;
13     }
14     public double getComprimento(){
15         return this.comprimento;
16     }
17     public double getLargura(){
18         return this.largura;
19     }
20     public void setAltura(double EntradaAltura){
21         altura = EntradaAltura;
22     }
23     public void setComprimento(double EntradaComprimento){
24         comprimento = EntradaComprimento;
25     }
26
27     public void setLargura(double EntradaLargura){
28         largura = EntradaLargura;
29     }
30
31     public static void main(String[] args){
32         Cubo cubo1 = new Cubo();
33         cubo1.setAltura(10);
34         cubo1.setComprimento(10);
35         cubo1.setLargura(5);
36         System.out.println(m.getAltura());
37         System.out.println(m.getComprimento());
38         System.out.println(m.getLargura());
39     }
40 }
```




Tipagem

- O Common Lisp tem tipagem forte, pois não permite um mesmo dado ser tratado como se fosse de outro tipo e dinâmica devido a verificação ser feita em cima do dado em si, já que as variáveis podem conter qualquer tipo de dado.
- Claro que em determinado momento uma variável só pode conter um tipo de dado e isto é verificado. Mas a principal diferença é que a esta verificação é feita em tempo de execução. Isto é feito através de uma infraestrutura auxiliar (uma máquina virtual ou uma biblioteca normalmente chamada de runtime).



That's all Folks!