PICAT: uma Linguagem Multiparadigma

Claudio Cesar de Sá, Rogério Eduardo da Silva, João Herique Faes Battisti, Paulo Victor de Aguiar

joaobattisti@gmail.com

pavaguiar@gmail.com

claudio.sa@udesc.br

Departamento de Ciência da Computação
Centro de Ciências e Tecnológias
Universidade do Estado de Santa Catarina

Sumário

Introdução

Características

Tipos de Dados

Exemplos
    Entradas e Saídas

Conclusão

Histórico

- Criada em 2013 por Neng-Fa Zhou e Jonathan Fruhman.
- Utilizou o B-Prolog como base de implementação, e ambas utilizam a programação em lógica baseada na Lógica de Primeira-Ordem (LPO)
- Uma evolução ao Prolog após seus mais de 40 anos de sucesso!
- Sua atual versão é a 2.0 (31 de janeiro de 2017)

Picat é Multiparadigma:

- Imperativo – Procedural
- Funcional
- Lógico

Algumas Características:

- Sintaxe $\Rightarrow$ elegância do código
- Velocidade de execução
- Portabilidade (todas plataformas)
- Extensão há outras ferramentas

Algumas Características:

- Terminologia: segue as bases teóricas da linguagem Prolog.
- Na **l**ógica de **p**rimeira-**o**rdem (LPO) os objetos são chamados por **termos**.
- O destaque de Picat é a sua natureza declarativa, funcional, tipagem dinâmica, e sintaxe *açucarada*
- `PICAT` é um anacrônico onde cada letra representa uma característica de sua funcionalidade (operacionalidade).

Anacrônico: **P.I.C.A.T.**

P: *Pattern-matching*: Utiliza o conceito de casamento padrão da LPO

I: *Intuitive*: oferece atribuições e laços de repetições análogo as outras linguagens de programação

C: *Constraints*: suporta a programação por restrições

A: *Actors*: suporte as chamadas a eventos, os atores (futuro gráfico)

T: *Tabling*: implementa a técnica de *memoization*, com soluções imediatas para problemas de Programação Dinâmica.

Instalação do PICAT

Tenham um editor de código de programa.
Sugestão: *geany* ou *sublime*

Sistema de Programação

- Picat é uma linguagem de multiplataforma, disponível em qualquer arquitetura de processamento e também de sistema operacional
- Utiliza a extensão .pi em seus arquivos de código fonte.
- Existem 2 modos de utilização do Picat: Modo linha de comando e Modo Interativo.

Tipos de Dados

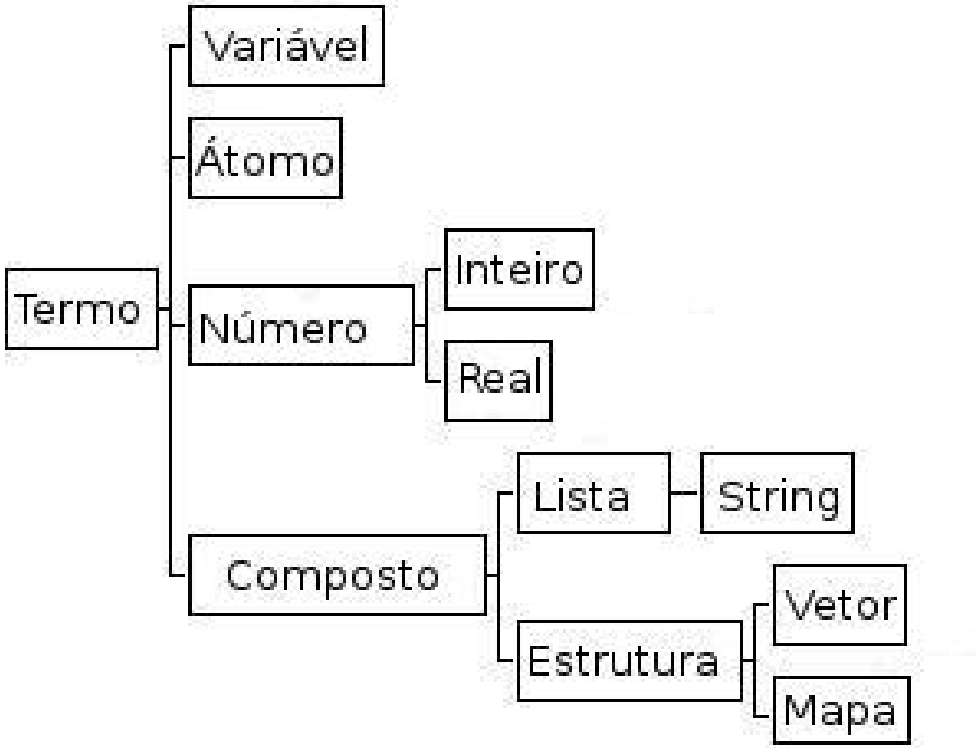

Figura: Hierarquia dos Tipos de dados

Número

```
Picat> A = 5, B = 7, number(A), number(B), max(A, B) =
Maximo, min(A, B) = Minimo.
A = 5
B = 7
Maximo = 7
Minimo = 5
yes.
```

Exemplos

Atribuição

```
Picat> X := 7, X := X + 7, X := X + 7.
X = 21
```

Estruturas de Controle

```
ex1 =>
X:=3, Y:=4,
if(X >= Y)
then printf("%d", X)
else printf("%d", Y)
end.
```

Entradas e Saídas

```
main =>
printf("Digite dois números:  "),
N_real01 = read_real(),
N_real02 = read_real(),
Media = (N_real01 + N_real02)/2,
printf("A média é:  %6.2f", Media),
printf("\n.....FIM.....\ n").
```

Regras em Lógica – os pais!

- $pai(platao, luna)$
- $pai(platao, pricles)$
- $pai(epimenides, platao)$

leia-se: *Platão é o pai de Luna*

leia-se: *Platão é o pai de Péricles*

leia-se: *Sócrates é o pai de Platão*

Regras em Lógica – os pais!

- $pai(platao, luna)$ leia-se: *Platão é o pai de Luna*
- $pai(platao, pricles)$ leia-se: *Platão é o pai de Péricles*
- $pai(epimenides, platao)$ leia-se: *Sócrates é o pai de Platão*
- Alguém que é avô tem um filho que é um pai de alguém

Regras em Lógica – os pais!

- $pai(platao, luna)$ leia-se: *Platão é o pai de Luna*
- $pai(platao, pricles)$ leia-se: *Platão é o pai de Péricles*
- $pai(epimenides, platao)$ leia-se: *Sócrates é o pai de Platão*
- Alguém que é avô tem um filho que é um pai de alguém
- Alguém que é irmão tem o mesmo pai e não é irmão consigo mesmo

Regras em Lógica – os pais!

- $pai(platao, luna)$ leia-se: *Platão é o pai de Luna*
- $pai(platao, pricles)$ leia-se: *Platão é o pai de Péricles*
- $pai(epimenides, platao)$ leia-se: *Sócrates é o pai de Platão*
- Alguém que é avô tem um filho que é um pai de alguém
- Alguém que é irmão tem o mesmo pai e não é irmão consigo mesmo
- As regras estão nos slides de lógica
- $\Rightarrow$ as saídas são **particularizações** (PU e PE)

Regras em PICAT (1)

```
1  %%% FATOS ...  desenhe a arvore geneologica
2  index(-,-)
3          pai(platao, luna).
4          pai(platao, pericles).
5          pai(platao, eratostenes).
6          pai(epimenides, platao).
7  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
8  listar_pais  ?=>
9            pai(X,Y) ,
10           printf("\n ==> %w  e pai de  %w", X , Y)  ,
11           false.
12
13 listar_pais =>
14           printf("\n ")  ,
15           true. %% the final rule of above
16
17 listar_avos  ?=>
18           avo(X,Y) ,
19           printf("\n ==> %w  e avo de  %w", X , Y)  ,
20           false.
21
```

Regras em PICAT (2)

```
listar_avos =>
          printf("\n ")   ,
          true. %% the final rule of above
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
avo(X,Y) ?=> pai(X,Z), pai(Z,Y).

irmao(X,Y) ?=> pai(Z,X), pai(Z,Y), X !== Y.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% main padrao
main ?=> listar_pais,
        avo(X,Y), printf("\n ==> %w  eh avo de  %w", X , Y)   ,
    irmao(Z,W), printf("\n ==> %w  eh irmao de  %w", Z , W),
    false.
main => true.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Conclusão

- PICAT é uma linguagem nova (2013), desconhecida, revolucionária e com um futuro promissor para áreas de pesquisas e utilização comercial.
- Atualmente há pouco material disponível e uma comunidade pequena de usuários
- Uso acadêmico
-

Referências

- O User Guide que está no diretório doc/
- `https://github.com/claudiosa/CCS/tree/master/picat`
- `http://picat-lang.org/`
- Assinem o fórum do PICAT
- Site do Hakan Kjellerstrand
- Site do Román Bartak
- Site do Dymichenko

Obrigado

Retornem os comentários para .....