

Universidade Federal do Amazonas

Faculdade de Tecnologia FT

Prolog, uma rápida introdução - Tutorial 3

Manaus, setembro de 2024

Aluno: Thiago Rodrigo Monteiro Salgado – 21954456

Professor: Edjard

# Introdução

O tutorial 3 contido no documento fornecido pelo professor [1], apresenta uma visão geral sobre conceitos essenciais de Prolog e suas aplicações na modelagem lógica. O objetivo é uma familiarização inicial com a linguagem.

# Tutorial 3

Na página 19 do documento [1], é indicado o que deve ser feito no tutorial 3, como podemos ver abaixo:

* Definir dois predicados para verificar se o tamanho da lista é ímpar ou par. Um exemplo de consulta é evenSize([4,5,5,1]), a resposta deve ser verdadeira (true), mas para evenSize([4,5,5]), a resposta deve ser falsa. Para a consulta oddSize([4,5,5,1]), a resposta deve ser falsa, e para oddSize([4,5,5]), a resposta deve ser verdadeira.
* Ainda foi solicitado que fosse definida a relação de reversão que associa uma lista à sua versão invertida. Por exemplo, para a consulta reverseOf([a,1,3,b],L), o valor de L deve ser a lista invertida [b,3,1,a].

## EvenSize

* **Caso base**: Uma lista vazia é considerada de tamanho par. Isso é representado pela regra evenSize([]). ;
* **Passo recursivo**: Para listas não vazias, removemos dois elementos (a cabeça e o próximo elemento da lista) e aplicamos a função recursivamente ao restante da lista. Se a lista restante também tiver um tamanho par, podemos concluir que a lista original tinha um número par de elementos. Isso é feito pela regra evenSize([\_,\_|T]) :- evenSize(T). .

OddSize

* **Caso base**: Uma lista com um elemento tem tamanho ímpar, representado por oddSize([\_]). ;
* **Passo recursivo**: Já no caso par,vamos remover dois elementos da lista e identificar se o restante tem um tamanho ímpar. Se a lista restante for ímpar, então a lista original também é. Isso é feito pela regra oddSize([\_,\_|T]) :- oddSize(T). .

ReverseOf

* **Caso base**: A lista vazia é reversa de si mesma, representado por reverseOf([], []). ;
* **Passo recursivo**: Para reverter uma lista não vazia, removemos o primeiro elemento (a cabeça da lista) e aplicamos o predicado recursivamente ao restante da lista (a cauda). Após reverter a cauda, adicionamos o elemento removido (a cabeça) ao final da lista invertida. Isso é feito pela regra reverseOf([H|T], Reversed) :- reverseOf(T, TempReversed), append(TempReversed, [H], Reversed). .

# Resultados









