**Estudo sobre a Linguagem de Programação Haskell**

**Gabriel Carvalho Iann Oliveira**

Departamento de Ciência da Computação – Universidade Federal de Roraima (UFRR)

Boa Vista – RR – Brasil

[Gabriel10araujo@hotmail.com](mailto:Gabriel10araujo@hotmail.com), iann\_oliver25@hotmail.com

**Resumo.** Aparecendo pela primeira vez em 1990, Haskell, uma linguagem puramente funcional e de alto nível, vem tomando espaço dentre outras linguagens funcionais existentes por ser projetada para ser uma linguagem única que englobe todas as melhores ideias práticas já feita. Utilizado em grandes empresas como Facebook, Intel e google, haskell se destaca pelo seu rápido desenvolvimento de softwares robustos, altamente flexível para manutenção e de grande interação com outras linguagens.

1. **História do surgimento da linguagem**

## Desenvolvido em 1930 pelo matemático Alonzo Church, o cálculo Lambda (λ-Cálculo), um modelo matemático que descreve relações entre funções, foi o grande responsável pelo início da programação funcional no mundo. O cálculo lambda foi estendido para constituir a primeira linguagem funcional, o LISP, desenvolvida pelo cientista da computação John Mcarthy em 1958, um pouco depois do FORTRAN.

## Em 1987 o número de linguagens funcionais existentes gerou desconforto na comunidade de Programação funcional, foi aí que surgiu a ideia de uma única linguagem que englobe não só todas as melhoras práticas já feitas, mas também ideias que já estavam sendo discutidas na época. Foi então que, em 1990, tendo como base a linguagem Miranda, foi liberada a primeira versão de Haskell, nomeada em homenagem ao matemático Haskell Brooks Curry.

1. **Domínios de aplicação**

Dentre diversas aplicações de haskell, destacam-se:

- Aplicações de IA:

* Sistemas Especialistas;
* Representação de conhecimento;
* Processamento de linguagem natural;

- Jogos;

- Compiladores;

Há diversas empresas que adotam haskell como por exemplo:

- Google

Haskell é utilizado em poucos projetos internos para suporte interno na infraestrutura de TI.

- Intel

Desenvolvido pela intel, um compilador em Haskell como parte das pesquisas de paralelismo de múltiplos núcleos em grande escala.

- Keera Studios

Empresa Europeia de desenvolvimento mobile, desktop, aplicativos web e jogos.

1. **Paradigmas suportados pela linguagem**

A linguagem de programação Haskell, tem como paradigma o funcional, sendo baseado em funções diferentemente do paradigma imperativo, ele utiliza expressões em formas de funções sem efeitos colaterais, ou seja, a função ira somente calcular algo e devolve-lo um resultado.

Uma das principais vantagens da linguagem funcional é a ausência de operações de atribuição tornando os programas funcionais muito mais simples para provas e análises matemáticas, também por a linguagem funcional não ser dependente das operações de atribuição permite aos programas avaliações em diferentes ordens, isso torna as linguagens funcionais umas das melhores para programação de computadores paralelos. Uma desvantagens desse paradigma é sua pouca eficiência.

1. **Variáveis e tipos de dados**

Em Haskell, as variáveis não variam. Ou seja, tecnicamente, são constantes. As variáveis assumem um valor quando são criadas e esse valor nunca será mudado.

Utilizamos a notação:

e :: T

indica que e é um valor do tipo T.

Exemplos na prática:

x :: Int - Inteiros de precisão fixa (0,3,-10)

x :: Integer - Inteiros de precisão arbitrária (0,3,-1)

x :: Float - Números de ponto flutuante (1.5,3.9,-4.3)

x :: Char - Caracteres simples (‘a’,’b’,’c’)

x :: String - Vetor de caracteres (“valor”,”nome”,”dia”)

x :: Bool - Valores lógicos (True,False)

Para lista utilizamos a notação:

[e] :: [T]

1. **Comando de controle**

**IF**

maiordeidade :: int -> String

maiordeidade x

| x >= 18 = “Maior de idade”

| otherwise = “Menor de idade”

**Case**

Case x of

0 -> 1

1 -> 2

\_ -> 0

Haskell por não ser uma linguagem imperativa e sim declarativa, não possui comandos como while, for ou algum outro comando de loops. Enquanto linguagens imperativas dizem como fazer, Haskell o programador precisa dizer o que fazer.

Exemplo:

Codigo para adicionar valores pares entre 0 e 10 em uma lista.

**C#**:

Var results = new List<int32>();

For (Int32 i = 1; i<10;i++)

If(i%2 == 0)

Results.Add(i);

**Haskell**:

Filter even [1..10] where even n = n ‘rem’ 2==0

1. **Escopo (regras de visibilidade)**

O escopo de uma variável significa até onde essa variável poderá ser reconhecida. No caso em Haskell, é estabelecido que o escopo de uma variável ou se extende até o primeiro parêntese descasado encontrado a partir da variável, ou atinja o final da λ-expressão.

1. **Exemplo prático de uso da linguagem de programação**
2. **Conclusão**

Apesar de parecer complicado, Haskell demonstra uma grande vantagem e aplicabilidade quando bem estudada. Mesmo com a popularidade de outros paradigmas, o paradigma funcional anda crescendo largamente nos últimos anos como o F# lançado pela Microsoft. Twitter utilizando Scala como back-end e até C# incorporando conceitos funcionais em sua linguagem. São provas que Haskell e todas as linguagens funcionais são importantes.

1. **Referências**

Luiz Borba. (2010) “Porque linguagens funcionais são importantes”, <http://borba.blog.br/2010/04/porque-linguagens-funcionais-sao-importantes/>.

Marcelo Parrela. (2014) “Haskell Básico – A História!”, <https://marceloparrela.wordpress.com/2013/11/01/haskell-a-historia/>.

Simão Melo. (2013) “Introdução ao Cálculo Lambda”, <https://www.di.ubi.pt/~desousa/2013-2014/CF/apresentacao-lambda.pdf>.

Victor Luiz. “Programação funcional em Haskell”, <http://www.ebah.com.br/content/ABAAAANLoAA/programacao-funcional-haskell?part=6#>.

Iorgama Porcely. “Introdução a linguagem de programação Haskell”, <http://www.slideshare.net/iorgama/haskell-15676502>.

“Paradigma Funcional: Coneitos básicos”, <http://din.uem.br/ia/ferramentas/lisp/lisp3.htm>.

Haskell in industry, “https://wiki.haskell.org/Haskell\_in\_industry”.