História do Haskell

- 1930: ALONZO CHURCH DESENVOLVEU O CÁLCULO DE LAMBDA, UM SIMPLES E PODEROSO TEOREMA DE FUNÇÕES, QUE É BASE DO HASKELL.
- NA DÉCADA DE 1970 , O CONCEITO DE AVALIAÇÃO PREGUIÇOSA JÁ ESTAVA NO MEIO ACADÊMICO .
- SETEMBRO DE 1987, CRIAÇÃO DE UM COMITÊ COM O OBJETIVO DE CONSTRUIR UM PADRÃO ABERTO PARA LINGUAGENS FUNCIONAIS .
- 1990- SURGIU HASKELL EM HOMENAGEM AO LÓGICO HASKELL BROOKS CURRY

O que é a programação funcional?

- É um paradigma de programação
- No paradigma imperativo, um programa é uma sequência de instruções que mudam.
- No paradigma funcional, um programa é um conjunto de definições de funções que aplicamos a valores.

Exemplo: somar os naturais de 1 a 10

- Em linguagem C:

```
int total = 0;
for (i=1; i<=10; ++i)
   total = total + i;
```

- O programa é uma sequência de instruções.
- O resultado é obtido por mutação das variáveis i e total.

Exemplo: somar os naturais de 1 a 10

- Em Haskell: sum [1..10]

- O programa consiste na aplicação da função sum à lista dos inteiros entre 1 e 10.

Vantagens da programação funcional

- Nível mais alto:

  Programas mais concisos

- Reuso Haskell facilita e simplifica refatoração e reuso

- Código menor, mais claro e mais fácil de manter.

- Avaliação lazy

Desvantagens da programação funcional

- Compiladores/interpretadores mais complexos;
- Difícil prever os custos de execução (tempo/espaço);

COMPILADA OU INTERPRETADA

- **HUGS**

1. **INTERPRETADOR**
2. **ESCRITO EM C**
3. **PORTÁVEL**
4. **LEVE**

GHC

COMPILADOR

ESCRITO EM HASKELL

MENOS PORTÁVEL

MAIS LENTO

EXIGE MAIS MEMÓRIA

PRODUZ PROGRAMAS MAIS RÁPIDOS

Características

- Concisa;
- Fácil compreensão;
- Forte sistema de tipo;
- Avaliação lazy;
- Abstração poderosa;
- Gestão automática de memória.
- Alta portabiliddae

Aplicações

- Computação simbólica;
- Processamento de listas;
- Aplicações científicas;
- Aplicações em IA

  Sistemas especialistas

  Representação do conhecimento
- Compiladores;
- Jogos.

SINTAXE

- Trabalha-se somente com funções,
- É case- Sensitive;
- Não possui comandos de repetição como While e for;
- Não é orientada a objetos.

Orientação a objetos Vs Funcional

- Programação orientada a objetos é um estilo de Programação que permite:

  Reuso de código (via classes)

As linguagens Funcionais aproveitam o código através de funções.

Tipos em Haskell

- **A linguagem Haskell suporta os tipos primitivos bool, char, string, int, integer, float.**

- **Tipo variável / funções polimórficas**

  head : : [a] -> a

- **Os tipos compostos da linguagem são: Tipo de lista –Uma lista é uma sequência de elementos do mesmo tipo.**

  Exemplo:[False,True,False]::[Bool]

- **Tipo de tupla-uma sequência finita de componentes de diferentes tipos**
- **Exemplo:(False,True,'a')::(Bool,Bool,Char)**

AMARRAÇÕES

- TODOS OS TIPOS SÃO CONHECIDOS EM TEMPO DE COMPILAÇÃO
- INFERÊNCIAS DE TIPOS
- TIPO ESTÁTICO

Comparação Com Outras Linguagens

Quicksort in C

```
// To sort array a[] of size n: qsort(a,0,n-1)

void qsort(int a[], int lo, int hi) {
{
   int h, l, p, t;
   if (lo < hi) {
   l = lo;
   h = hi;
   p = a[hi];
      do {
        while ((l < h) && (a[l] <= p))
            l = l+1;
         while ((h > l) && (a[h] >= p))
            h = h-1;
```

Comparação Com Outras Linguagens

```
if (l < h) {
            t = a[l];
            a[l] = a[h];
            a[h] = t;
}
} while (l < h);
a[hi] = a[l];
a[l] = p;

qsort( a, lo, l-1 );
qsort( a, l+1, hi );
}
}
```

Comparação Com Outras Linguagens

- ***Quicksort in Haskell***

- qsort [] = []
- qsort (x:xs) = qsort (filter (< x) xs) ++ [x] ++ qsort (filter (>= x) xs)