Seminário Programação Funcional

MIRANDA

Dupla: Jonas Lopes e Samuel Marques

Prof: Ricardo Reis

Introdução

Características:

- Preguiçosa
- Puramente Funcional
- Não estrita
- Polimórfica
- Alta Ordem
- Poderoso sistema de módulos

Histórico

Projetada por David Torner, da Universidade de Kent
Sucessora das linguagens:

- SALS (St Andrews Static Language)
- KRC (Kent Recursive Calculator)

Produzida pelo Research Software Ltd. of England

Histórico

Primeira linguagem puramente funcional a ser suportada comercialmente.

Foi lançado em 1985, como um rápido interpretador em C para sistema operacional Unix-flavour, com subsequente liberação, em 1987 e 1989.

Mais tarde a linguagem de programação Haskell surge e é semelhante a muitos formatos de Miranda.

Nomenclatura

Por que do nome Miranda?

Miranda que no Latim significa “para ser admirado em”

O seu primeiro uso como nome próprio a uma menina foi devido a Shakespeare, em sua peça teatral A Tempestade (1611).
No ato 5, cena 1 da peça Miranda faz um discurso com as palavras: “O Brave New World!” (“O Bravo Novo Mundo!”).

A ideia é de que a linguagem Miranda é uma introdução ao Bravo Novo Mundo da programação funcional.

Estrutura

Um programa em Miranda é normalmente 5 a 15 vezes mais curto do que seu correspondente em C ou Java.

- Miranda é considerada uma linguagem puramente funcional e **sem efeitos colateral.**
  - Em Ciência da Computação, se diz que uma operação, função ou expressão tem efeitos colaterais se esta, além de retornar um valor, modifica o estado de alguma variável fora do seu ambiente local (seu escopo))

- Alta ordem (suporte a *functional data*)

Estrutura

- Avaliação preguiçosa (suporta funções não estritas e objetos de dados infinitos)
  - Avaliação preguiçosa (ou call-by-need) é uma estratégia de avaliação que retarda a avaliação de uma expressão até que esta seja necessária, e que também evita avaliações repetidas

- Compreensão de listas

O seu primeiro uso como nome próprio a uma menina foi devido a Shakespeare, em sua peça teatral *A Tempestade* (1611).
No ato 5, cena 1 da peça Miranda faz um discurso com as palavras: *"O Brave New World!"* (*"O Bravo Novo Mundo!"*).

Estrutura

- Tipagem forte polimórfica
  - Usando o polimorfismo, a função ou tipo de dado pode ser escrita genericamente para que possa suportar valores idênticos sem depender de seu tipo.

- Tipos e módulos abstratos de dados
  - Nos casos em que um módulo define um novo tipo de dado e o conjunto de operações para manipular dados desse tipo, falamos que o módulo representa um tipo abstrato de dados.

Estrutura

O compilador de Miranda é incorporado como um sistema interativo implementado sob UNIX fornecendo acesso a um editor de tela.

Os arquivos Miranda tem a extensão .m – Exemplo, questao1.m

Tipos de Dados

Os tipos básicos embutidos em Miranda são:

- Números (inteiros de tamanho ilimitado e float de precisão dupla)
- Caracteres
- Listas
- Tuplas
- Funções

Aqui o tipo String é simplesmente uma lista de caracteres

Tuplas são sequências de elementos que podem ser de tipos diferentes, assim como em Haskell, e são escritas delimitadas por parênteses.

Exemplos:

Execução básica de código Miranda:

Execução de operações

```
Miranda 3 ^ 3
27

Miranda 4.0 ^ -1
0.25

Miranda days div 7
52
```

Arredondar para inteiro

```
Miranda entier 3.2
3

Miranda entier 3.7
3
```

Código do caracter na Tabeça ASCII

```
Miranda code 'A'
65

Miranda decode 65
'A'
```

Erro divisão inteiro por fracional

```
Miranda 365 div 7.0
program error: fractional number where integer expected (div)
```

Exemplos:

Função show

```
Miranda "mir" ++ (show 123) ++ "anda"
mir123anda

Miranda "mir" ++ (shownum 123) ++ "anda
mir123anda

Miranda (shownum (12 * 3)) ++ "anda"
36anda
```

Operadores lógicos

```
Miranda (y = 42) & (x = 0) & (y = x)
False

Miranda (y = 42) & ((x = 0) & (y = x))
False

Miranda (x = 0) \/ ((y div x) > 23)
True
```

Table 1.3 Logical operations on Booleans.

~	logical negation
&	logical conjunction
\/	logical disjunction

Exemplos:

Todos os número primos:

```
nats = [1..]

evens = [2,4..]

negs = [-1,-2..]

fibs = [a | (a,b) ← (1,1), (b,a+b)..]

primes = sieve [2..]
         where
         sieve(p : x) = p : sieve[n←x | n rem p~=0]
```

Exemplos:

Fibonacci:

fibs.m

```
fibs = map fib [0..]
fib 0 = 0
fib 1 = 1
fib (n+2) = fibs!(n+1) + fibs!n

test = layn (map shownum fibs)
```

Fibonacci com lista de compreensão:

```
fibs = [a | (a,b) <- (0,1), (b,a+b) ..]
```

Exemplos:

QuickSort:

quicksort.m

```
qsort [] = []
qsort (a:x) = qsort [b|b<-x;b<=a]
                  ++ [a] ++
              qsort[b|b<-x;b>a]
```

Exemplos:

Foldr, product, sum, and e or:

```
foldr op z = g
             where
             g [] = z
             g (a : x) = op a (g x)

product = foldr (*) 1
sum = foldr (+) 0
and = foldr (&) true
or = foldr (∨) false

example:-  product[1,2,3,4] = 24
```

Exemplos:

Miranda usa equações guards ao invés de expressões condicionais para expresser análise de caso.

```
gcd a b  = gcd (a-b) b, a>b
         = gcd a (b-a), a<b
         = a, a=b
```

OBRIGADO!