Faculdade de Computação
Programação Funcional (BCC/BSI) - 1° Período
Aula Prática: Tuplas em Haskell

Tuplas

Numa tupla podemos combinar os componentes de um dado numa única estrutura, e os componentes podem ter tipos e propriedades distintas. Exemplos:

```
(True, 1, 1)
("Alo Mundo", False)
(4, 5, "Seis", True, 'b')
((3,'a'), [1,2,3])
```

As tuplas podem ser utilizadas para modelar itens do mundo real, como coleções de itens. Por exemplo, uma pessoa pode ser representada por seu nome, telefone, endereço e ano de nascimento. Desta forma, a tupla é formada por valores de tipos diferentes:

```
maria :: (String, String, String, Int)
maria = ( "Maria Silva", "222-2222", "Rua A, 35", 1998)
```

Para mostrar que queremos usar essa tupla para representar pessoas, podemos definir um tipo Pessoa a partir da definição da tupla.

```
type Pessoa = (String, String, String, Int)
maria::Pessoa
maria = ( "Maria Silva", "222-2222", "Rua A, 35", 1998)
```

Programando com Tupla:

O exemplo a seguir mostra a definição de tipos para uma **data**, sendo representada por uma *tupla-3*, em que a primeira posição é o dia, a segunda é o mês e a terceira é o ano.

```
type Data = (Int,Int,Int)
```

Podemos escrever um programa que dada uma determinada data, verifica se a mesma é válida:

```
valida::Data->Bool
valida (d,m,a)
      | d >= 1 && d <= 31 && (m == 1 || m == 3 || m == 5 ||
           m == 7 || m == 8 || m == 10 || m == 12) = True
      | d >= 1 && d <= 30 && (m == 4 || m == 6 || m == 9 ||
           m == 11) = True
      | d >= 1 && d <= 28 && m == 2 && not (bissexto a) = True
      | d >= 1 && d <= 29 && m == 2 && (bissexto a) = True
      | otherwise = False

bissexto:: Int-> Bool
bissexto x | (mod x 400 == 0) = True
           | (mod x 4 == 0) && (mod x 100 /= 0) = True
           | otherwise = False
```

Faculdade de Computação
Programação Funcional (BCC/BSI) – 1° Período

Aula Prática: Tuplas em Haskell

Exercícios:

1) Seja o cadastro de pessoas dado pela função a seguir:

```
import Char
type Pessoa = (String, Int, Float, Char)

pess :: Int->Pessoa
pess x
      |x==1 = ("Cristina", 27, 1.66,'F')
      |x==2 = ("Flávio", 26, 1.85,'M')
      |x==3 = ("Mariana", 67, 1.55,'F')
      |x==4 = ("Cecília", 48, 1.78, 'M')
      |x==5 = ("Paulo", 24, 1.93, 'M')
      |x==6 = ("Clara", 38, 1.70,'F')
      |x==7 = ("Rodrigo", 12, 1.85, 'M')
      |x==8 = ("Giovana", 31, 1.58,'F')
      |x==9 = ("Daniel", 75, 1.74, 'M')
      |x==10 = ("Eduardo", 21, 1.69,'F')
      |otherwise = ("Acabou!",0, 0.0, 'x')
```

Construa funções que retornem os seguintes dados:

- Dados dois números de pessoa, retornar o número da pessoa mais alta.
- A diferença de altura entre duas pessoas, dados seus números.
- O número da pessoa mais alta

2) Reescreva a função para calcular uma equação do segundo grau ($ax^2 + bx + c$), retornando as duas raízes (se houver) numa tupla-2.

3) Dados três comprimentos de lados, verifique se podem formar um triângulo.

Observações:
O comprimento de um lado do triângulo é sempre menor do que a soma dos outros dois.
Eqüilátero > Todos lados iguais
Isósceles > Dois lados iguais
Escaleno > Todos os lados diferentes

4) Dado um valor monetário em Reais, faça um programa que devolve uma tupla-3 contendo o valor em Real, e sua conversão para Euro (1Real = 0.448Euro) e Dolar (1R = 0.547USD), como no exemplo abaixo:

```
> valorRealConvertido 500.8
((500.8,"Real"),(224.3584,"Euro"),(273.9376,"Dolar"))
```