Paradigmas de Programação

Prof. Maicon R. Zatelli

Haskell - Programação Funcional
Módulos

Universidade Federal de Santa Catarina
Florianópolis - Brasil

Haskell

Módulos em Haskell contém:

- Declarações de precedência (`fixity`)
- Declarações de tipos
- Declarações de classes e instâncias das mesmas
- Definições de tipos
- Definições de funções

O uso de módulos traz algumas vantagens:

- Permitem reuso de código
- Facilitam a organização e manutenção do código

Haskell

fatorial.hs

```
module Fatorial (fatorial) where

-- Recebe um Int e retorna um Int
fatorial :: Int -> Int
fatorial 0 = 1
fatorial n = n * fatorial (n-1)
```

- Aqui crio um módulo chamado Fatorial, o qual exporta a função fatorial.
- Posso exportar apenas algumas funções ou todas elas.

Haskell

principal.hs

```
module Main (main) where

import Fatorial

main = do
    nString <- getLine
    let n = (read nString :: Int)
    print (fatorial n)
```

- Declarar o módulo `Main` é opcional, mas se for dar um nome para o módulo que contém o main, ele deve ser necessariamente o nome Main.
- O `import` me permite utilizar as funcionalidades implementadas no módulo `Fatorial`.
- Para compilar utilize um dos seguintes comandos:
  - ghc −o arquivoExecutavel fatorial.hs principal.hs
  - ghc −−make principal.hs
  - Neste caso, os nomes dos módulos e dos .hs devem ser iguais

Haskell

formas.hs

```
module Formas (Forma (Retangulo,
                      Elipse,
                      Circulo,
                      Triangulo,
                      Poligono), area) where

data Forma = Retangulo Float Float
           | Elipse Float Float
           | Circulo Float
           | Triangulo Float Float
           | Poligono [(Float, Float)]
           deriving Show

area :: Forma -> Float
area (Retangulo base altura) = base * altura
area (Triangulo base altura) = base * altura
area (Elipse raio1 raio2) = pi * raio1 * raio2
area (Circulo raio) = pi * raio * raio
```

Haskell

principal.hs

```
module Main (main) where

import Formas

main = do
    print (area (Retangulo 5 2))
    print (area (Circulo 5))
```

Haskell

```
import Prelude hiding (max,toUpper,isDigit)
```

- Posso também ignorar algumas funcionalidades de algum módulo que estou importando. Para isso, uso a palavra reservada `hiding`.
- No exemplo acima, importo o módulo Prelude e ignoro as funções `max`, `toUpper` e `isDigit`.

```
import Pilha (Pilha (Stack), emptyStack, push, pop, top)
```

- Também posso decidir quais funcionalidades importar informando cada uma delas no `import`.

Haskell - Alguns Links Úteis

- `http://learnyouahaskell.com/modules`
- `https://www.haskell.org/tutorial/modules.html`
- `https://en.wikibooks.org/wiki/Haskell/Modules`

Haskell

Ver atividade no Moodle