Seu nome: < Marcos José Melo Aguiar >
Turma: < 2022.1 - PP - BCC >

Questão 1 (400 pontos) - No final dos slides da aula sobre listas em Haskell estão descritos os tipos de uma biblioteca e as funções que são usadas para realizar consultas, empréstimos e devolução de livros. Considere como possível implementação para as funções `emprestimo` e `livros` o código a seguir.

```
-- retorna todos os livros com uma pessoa
livros :: BancoDados -> Pessoa -> [Livro]
livros bd pessoa = [l | (p, l) <- bd, p == pessoa]

-- retorna as pessoas que pegaram emprestado um determinado livro
emprestimos :: BancoDados -> Livro -> [Pessoa]
emprestimos [] l = []
emprestimos ((p,l2):as) l  | (l2 == l) = p : emprestimos as l
                           | otherwise = emprestimos as l
```

Este exercício usa uma nova versão da biblioteca com os tipos a seguir. Para os tipos que seguem é que você precisar criar as funções pedidas.

```
type Pessoa = String
type Livro = String
type Emprestado = Bool
type BancoDados = [(Pessoa,Livro,Emprestado)]

exemploBD :: BancoDados
exemploBD = [("Leandro","Java",True),("Joabe", "CSP",
True),("Lucas", "UML", True), ( "Lucas" , "Haskell", True),(
"Sidney" , "CSP", True),( "" , "Java", False),( "" ,
"Concorrencia", False)]
```

Na nova biblioteca, livros que não estão emprestados têm no primeiro elemento da tripla a string vazia para o nome da pessoa que pegou o livro, no segundo o nome do livro e no terceiro o valor False. Livros que estão emprestados possuem um nome não vazio e o valor True .

Escreva o corpo das funções Haskell a seguir que implementam operações da nova biblioteca.

```
-- retorna os livros do acervo, sem repetição
livros :: BancoDados -> [Livro]

-- retorna True se o livro l está disponível
livroDisponivel :: BancoDados -> Livro -> Bool

-- retorna os livros que estao com uma pessoa
livrosPessoa :: BancoDados -> Pessoa -> [Livro]

-- retorna pares com nome do livro e a pessoa que está emprestado
emprestimos :: BancoDados -> [(Livro,Pessoa)]

-- adiciona emprestimo, se livro está disponível
emprestar :: BancoDados -> Pessoa -> Livro -> BancoDados

-- devolve livre, se está emprestado
devolver :: BancoDados -> Pessoa -> Livro -> BancoDados
```

Na implementação:

- Pode criar funções auxiliares e/ou usar funções da biblioteca de Haskell.
- Use uma função de alta ordem que foi vista na disciplina em pelo menos uma função.
- Use compreensão de lista em pelo menos uma função.
- Use a implementação de uma função que você mesmo definiu em outra que você vai definir. Exemplo: emprestar pode usar livroDisponivel para saber se o livro está disponível antes de modificar a base da biblioteca.
- Deixa na forma de comentário, logo depois da definição da cada função que você vai implementar, pelo menos um teste. Segue exemplo de teste para a função livros:

```
-- livros exemploBD
-- retorna [“Java”, “CSP”, “UML”, “Haskell”, “Concorrência”]
```

Cole dentro desta caixa o link para o projeto do replit.com com o seu código.

Você pode usar qualquer ferramenta que desejar para desenvolver o código, O repl.it será usado para compartilhar o código comigo e facilitar a correção, pois poderei executar o código online sem precisar baixar.

Cole o seu código dentro desta caixa.

Este passo é redundante, porém vai me ajudar a inserir comentários no seu código.