Disciplina: Paradigmas de Programação
Professor: Maicon Rafael Zatelli
Entrega: *Moodle*

Atividade VI - Haskell

Atenção: Faça um ZIP com todos os arquivos de solução. Use o nome do arquivo de maneira a entender qual problema você está resolvendo. Por exemplo, problema1.hs, problema2.hs e assim por diante.

Resolva os seguintes problemas na linguagem Haskell:

1. Crie um tipo de dados Aluno, usando **type**, assim como criamos um tipo de dados Pessoa. O tipo Aluno deve possuir um campo para o nome, outro para a disciplina e outros três campos para notas. Agora, execute os passos abaixo:

   A: Crie uma função no mesmo estilo que a função pessoa, vista em sala e disponível nos slides no Moodle, ou seja, que receba um inteiro e retorne um Aluno correspondente ao valor inteiro.

   B: Crie alguns alunos de exemplo, assim como também feito no exemplo da pessoa.

   C: No main, imprima o primeiro nome de um aluno, portanto crie uma função para obter o primeiro nome.

   D: Crie uma função que receba um Int e retorne a média do aluno correspondente.

   E: Crie uma função que calcule a média da turma, ou seja, considerando todos os alunos. DICA: crie uma função recursiva que receba o primeiro identificador de aluno e incremente o identificador a cada chamada recursiva, até chegar no último aluno. Não use listas!

2. Altere nosso exemplo da forma e inclua uma nova forma (Triangulo) no construtor do tipo Forma e também calcule sua área.

3. Crie um novo tipo Ponto, usando **data**, que pode ser um ponto 2D ou um ponto 3D. Depois, crie uma função que receba dois pontos (necessariamente ambos sendo 2D ou ambos sendo 3D), e retorne a distância entre eles.

4. Motifique o arquivo `arvore.hs` (disponível no Moodle) de forma a adicionar novas operações a nossa árvore:

   A: Crie uma função com a seguinte assinatura: `ocorrenciasElemento :: Arvore -> Int -> Int`, a qual recebe um número e deve retornar a quantidade de ocorrências dele na árvore.

   B: Crie uma função com a seguinte assinatura: `maioresQueElemento :: Arvore -> Int -> Int`, a qual recebe um número e deve retornar a quantidade de números maiores que ele na árvore.

   C: Crie uma função com a seguinte assinatura: `mediaElementos :: Arvore -> Float`, a qual deve retornar a média dos números na árvore. DICA: utilize a função `fromIntegral` para converter um tipo inteiro para um tipo compatível com o operador de divisão `/`

   D: Crie uma função com a seguinte assinatura: `quantidade :: Arvore -> Int`, a qual deve retornar a quantidade de elementos na árvore.

   E: Crie uma função com a seguinte assinatura: `elementos :: Arvore -> [Int]`, a qual deve retornar uma lista com todos os elementos na árvore.

5. Pesquise o que é o **newtype**. Qual é a diferença dele para o **type** e para o **data**? Faça um pequeno exemplo de aplicação e explique seu funcionamento.