Universidade Federal de Uberlândia
Faculdade de Computação
Professor: Wendel Melo

Primeira lista de exercícios de Programação Funcional (2017-1)

Atenção: Procure separar a computação em diversas funções para a realização dos procedimentos necessários.

Questão 1: Desenvolva um programa em Haskell que leia do teclado o valor do raio de um círculo e imprima seu diâmetro, perímetro e área. Seu programa deve definir uma função para o cálculo do diâmetro, outra para o cálculo do perímetro e uma terceira para o cálculo da área:

Exemplo:

```
Entre com o valor do raio do circulo: 10
Diametro: 20
Perímetro: 62.83
Area: 314.15
```

Questão 2: Um dia tem 24 horas ou 1440 minutos. Desenvolva uma função chamada *horario2minuto* que recebe dois números inteiros representando um horário H (hora e minuto) e retorna o minuto do dia ao qual o horário recebido corresponde (número inteiro). Observe os exemplos a seguir (note que só é solicitado o desenvolvimento de uma função, e não de um programa completo com entrada e saída):

```
> horario2minuto 3 15     -- calculando o minuto relativo ao horario 3:15
195
> horario2minuto 17 42    -- calculando o minuto relativo ao horario 17:42
1062
```

Questão 3: Desenvolva uma função chamada *minuto2horario* que realiza a operação inversa do exercício anterior, isto é, a função recebe um inteiro representando os minutos relativos ao horário H e retorna uma lista com dois componentes inteiros relativos a hora e minuto de H, conforme os exemplos:

```
> minuto2horario 200      -- calculando o horario relativo ao minuto 200
[3, 20]
> minuto2horario 1070     -- calculando o horario relativo ao minuto 1070
[17, 50]
```

Questão 4: Um trem de longa distância parte de uma estação inicial em um determinado horário H_1 (hora e minuto) e chega à estação final em um determinado horário H_2. Faça um programa em Haskell que leia os horários de partida e chegada do trem e informe o tempo total de viagem (em hora e minuto). Assumindo que a velocidade média do trem é de 100 km/h, informe também a estimativa da distância percorrida entre as estações. Assuma ainda que as viagens sempre iniciam e terminam no mesmo dia. Não se esqueça de definir funções separadas para realizar os cálculos necessários.

Exemplo:

```
Entre com a hora de partida: 10
Entre com o minuto de partida: 35

Entre com a hora de chegada: 11
Entre com o minuto de chegada: 25

O trem partiu as 10:35 e chegou as 11:25.
Tempo de viagem: 00:50
Distancia percorrida: 83.3 km
```

Questão 5: Escreva um programa que leia os comprimentos dos lados de um triângulo e informe se o mesmo é equilátero, isósceles ou escaleno. Você deve definir uma função que recebe os comprimentos dos lados e retorna uma String com a respectiva classificação do triângulo.

Questão 6: Faça um programa que leia do teclado um numero relativo a um dia da semana e imprima o nome do dia correspondente a esse numero. Assuma que a semana começa no domingo, conforme o exemplos:

```
> Entre com o dia da semana: 3
terca-feira

> Entre com o dia da semana: 7
sabado

> Entre com o dia da semana: 9
dia invalido
```

Questão 7: Defina uma função chamada *raizSegGrau* que recebe os coeficientes a, b e c de uma equação de segundo grau no formato:

$$ax^2 + bx + c = 0$$

e retorne uma lista contendo as raízes reais da respectiva equação. Note que essa lista pode vir a ser vazia (se a equação não tiver raízes reais), e que se a $a = 0$, sua função deve calcular uma raiz de equação de primeiro grau. Observe também que a lista de raízes não deve apresentar raízes repetidas. Além da função raizSegGrau, você também deve definir:

- Uma função para calcular uma raiz de equação de primeiro grau;
- Uma função para calcular o discriminante Δ;
- Uma função para calcular a raiz relativa a raiz quadrada positiva de Δ;
- Uma função para calcular a raiz relativa a raiz quadrada negativa de Δ;

Exemplos:

```
> raizSegGrau 1 (-5) 6
[3, 2]

> raizSegGrau 2 4 2
[-1]

> raizSegGrau 3 1 1
[]

> raizSegGrau 0 3 12
[-4]
```