1 Diversão com Eletrônica Analógica

Resistores são componentes elétricos / eletrônicos muito utilizados para redução de corrente em circuitos. A medida do quanto de resistência possui um resitor é medida em Ohms (Ω) e este valor é usualmente descrito usando um código formado por cores. As duas primeiras cores do código representam os dois dígitos mais significativos do valor, a terceira cor representa uma potência de 10.

A tabela seguinte apresenta o significado de cada uma das cores em suas respectivas posições:

Cor	1^a pos.	2^a pos.	3^a pos. (Mult.)
Preto	0	0	10^0
Marrom	1	1	10^1
Vermelho	2	2	10^2
Laranja	3	3	10^3
Amarelo	4	4	10^4
Verde	5	5	10^5
Azul	6	6	10^6
Violeta	7	7	10^7
Cinza	8	8	10^8
Branco	9	9	10^9
Ouro	-	-	10^{-1}
Prata	-	-	10^{-2}

Como exemplo, considere o seguinte código: Verde-Azul-Amarelo. Como as duas primeiras cores são Verde e Azul, temos que os dois dígitos mais significativos deste resistor são: 5 e 6. O multiplicador (terceira cor) deve ser 10^4.

Desta maneira, temos que o resistor de código Verde-Azul-Amarelo possui uma resistência de $56 \times 10^4\Omega$.

Considere o seguinte tipo de dados Haskell, que representa as cores utilizadas para codificar valores de resistência:

```
data Color = Black | Brown | Red | Orange | Yellow | Green
           | Blue | Violet | Gray | Gold | Silver
           deriving (Eq, Ord)
```

e a seguinte tabela que representa os possíveis valores associados a cada cor:

```
type Table = [(Color, [Maybe Float])]

table :: Table
table = [(Black,  [Just 0.0, Just 0.0, Just 0.0]),
```

```
(Brown,  [Just 1.0, Just 1.0, Just 10.0]),
(Red,    [Just 2.0, Just 2.0, Just 100.0]),
(Orange, [Just 3.0, Just 3.0, Just 1000.0]),
(Yellow, [Just 4.0, Just 4.0, Just 10000.0]),
(Green,  [Just 5.0, Just 5.0, Just 100000.0]),
(Blue,   [Just 6.0, Just 6.0, Just 1000000.0]),
(Violet, [Just 7.0, Just 7.0, Just 100000000.0]),
(Gray,   [Just 8.0, Just 8.0, Just 1000000000.0]),
(Gold,   [Nothing,  Nothing, Just 0,1]),
(Silver, [Nothing,  Nothing, Just 0,01])]
```

Observe que o valor `table` é uma lista de pares formados por um valor de tipo `Color` e uma lista de valores de tipo `Maybe Float`, onde uma posição da lista representa uma determinada coluna na tabela. Por exemplo, a cor vermelho (representada pelo valor `Red`) possui como valores associados à primeira, segunda e terceira coluna os números 2, 2 e 10^2 respectivamente e isto é representado no valor `table` pelo par `(Red, [Just 2.0, Just 2.0, Just 100.0])`. Quando um determinado valor não está presente na tabela, este é representado por `Nothing`.

Com base no anteriormente apresentado, desenvolva o que se pede:

1. Desenvolva a função

   ```
   value ::  Color -> Int -> Maybe Float
   ```

   que recebe como parâmetros uma cor e um número inteiro correspondente ao número de uma coluna da tabela de valores de cores e retorna o valor associado à cor e a respectiva posição desta na tabela. Caso o número da coluna seja inválido (menor que 0 ou maior que 3), sua função deve retornar como resultado o valor `Nothing`.

2. Considere agora, o seguinte tipo de dados que representa um resistor:

   ```
   data Resistor = Resistor [Color] deriving (Eq, Ord)
   ```

   Um resistor de código Vermelho-Marrom-Amarelo seria representado pelo seguinte valor:

   ```
   Resistor [Red, Brown, Yellow]
   ```

   (a) A representação de resistores utilizando o tipo `Resistor` permite que um resitor seja construído utilizando um número arbitrário de cores, o que é incorreto, já que o código de cores é formado apenas por três posições. Desenvolva a função

   ```
   valid ::  Resistor -> Bool
   ```

que retorna verdadeiro se o resistor fornecido como parâmetro possui 3 e somente 3 cores.

(b) Utilizando a função `value` definida por você no exercício 1, defina a função:

```
resistence ::  Resistor -> Float
```

que calcula o valor numérico do código de cores deste resistor. Como exemplo considere:

```
resistence (Resistor [Red, Brown, Yellow]) = 210000.0
```

3. Resistores são mais úteis quando unidos para formar circuitos. Considere o seguinte tipo de dados que representa circuitos:

```
data Circuit = Parallel Circuit Circuit
             | Series   Circuit Circuit
             | Single Resistor
             deriving (Eq, Ord)
```

Onde o construtor `Single` representa um circuito formado por apenas um resistor, o construtor `Series` representa dois circuitos de resistores em série e o construtor `Parallel` representa dois circuitos de resistores em paralelo. As figuras 1 e 2 apresentam circuitos em série e paralelo formados por três resistores `R1`, `R2` e `R3`. Supondo que `r1`, `r2` e `r3`

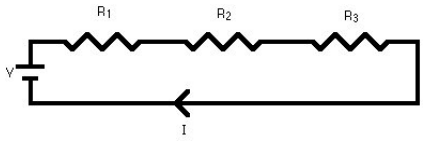

Figura 1: Circuito de resistores em série

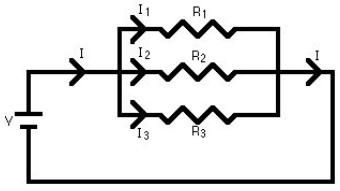

Figura 2: Circuito de resistores em paralelo

sejam valores do tipo `Resistor`, os circuitos apresentados nas figuras 1 e 2 podem ser representados pelos seguintes valores do tipo `Circuit`, respectivamente.

```
circuit1 = Series (Series (Single r1) (Single r2))
                  (Single r3)

circuit2 = Parallel (Parallel (Single r1) (Single r2))
                    (Single r3)
```

(a) Considerando, que `r1`, `r2`, `r3` e `r4` são valores do tipo resistor previamente definidos, defina o circuito formado por estes resistores que está representado na figura 3 utilizando um valor do tipo `Circuit`:

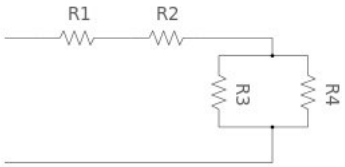

Figura 3: Circuito de resistores

(b) Um circuito representado por um valor do tipo `Circuit` é dito ser válido se e somente se cada um dos resistores que o compõe é válido, isto é, possui um código de cores correto. Utilizando a função `valid :: Resistor -> Bool`, implementada por você no item 2-a), desenvolva a função:

`validCircuit :: Circuit -> Bool`

que retorna verdadeiro se o valor do tipo `Circuit` fornecido como parâmetro for formado apenas por resistores válidos e falso caso contrário.

(c) É fato sabido que a resistência equivalente de um circuito formado por n resistores em série pode ser obtida pela seguinte fórmula:

$$r_{eq} = \sum_{i=1}^{n} r_i$$

onde r_i é o valor em Ohms da resistência do i-ésimo resistor deste circuito em série. Para circuitos formados por n resistores em paralelo temos que a resistência equivalente deste circuito é dada pela seguinte fórmula:

$$\frac{1}{r_{eq}} = \sum_{i=1}^{n} \frac{1}{r_i}$$

Com base no apresentado, implemente a função

`circuitResistence :: Circuit -> Float`

que calcula a resistência equivalente de um circuito utilizando as fórmulas anteriormente apresentadas e a função `resistence`,

definida no item 2-c). *Dica:* Para calcular o inverso de n, isto é, $\frac{1}{n}$, utilize a função `recip` (já definida na biblioteca da linguagem). Para qualquer valor numérico `n`, `recip n` $= \frac{1}{n}$.