Centro de Informática - UFPE
Primeiro Exercício Escolar — IF686 (2023.1) Turma: I5
Data: 01/08/2023 Horário: 8h-9h50
Docente: Márcio Lopes Cornélio
Discente:

Instruções para entrega: Entregar dois arquivos no Classrrom : (1) arquivo .hs, com todas as respostas (copiar e colar os códigos das respostas) e (2) um arquivo no formato pdf com todas as respostas. Escrever o nome na primeira linha dos arquivos.

1. O uso de um eletrodoméstico a é um par (a, t), em que t é o tempo (em horas) em que o eletrodoméstico é usado. Uma lista de uso é uma lista dos usos de cada eletrodoméstico durante ao longo de um dia. Isso é modelado por:

```
type Equipamento = String
type Uso = (Equipamento, Int)
type ListaUso = [Uso]
```

No GHCi, podemos ter as seguintes declarações

```
let e1 = ("maquina de lavar", 2)
let e2 = ("cafeteira", 1)
let e3 = ("lava loucas", 2)
let l = [e1, e2, e3, e1, e2]
```

(1,0) (a) Implemente a função inv :: ListaUso -> **Bool** que verifica se todos os tempos de uso que ocorrem na lista ListaUso são inteiros positivos.

(1,0) (b) Implemente a função duracaoDe :: Equipamento -> ListaUso -> **Int** que calcula o tempo acumulado de uso de um equipamento na lista de uso. Por exemplo, 'duracaoDe ``cafeteira" l' deveria resultar em 1.

(2,5) (c) Uma lista de uso é bem formada se satisfaz inv e o tempo acumulado de uso de qualquer eletrodoméstico é menor que 24 horas. Implemente a função que verifica a condição de boa formação.

```
*Main> bemFormada l
True
```

(1,0) (d) Implemente a função removerEqp :: Equipamento -> ListaUso -> ListaUso que retorna a lista obtida a partir da lista dada como argumento a partir da remoção do equipamento dado como argumento.

```
*Main> removerEqp "cafeteira" l
[("maquina de lavar",2),("lava loucas",2),
("maquina de lavar",2)]
```

Considere o preço de uso de eletrodomésticos modelado da seguinte forma:

```
type Preco   = Int
type Tarifa  = (Equipamento, Preco)
type Tarifas = [Tarifa]
```

No GHCi, por exemplo,

```
let tr1 = ("maquina de lavar", 20)
let tr2 = ("cafeteira", 3)
let tr3 = ("lava loucas", 15)
let trf = [tr1, tr2, tr3]
```

(2,5) (e) Implemente a função definidoEm :: ListaUso -> Tarifas -> **Bool** que é verdadeira se, e somente se, há um par em Tarifas para todo eletrodoméstico na lista de uso dada como argumento.

```
*Main> definidoEm trf l
True
```

(2,0) (f) Implemente a função precoDe :: ListaUso -> Tarifas -> Preco que resulta no preço total de usar os eletrodomésticos presentes na lista de uso dada como argumento. A função deve levantar uma exceção caso o eletrodoméstico não tenha tarifa definida na lista de tarifas.

```
*Main> precoDe l trf
116
*Main> precoDe l [("mixer", 4)]
********** Exception: Tarifa nao definida
```

2. A função vendas :: **Int** -> **Int** retorna a quantidade semanal de vendas de uma loja. As semanas são numeradas em uma sequência 0, 1, 2, ... Implemente a função zeroVendas que se comporta da seguinte maneira: dado um número n, que assumimos como não negativo, retorna o número de semanas na faixa 0, 1, ..., n em que a quantidade de itens vendidos foi 0 (zero). Implemente definições de zeroVendas

(1,0) (a) Usando compreensão de lista e a função **length**

(1,0) (b) Usando qualquer função padrão de Haskell, mas sem definir função recursiva, não usando **foldr** ou **foldl**, e sem usar compreensão de lista