Centro de Informática - UFPE
Primeiro Exercício Escolar — IF686 (2025.1)
Data: 17/06/2025 Horário: 08h-10h
Docente: Márcio Lopes Cornélio
Discente:

OBS: A entrega das respostas da prova deve ser feita por meio de dois arquivos., ambos depositados no Classroom. No início do corpo do arquivo deve haver a identificação do discente, com as respostas em seguida. Um arquivo deve ser entregue no formato texto (com extensão .hs), sendo nomeado de acordo com o padrão <nomeCompleto_1ee_plc_2025_1>. O outro arquivo (pdf) deve ser gerado a partir do arquivo com as respostas (código fonte).

(1,0) 1. Escreva uma função prswap :: [a] −> [a] que troque os elementos de uma lista em uma base de pares. Ou seja, o primeiro e o segundo elementos são trocados, o terceiro e o quarto são trocados, etc. Assuma que a lista tem um número par de elementos, mas pode ser possivelmente vazia.

```
*Main> prswap1 [1..10]
[2,1,4,3,6,5,8,7,10,9]
```

(1,0) 2. Escreva uma função fml ::[a] −> (a,a,a) que retorne uma tripla com o primeiro, o meio e o último elementos de uma lista. Suponha que a lista não seja vazia e tenha pelo menos um elemento. Não utilize a função de indexação (!!).

```
*Main> fml [1..8]
(1,5,8)
*Main> fml [1..7]
(1,4,7)
```

(1,0) 3. Escreva uma expressãona forma de uma compreensão de lista para calcular todos os números primos. Para determinar se um número i é primo, teste se nenhum número de 2 a i - 1 divide i. Você pode usar as funções **all** :: (a −> **Bool**) −> [a] −> **Bool**, onde **all** p xs resulta em **True** se, e somente se, p x resulta em **True** para todos os elementos x da lista xs, a função **not** :: **Bool** −> **Bool** e a função divide conforme definido abaixo.

```
divide i j = mod j i == 0
```

(1,0) 4. Implemente **map** em termos de **foldr**. Sua solução deve parecer com

```
map f list = foldr ...
```

(1,0) 5. (a) Defina a função recursiva merge :: **Ord** a =>[a] −> [a] −> [a] que une duas listas ordenadas e resulta em uma única lista ordenada.

(1,5) (b) Defina metade :: [a] −> ([a],[a]) que divide uma lista em duas metades com tamanhos que diferem no máximo de 1. Esta função será usada na função msort.

(1,5) (c) Usando merge, defina a função msort :: **Ord** a =>[a] −> [a] que implementa merge sort, no qual uma lista vazia ou lista com um elemento está ordenada e qualquer outra lista é ordenada por mesclar as duas listas que resultam de ordenar as duas metades da lista separadamente.

(0,5) 6. (a) Defina um tipo algébrico polimórfico Pilha que pode ser exibido. Uma pilha pode estar vazia ou, quando não vazia, manter valores em alguma estrutura de dados.

(1,5) (b) Defina as funções: push e pop. Trate a situação de pilha vazia como argumento da função pop.

```
push :: t -> Pilha t -> Pilha t
pop :: Pilha t -> (t, Pilha t)
```