1) Escreva a assinatura das seguintes funções:

```
dobro x = x * 2
primeiro (x, y) = x
maiorQue10 x = x > 10
somaLista xs = sum xs
```

2) Defina e implemente uma função chamada ehPar que recebe um número inteiro e retorna um booleano indicando se ele é par. Escreva a assinatura da função.

3) Implemente uma função quadrado que recebe um número x e retorna x ao quadrado. Escreva a assinatura da função.

4) Implemente uma função recursiva fatorial :: Integer -> Integer que calcula o fatorial de um número.

5) Implemente uma função recursiva somaN :: Integer -> Integer que retorna a soma de todos os números de 1 até n.

6) Implemente uma função recursiva fibonacci :: Integer -> Integer que calcula o n-ésimo termo da sequência de Fibonacci.

7) Implemente uma função contaElementos :: [a] -> Int que conta quantos elementos há em uma lista. Não use length.

8) Implemente uma função reverter :: [a] -> [a] que inverte uma lista. Não use reverse.

9) Defina um novo tipo chamado DiaSemana representando os dias da semana.

10) Implemente a função ehFimDeSemana :: DiaSemana -> Bool que retorna True se for sábado ou domingo e False caso contrário.

11) Implemente a função produtoLista :: [Integer] -> Integer que calcula o produto de todos os elementos de uma lista. Não use product.

12) Implemente a função recursiva elementoN :: [a] -> Int -> a que retorna o n-ésimo elemento de uma lista (o primeiro elemento tem índice 0). Caso o índice seja inválido, retorne um erro.

13) Implemente a função somaImpares :: [Integer] -> Integer que recebe uma lista de inteiros e retorna a soma dos números ímpares.

14) Implemente a função contaOcorrencias :: Eq a => a -> [a] -> Int que conta quantas vezes um determinado elemento aparece em uma lista.

15) Implemente a função removeElemento :: Eq a => a -> [a] -> [a] que remove todas as ocorrências de um elemento da lista.

16) Implemente a função duplicarElementos :: [a] -> [a] que duplica cada elemento da lista.

17) Implemente a função intercalar :: [a] -> [a] -> [a] que intercala os elementos de duas listas. Se uma lista for maior que a outra, os elementos extras devem ser adicionados ao final.

18) Implemente a função removerDuplicatas :: Eq a => [a] -> [a] que remove elementos repetidos consecutivos de uma lista.