

UNIVERSIDADE FEDERAL DE SANTA MARIA

Curso: Ciência da Computação

Disciplina: Paradigmas de Programação

Professora: Juliana Kaizer Vizzotto

0.1 Exercícios

1. Implemente a função:

```
isPalindrome :: [Int] -> Bool
*Main> isPalindrome [1,2,3]
False
*Main> isPalindrome "madamimadam"
True
*Main> isPalindrome [1,2,4,8,16,8,4,2,1]
True
```

que verifica se uma lista é palindromo ou não.

2. Implemente a função :

```
compress :: [t] :: [t]
*Main> compress ["a","a","a","a","b","c","c","a","a","d","e","e","e","e"]
["a","b","c","a","d","e"]
```

que elimina duplicadas consecutivas em uma lista.

3. Implemente a função:

```
pack :: [t] -> [[t]]
*Main> pack ['a','a','a','a','b','c','c','a','a','d','e','e','e']
["aaaa","b","cc","aa","d","eee"]
```

que empacota os elementos duplicados consecutivos em sublistas.

4. Implemente a função

```
encode :: [t] -> [(Int,t)]
encode "aaaabccaadeeee"
[(4,'a'),(1,'b'),(2,'c'),(2,'a'),(1,'d'),(4,'e')]
```

que especifica o método de compressão de dados baseado no tamanho da sequência repetida. Neste método os elementos duplicados consecutivos são codificados como duplas (N, E) , onde N é o número de duplicadas do elemento E .

5. Implemente a função

```
decode :: [(Int,t)] -> [t]
```

a qual, dada uma lista codificada como no exercício anterior, gera a lista original.

6. Implemente a função

```
rotate :: [t] -> Int -> [t]
*Main> rotate ['a','b','c','d','e','f','g','h'] 3
"defghabc"
*Main> rotate ['a','b','c','d','e','f','g','h'] (-2)
"ghabcdef"
```

a qual rota uma lista N elementos para a esquerda.

7. Implemente a função:

```
primesI :: Int -> Int -> [Int]
*Main> primesI 10 20
[11,13,17,19]
```

que dado dois números que representam um intervalo de inteiros, i.e., são informados o menor e o maior - inclusive - números do intervalo, retorna uma lista com todos os números primos nesse intervalo.

8. Um código cinza de n -bits é uma sequência de n bits construída de acordo com certas regras. Por exemplo,

```
n=1 : C(1) = ["0","1"]
n=2 : C(2) = ["00","01","10","11"]
n=3 : C(3) = ["000","001","011","010","110","111","101","100"]
```

Implemente a função:

```
gray :: Int -> [String]
*Main> gray 3
["000","001","011","010","110","111","101","100"]
```