Universidade Federal do Piauí

***Centro de Ciências da Natureza***

1. Departamento de Computação

***Disciplina: Linguagens Funcionais***

***Professor: Francisco Vieira de Souza***

***2ª. Lista de exercícios (para o dia 1º de julho de 2014)***

1. Defina, em Haskell uma função "f" que, dadas uma lista "i" de inteiros e uma outra lista "l" qualquer, retorne uma nova lista constituída por "l" seguida de seus elementos que têm posição indicada em "i",conforme o exemplo a seguir:

f [2,1,4] ["a", "b", "c", "d"] = ["a", "b", "c", "d", "d", "a", "b"]

2. Defina, em Haskell, uma função que calcula o Determinante de uma matriz quadrada de ordem “n".

3. Resolva o problema das oito rainhas, usando Haskell.

4. Defina, em Haskell, uma função “f" que, dada uma lista “l", construa duas outras listas “l1" e “l2" de

forma que “l1" contenha os elementos de “l" de posição ímpar e “l2" contenha os elementos de “l" de

posição par, preservando a posição relativa dos elementos, conforme os exemplos a seguir:

f [a, b, c, d] = [[a, c], [b, d]]

f [a, b, c, d, e] = [[a, c, e],[b, d]]

5. Um pequeno visor de cristal liquido (LCD) contém uma matriz 5x3 que pode mostrar um número, como 9 ou 5. Por exemplo:

\*\*\* \*\*\*

\* \* \*

\*\*\* \*\*\*

\* \*

\* \*\*\*

O formato de cada número é definido por uma lista de inteiros que indica quantos '\*' se repetem,

seguidos de quantos brancos se repetem, até o final da matriz 5x3, começando da primeira linha até a

última:

nove,cinco,um,dois,tres,quatro,seis,sete,oito,zer :: [Int]

nove = [4,1,4,2,1,2,1]

cinco = [4,2,3,2,4]

um = [0,2,1,2,1,2,1,2,1,2,1]

dois = [3,2,5,2,3]

tres = [3,2,4,2,4]

quatro = [1,1,2,1,4,2,1,2,1]

seis = [4,2,4,1,4]

sete = [3,2,1,2,1,2,1,2,1]

oito = [4,1,5,1,4]

zero = [4,1,2,1,2,1,4]

indicando que o número nove é composto por 4 '\*'s (três na primeira linha e um na segunda), seguido de 1 espaço, mais 4 '\*'s, 2 espaços, 1 '\*', 2 espaços e 1 '\*'). Faça funções para:

1. Dado o formato do número (lista de inteiros) gerar a String correspondente de '\*' e espaços.

toString :: [Int] -> String

toString nove ==> "\*\*\*\* \*\*\*\* \* \*"

1. Faça uma função que transforme a String de '\*'s e espaços em uma lista de Strings, cada uma representando uma linha do LCD:

type Linha = String

toLinhas :: String -> [Linha]

toLinhas "\*\*\*\* \*\*\*\* \* \*" ==> ["\*\*\*","\* \*","\*\*\*"," \*"," \*"]

1. Faça uma função que pegue uma lista de Strings e a transforme em uma única String com '\n' entre cada uma delas:

showLinhas :: [Linha] -> String

showLinhas ["\*\*\*","\* \*","\*\*\*"," \*"," \*"] = "\*\*\*\n\* \*\n\*\*\*\n \*\n \*"

1. Faça uma função que pegue duas listas de linhas e transforme-as em uma única lista de linhas, onde as linhas originais se tornam uma única, com um espaço entre elas:

juntaLinhas :: [Linha] -> [Linha] -> [Linha]

juntaLinhas ["\*\*\*","\* \*","\*\*\*"," \*"," \*"] ["\*\*\*","\* \*","\*\*\*"," \*"," \*"] =

["\*\*\* \*\*\*","\* \* \* \*","\*\*\* \*\*\*"," \* \*"," \* \*"]

1. Faça uma função que, dado um inteiro, o imprima usando '\*'s, espaços e '\n's. A função tem que funcionar para inteiros de 2 e 3 dígitos.

tolcd :: Int -> String

**Dica**: use as funções div e mod de inteiros, a função (!!), e a lista números, dada abaixo:

numeros :: [[Int]]

numeros = [zero,um,dois,tres,quatro,cinco,seis,sete,oito,nove]

1. Faça uma função que, dada uma String de '\*'s e espaços, retorne a representação da String como [Int] no formato usado no LCD (i.e. a função inversa de toString).

toCompact :: String -> [Int]

toCompact "\*\*\*\* \*\*\*\* \* \*" = [4,1,4,2,1,2,1]

6. Defina, em Haskell, uma função que aplicada a uma lista “l” e a um inteiro “n” retorne todas as sublistas de “l” com comprimento maior ou igual a “n”.

1. Projete, em Haskell, uma função substr :: String -> String -> String -> String que substitua uma

substring1 por uma substring2, caso sbstring1 ocorra em uma string3. Por exemplo,

substr “adora” “ama” “Ela adora o mar” = “Ela ama o mar”

Caso a substr1 não ocorra na string3, o resultado deve ser a string3.

1. Modifique o banco de dados (Database = [(Pessoa, Livro)]) da biblioteca de livros e as funções

de acesso:

l\_emprestado\_a :: Database -> Pessoa -> [Livro]

devedores :: Database -> Livro -> [Pessoa]

esta\_emprestado :: Database -> Livro -> Bool

num\_emprestados :: Database -> Pessoa -> Int

toma\_emprestado :: Database -> Pessoa -> LIvro -> Database e

devolve\_livro :: Database -> Pessoa -> Livro -> Database

de forma que:

1. exista um número máximo de livros que uma mesma pessoa possa tomar emprestado,
2. exista uma lista de palavras chaves associadas a cada livro de forma que os livros possam ser encontrados através das palavras chaves associadas a eles e
3. datas sejam associadas com empréstimos de forma que livros atrasados possam ser encontrados e a lista de livros emprestados a uma determinada pessoa possa ser ordenada por datas
4. Usando expressões ZF, defina as funções subListas e subsequências, ambas do tipo [t] -> [[t]] que retornem todas as sublistas e todas as subsequências de uma lista. Uma sublista de uma lista é obtida omitindo-se alguns dos elementos de uma lista e uma sequência é um bloco contínuo de uma lista. Por exemplo, [2,4] e [3,4] são sublistas de [2,3,4], mas apenas [3,4] é uma subsequência desta mesma lista.