1ª Lista de Exercícios de Paradigmas de Linguagens Computacionais
Professor: Fernando Castor
Monitores:

Ciência da computação: Erick Lucena Palmeira Silva (elps), João Gabriel Santiago Maurício de Abreu (jgsma), Marcos Paulo Barros Barreto (mpbb) , Tulio Paulo Lages da Silva (tpls), Victor Félix Pimenta (vfp)

CIn-UFPE – 2014.1
Disponível desde: 16/04/2014
Entrega: 30/04/2014

A lista deverá ser respondida **em dupla**. A falha em entregar a lista até a data estipulada implicará na perda de 0,25 ponto na **média** da disciplina para os membros da dupla. Considera-se que uma lista na qual **menos que 11** das respostas estão corretas não foi entregue. A entrega da lista com **pelo menos 17** das questões corretamente respondidas implica em um acréscimo de 0,125 ponto na média da disciplina para os membros da dupla. Se **qualquer situação de cópia de respostas** for identificada, os membros **de todas as duplas envolvidas** perderão 0,5 ponto na média da disciplina. O mesmo vale para respostas obtidas a partir da Internet. As respostas deverão ser entregues **exclusivamente em formato texto ASCII** (nada de .pdf, .doc, .docx ou .odt) e deverão ser enviadas para o monitor responsável por sua dupla, **sem** cópia para o professor. Devem ser organizadas em arquivos separados, um por questão, entregues em um único arquivo compactado, ou seja, um único arquivo .zip contendo as respostas para todas as questões. Um membro de cada dupla deve ir até a página da monitoria correspondente de CC (https://sites.google.com/a/cin.ufpe.br/monitoria-plc/) e registrar os nomes e logins dos membros da sua dupla sob o nome de um monitor. A escolha do monitor deve seguir uma política *round-robin* de modo a balancear a carga de duplas entre os monitores da maneira mais equitativa possível. A não-observância desta política pode resultar na transferência da dupla para outro monitor.

1) Implemente uma função que calcula posições no Triângulo de Pascal (http://pt.wikipedia.org/wiki/Tri%C3%A2ngulo_de_Pascal) . Ela deve receber dois números, os números da linha e da coluna, e calcular o elemento correspondente no triângulo.
Exemplo:
Prelude> pascal 5 3
10
Prelude> pascal 3 2
Prelude> pascal 6 4
15

2) Implemente uma função que recebe um ano e informa se ele é bissexto.
Lembrando que as regras para o cálculo de anos bissextos são:

1. De 4 em 4 anos é ano bissexto.
2. De 100 em 100 anos não é ano bissexto.
3. De 400 em 400 anos é ano bissexto.
4. Prevalecem as últimas regras sobre as primeiras.

Exemplo:
Prelude> bissexto 2000
True
Prelude> bissexto 1997
False
Prelude> bissexto 2008
True
Prelude> bissexto 2014
True

3) A partir da tabela pitagórica é possível mapear nomes em números de 1 a 9 da seguinte maneira:
I) Extraia da tabela o número correspondente a cada letra do nome e vá concatenando, ficando com um número final com a quantidade de dígitos igual a quantidade de letras da palavra.
(ex: “paradigma” -> 719149741)
II) Some todos os dígitos do número (ex: 231->6)
III) Se a soma der um número de 2 ou mais dígitos (ex: 187->16), repita o passo 2.

IV) Quando a soma der um número com apenas um dígito esse é o resultado
(ex: “paradigma” -> 719149741
719149741 -> 43
43 -> 7. Resultado é 7)

Tabela:
1 2 3 4 5 6 7 8 9
A B C D E F G H I
J K L M N O P Q R
S T U V W X Y Z

Faça uma função que dado um nome retorne o seu valor numérico segundo a tabela pitagórica. (Utilize obrigatoriamente a função map. Ver se é necessário)

4) A Cifra de César (Caesar cipher) é uma técnica de criptografia antiga (e muito simples). A criptografia consiste em fazer uma variação de cada letra da String de entrada, dando um shift para a direita com o valor da chave de criptografia fornecida.
Por exemplo, a frase “**Eu amo PLC”**, com chave **3** se tornaria: “Hx dpr SOF”
Vendo que essa criptografia é muito simples de ser quebrada, um aluno de PLC com tempo de sobra criou a **Caesar Salad.**
Nessa criptografia se usa uma lista de chaves, em invés de apenas uma. Assim, o primeiro caractere da String de entrada deve sofrer uma varição definida pela primeira chave, o segundo caractere pela segunda, e assim sucessivamente. A lista de chaves deve ser usada de maneira cíclica, caso seja menor do que a String de entrada.
Sabendo disso, implemente a função: **caesarSalad :: String -> [Int] -> String**

5) Implemente a função **filtroMediana :: [[int]]->Int -> [[int]],** que recebe uma imagem em formato de matriz [[int]], onde cada pixel é representado por um valor entre 0 – 255 (RGB) e devolve a mesma imagem após receber efeito de filtro de mediana. O tamanho da vizinhança deve ser o segundo argumento da função (n). Dessa forma, para cada posição da matriz, você deve pegar uma vizinhança de tamanho nxn e substituir o número da posição atual, pela mediana dos números presentes na vizinhança.
http://www.ime.usp.br/~reverbel/mac110-BCC-07/eps/ep4.pdf

6) Implemente uma função que devolva o n-ésimo número da sequencia look-and-say, defina da seguinte maneira:

- 1 é o primeiro número da sequencia.
- 1 é lido como "um 1", logo o segundo valor da sequencia é 11.
- 11 é lido como "dois 1s", logo o terceiro valor é 21.
- 21 é lido como "um 2, então um 1", logo temos 1211.
- 1211 é lido como "um 1, então um 2, então dois 1s", logo temos 111221.
- 111221 é lido como "três 1s, então dois 2s, então um 1", logo temos 312211.

1, 11, 21, 1211, 111221, 312211, 13112221, 1113213211...
http://en.wikipedia.org/wiki/Look-and-say_sequence

7) O Código Morse é um sistema de representação de letras, números e sinais de pontuação através de um sinal codificado enviado intermitentemente. Seu uso é muito importante e existem várias formas de se enviar informação de acordo com esse sistema. Para entender a mensagem recebida, é necessário que a pessoa converta os símbolos contidos nela em letras, números ou sinais. Você ajudará essa pessoa a construir o seu decodificador de Código Morse, implementando a função
morseTranslator :: [Char] -> [Char].

OBS1: Separações entre letras são representadas por um espaço (' '), enquanto que separações entre palavras são representadas por ' / '
OBS2: Para mais informações sobre código morse:
http://en.wikipedia.org/wiki/Morse_code
http://morsecode.scphillips.com/jtranslator.html

Exemplos:

Prelude> morseTranslator "-... .- -.. / .--- --- -.- . / - .. -- ."
"BAD JOKE TIME"
Prelude> morseTranslator ".. / .-.. --- ...- . / .--. .-. . ... -. --. / ..-. ..... .-.-.- / .. - ... / ... --- / .-. . ..-. .-. . ... -. --."
"I LOVE PRESSING F5. ITS SO REFRESHING"
Prelude> morseTranslator ".- / ... --.- .-.. / --.- ..- . .-. -.-- / --. --- . ... / .. -. - --- / .- / -... .- .-. --..-- / .-- .- .-.. -.- ... / ..- .--. / - --- / - .-- --- / - .- -... .-.. . ... / .- -. -.. / .- ... -.- ... --..-- / .-..-. -.-. .- -. / .. / .--- --- .. -. / -.-- --- ..- ..--.. .-..-."
"A SQL QUERY GOES INTO A BAR, WALKS UP TO TWO TABLES AND ASKS, "CAN I JOIN YOU?""
Prelude> morseTranslator ".... --- .-- / -- .- -. -.-- / .--. .-. --- --. .-. .- -- -- . .-. ... / -.. --- . ... / .. - / - .- -.- . / - --- / -.-. .... .- -. --. . / .- / .-.. .. --. .... - / -... ..- .-.. -... ..--.. / -. --- -. . --..-- / - .... .- - ... / .- / .... .- .-. -.. .-- .- .-. . / .--. .-. --- -... .-.. . --"
"HOW MANY PROGRAMMERS DOES IT TAKE TO CHANGE A LIGHT BULB? NONE, THATS A HARDWARE PROBLEM"
Prelude> morseTranslator "- --- / ..- -. -.. . .-. ... - .- -. -.. / .-- .... .- - / .-. . -.-. ..- .-. ... .. --- -. / .. ... / -.-- --- ..- / -- ..- ... - / ..-. .. .-. ... - / ..- -. -.. . .-. ... - .- -. -.. / .-. . -.-. ..- .-. ... .. --- -."
"TO UNDERSTAND WHAT RECURSION IS YOU MUST FIRST UNDERSTAND RECURSION"
Prelude> morseTranslator ".- / --. --- --- -.. / .--. ..- -. / .. ... / .. - ... / --- .-- -. / .-. . .-- --- .-. -.."
"A GOOD PUN IS ITS OWN REWORD"

8) Dada uma lista de números, devolva a maior lista de números contidos na lista original que possua apenas primos entre si. Se for possível, (obrigatoriamente) use compreensão de listas, se não for possível, explique porque (comentários no código) e implemente sem.

9) Implemente uma função capaz de, dado um par de funções, executar um "fold matricial" comprimindo os elementos de uma determinada linha usando uma das funções e os resultados usando a outra.
*Main> foldMatrix (+) (*) [[1,2,3],[1,2,3]]
36
*Main> foldMatrix (*) (+) [[1,2,3],[1,2,3]]
12

10) Implemente suas próprias versões das seguintes funções (obviamente não é permitido usar as funções originais):
a)map
b)filter
c)foldr

11) Defina a função **testeListas::[int]->[int]** que recebe uma lista de inteiros e devolve: o primeiro número da lista original, o segundo número da lista original e todos os números de posição i **na lista original**, que forem maiores do que o número da posição i-2 **da lista original**. Se for possível, (obrigatoriamente) use compreensão de listas, se não for possível, explique porque (comentários no código) e implemente sem.

Prelude> testeListas [1,2,0,4,1,3,2]
[1,2,4,1,2]
Prelude> testeListas [3,4,2,3,5,7,4,6,2,5]
[3,4,5,7]

12) Crie uma função **maioresPalindromos :: [Char] -> [Char]** que deve receber uma String e retornar uma lista de Strings contendo o(s) maior(es) palíndromo(s) (http://pt.wikipedia.org/wiki/Pal%C3%ADndromo) presente(s) na palavra. Por exemplo, o maior palíndromo da palavra "arara" é ela mesma, mas ela possui outros palíndromos dentro de si, como "ara" ou "rar". Caso a palavra fosse “rara”, os maiores palíndromos seriam “rar” e “ara”.

13) Crie uma typeclass chamada PrimeirosPrimos com as funções **primeFib** e **primeFat** que devolvam os n primeiros primos da sequencia de fibonacci e fatorial respectivamente. Você deve criar uma instancia da classe PrimeirosPrimos para o tipo Integer. Utilize obrigatoriamente as funções map e filter.
Exemplo:

Prelude> primeFib 2
[2,3]
Prelude> primeFib 5
[2,3,5,13,89]
Prelude> primeFib 8
[2,3,5,13,89,233,1597,28657]

14) Crie uma função em Haskell que simule um AFD (Automato Finito Determinístico). A função deve receber uma lista de inteiros representando a String de entrada (cujo alfabeto é {0, 1}), uma lista de transições, uma lista de estados e um valor que indique qual o estado inicial do AFD. Ao fim de sua execução, a função deve retornar um Bool indicando se a String de entrada foi aceita ou não pelo AFD.

Escolha a maneira que achar mais adequada para representar os componentes do AFD. Justifique essas escolhas em forma de comentário no seu código.

15) Implemente um filtro para eliminar CPF's inválidos, i.e., com dígitos verificadores incorretos. A lista recebida será composta de Strings, assim como a lista final deverá ser. Considere o cálculo do dígito verificador de CPF listado aqui:
http://www.jalucrei.com.br/calculo_dv_cpf_cgc.htm

*Main> cpfilter ["010.222.010-77", "222.333.666-38", "555.001.002-04"]
["010.222.010-77","222.333.666-38"]

16) O prefeito de Seilacomoseescreve decidiu, juntamente ao Conselho Municipal de Ortografia, adotar um plano de 5 anos para trocar a língua oficial da cidade, por uma língua mais simples que o português: o Br. Inicialmente, todos os acentos são abolidos e o hífen é trocado por espaço.

exceção -> excecao você -> voce guarda-chuva -> guarda chuva

As regras do plano de 5 anos são:

1- No ano 1, a letra ‘c’ seria trocada por ‘s’ antes de ‘a’, ‘e’ e ‘i’ e por ‘k’ antes de ‘o’ e ‘u’. Outra medida seria trocar ‘ss’ seria por um único ‘s’.

como -> komo
formacao -> formasao
assar -> asar

2- No ano 2, ‘qu’ também seria trocado por ‘k’.

que -> ke
queijo -> keijo

3- No ano 3, ‘h’ deixaria de existir, ‘ch’ viraria ‘x’, ‘nh’ viraria ‘ni’ e ‘lh’ viraria ‘li’.

hora -> ora
chave -> xave
manha -> mania
molho -> molio

4- No ano 4, ‘j’ seria trocado por ‘x’ e apenas utilizado antes de ‘a’ e ‘o’. Antes de ‘e’, ‘i’ e ‘u’, seria utilizado ‘g’.

jardim -> xardim
arranjo -> arranxo
jilo -> gilo
jurar -> gurar

5- No ano 5, não existiriam vogais.

vogais -> vgs

Você, aluno de PLC, foi contratado pelo prefeito de Seilacomoseescreve para criar uma função em Haskell que recebe converte uma string com palavras do português atual para o Br.

Para isso, sua função deverá receber um inteiro que simboliza em que estágio do plano de 5 anos o Br está, e uma do português atual (assuma que já não existem acentos nem hífens) e traduzirá para o Br seguindo as regras estabelecidas até o ano dado.

17)Faça uma função que receba duas listas ordenadas, L1 de tamanho m e L2 de tamanho n e devolva uma terceira lista com os integrantes de L1 e L2 também ordenada, com MAX(m,n) comparações.

18) R*b*rts*n N*v*l*nd* é um estudante de Ciência da Computação que está fazendo intercâmbio no Reino Unido, na terra da Rainha. Na festa de Saint Patrick, ele acaba se encontrando com 8 ex-namoradas, mas como eram todas recalcadas, ele não queria que elas se encontrassem. Como era amigo do garçom, ele pediu pra que o garçom arrumasse as mesas de forma com que uma não ficasse no campo de visão da outra, assim ele poderia falar com elas, uma por uma. Há 64 mesas, dispostas em uma formação de 8x8, e devido aos enfeites da festa, uma mesa só está no campo de visão de pessoas em outras mesas que estão ou na mesma horizontal, na mesma vertical ou em uma das diagonais. Com isso, crie uma função **arrumarMesas :: Int -> [Int]** que recebe um inteiro n e deve retornar a n-ésima configuração possível (se existir) de onde a mesa de cada garota deve estar. Por exemplo, a lista resultante [4,2,7,3,6,8,5,1] significa que a primeira garota está na quarta coluna da primeira linha, a segunda está na segunda coluna da segunda linha e assim por diante.

19) Sudoku é um quebra-cabeça montado sobre um tabuleiro de números 9x9, que consequentemente se divide em 9 quadrados 3x3. Dessa forma 3 regras devem ser seguidas. (existem variantes de tamanho, mas devem ser desconsideradas)

1) Nenhum número pode se repetir em uma linha
2) Nenhum número pode se repetir em uma coluna
3) Nenhum número pode se repetir em um quadrado 3x3

Para melhor visualização: http://pt.wikipedia.org/wiki/Sudoku

Escolha a maneira que preferir para representar o tabuleiro sudoku, contanto que não haja redundância (guardar a mesma casa do tabuleiro duas ou mais vezes). Descreva a maneira escolhida em comentários no código.
Faça uma função que receba um tabuleiro (na representação definida por vocês) totalmente preenchido e verifique se ele está correto (se segue as 3 regras do sudoku)
Por fim, coloque um exemplo da sua função com uma entrada funcionando comentado no código.