Teste 2 (Individual ou em Dupla)

INF-0612 – Análise de Dados

O objetivo deste teste é exercitar os conceitos vistos até agora no curso. São quatro problemas a serem resolvidos:

(3 pontos) Agrupamento

Implemente uma função com a assinatura `groupsum(df, colgroup, colsum)`. Essa função receberá um data frame `df` e o nome de duas colunas `colgroup` e `colsum`. A função deve agrupar as linhas seguindo o valor de `colgroup` e acumular o valor da coluna `colsum`. A saída da função deve ser um novo data frame com apenas as colunas `colgroup` e `colsum`. Note que a ordem das linhas da saída é irrelevante.

```
#considere o data frame 'chuvas' abaixo:
> chuvas
     cidade chuva dia
1  Campinas  0.15   1
2   Vinhedo  0.02   1
3  Campinas  0.01   2
4   Limeira  0.13   2
5  Campinas  0.12   3
6   Vinhedo  2.19   3
7  Campinas  1.11   4
8   Vinhedo  0.76   4
9   Limeira  2.98   5
10 Campinas  0.45   5
> groupsum(chuvas, "cidade", "chuva")
    cidade chuva
1 Campinas  1.84
2  Limeira  3.11
3  Vinhedo  2.97

#utilizando o dataframe customer_churn do teste anterior
> customer_churn = read.table("customer_churn.csv", sep=",", header = TRUE,
    stringsAsFactors= FALSE)
> groupsum(customer_churn, "Contract", "MonthlyCharges")
        Contract MonthlyCharges
1 Month-to-month       273629.5
2       One year       102111.6
3       Two year       110560.4
```

(3 pontos) Binário para Decimal

Implemente uma função com a assinatura `binToDec(...)`. Essa função receberá como parâmetros uma quantidade indefinida de vetores binários. Cada vetor de binários simboliza a representação binária de um número inteiro. Por exemplo, o vetor c(1, 0, 1) representa o número $5 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$. A saída esperada para essa função deve ser um vetor de inteiros onde a primeira posição desse vetor é representação decimal do primeiro vetor de binários dado como entrada, a segunda posição do vetor é a representação decimal do segundo vetor de binários dado como entrada, e assim por diante.

Exemplos de saídas esperadas para a função:

```
> binToDec(c(1, 0))
[1] 2
```

```
> binToDec(c(0, 0, 1), c(1, 1))
[1] 1 3

> binToDec(rep(1, 3), rep(0, 2), rep(c(1,0), 2))
[1] 7 0 10
```

(3 pontos) Ocorrência de Palavras

A repetição de uma mesma palavra em um texto não é uma boa prática de escrita. Imagine que você é um(a) funcionário(a) de uma grande empresa de publicidade. Atualmente, na sua empresa, o processo que contabiliza a ocorrência de um determinada palavra em um texto é feito de forma manual por funcionários. Seu gerente quer automatizar esse processo, e deixou você encarregado de criar um mecanismo para tal. Implemente uma função com a assinatura `wordCount(word, text)`, que recebe dois parâmetros, ambos do tipo `character` (String), o primero deles (`word`) representa a palavra e o segundo (`text`) representa o texto. A saída dessa função deve ser um número inteiro que indique a quantidade de ocorrências da palavra no texto.

Observações:

- Você deve implementar a função de maneira *case insensitive*, ou seja, uma mesma palavra pode estar escrita em caixa alta ou caixa baixa, mas devem ser consideradas iguais.
- Você pode assumir que o texto e a palavra não terão acentos.
- Você pode assumir que o texto só apresentará as seguintes pontuações: "`.`", "`,`", "`!`" e "`?`".

Exemplos de saídas esperadas para a função:

```
> text <- "O rAto roeu a roupa do Rei de Roma! RainhA raivosa rasgou o resto."
> wordCount("rato", text)
[1] 1
> wordCount("roma", text)
[1] 1

> text <- "A vaca malHada foi molhADA por outra VACA, MOLhada e MALhaDa."
> wordCount("outra", text)
[1] 1
> wordCount("vaca", text)
[1] 2
> wordCount("malhada", text)
[1] 2

> text <- "Se a liga me ligasse, eu tambem ligava a liga. Mas a liga nao me
    liga, eu tambem nao ligo a liga."
> wordCount("liga", text)
[1] 5
> wordCount("ligasse", text)
[1] 1
```

(1 ponto) Ordenação de Panquecas

Você deve escrever uma função que ordene uma pilha de panquecas de forma que a maior panqueca fique na base e a menor panqueca no topo. As panquecas são representadas por um vetor, sendo que o primeiro elemento indica a panqueca na base da pilha (e o último, a paquenca no topo da pilha). Cada panqueca é representada por um número que indica o seu diâmetro. Todas as panquecas da pilha possuem tamanhos diferentes com valores de 1 a n, onde n é quantidade de panquecas na pilha.

A ordenação da pilha de panquecas só pode ser feita por "giros". Um giro consiste em inserir uma espátula entre duas panquecas (ou entre a base e a primeira panqueca da pilha) e virar todas as panquecas que estão acima da espátula. A sua primeira tarefa é criar a função auxiliar `giro(panquecas, i)` que realiza o giro das panquecas,

considerando que a espátula é inserida abaixo da panqueca da posição i, ou seja, a operação de giro inverte os elementos das posições `i` até `length(panquecas)`.

```
> panquecas <- c(3,4,1,2); panquecas
[1] 3 4 1 2
> giro(panquecas, 2)
[1] 3 2 1 4
```

A sua segunda tarefa é implementar a função `ordenar(panquecas)` usando apenas giros. A cada giro realizado, você deve imprimir o estado da pilha de panquecas. Sua ordenação para qualquer pilha com n panquecas não pode conter mais de $2n$ giros.

```
> panquecas
[1] 3 4 1 2
> panquecas <- ordenar(panquecas)
[1] 3 2 1 4
[1] 4 1 2 3
[1] 4 3 2 1
> panquecas
[1] 4 3 2 1

> pilha <- c(2, 10, 4, 8, 6, 9, 1, 5, 7, 3); pilha
 [1]  2 10  4  8  6  9  1  5  7  3
> pilha <- ordenar(pilha)
 [1]  2  3  7  5  1  9  6  8  4 10
 [1] 10  4  8  6  9  1  5  7  3  2
 [1] 10  4  8  6  2  3  7  5  1  9
 [1] 10  9  1  5  7  3  2  6  8  4
 [1] 10  9  1  5  7  3  2  6  4  8
 [1] 10  9  8  4  6  2  3  7  5  1
 [1] 10  9  8  4  6  2  3  1  5  7
 [1] 10  9  8  7  5  1  3  2  6  4
 [1] 10  9  8  7  5  1  3  2  4  6
 [1] 10  9  8  7  6  4  2  3  1  5
 [1] 10  9  8  7  6  5  1  3  2  4
 [1] 10  9  8  7  6  5  4  2  3  1
 [1] 10  9  8  7  6  5  4  2  1  3
 [1] 10  9  8  7  6  5  4  3  1  2
 [1] 10  9  8  7  6  5  4  3  2  1
> pilha
 [1] 10  9  8  7  6  5  4  3  2  1
```

Considerações Finais

- A utilização de qualquer função não vista em sala de aula, ou seja, que não encontra-se nos slides fornecidos no Moodle, acarretará em um desconto de 50% da nota daquele item.
- Para testes realizados em dupla, apenas um membro da dupla deve enviar a solução. Os nomes dos membros devem constar no cabeçalho de cada arquivo "`.R`".
- Envie os arquivos pelo sistema Moodle, clicando no link "`Teste 2`" da Seção "`Avaliações`". Clique em "`Adicionar tarefa`", anexe os arquivos e, por fim, clique em "`Salvar mudanças`". Você voltará para a tela da atividade e deverá constar o status "`Enviado para avaliação`". A qualquer momento, antes do prazo final de submissão, você pode alterar sua submissão clicando em "`Editar envio`".

Prazo de entrega: 15 de março de 2019 (Sexta-Feira), até às 23h55.

Forma de entrega: via sistema Moodle:

- `https://moodle.lab.ic.unicamp.br/moodle/course/view.php?id=336`

Pontuação: Este teste será pontuado de 0 a 10, e corresponderá 30% da nota final.