NOVA IMS – Universidade Nova de Lisboa
Computação em Estatística e Gestão de Informação
2º Semestre 2015/16

Exame 1ª Época – Versão A – 11/07/2016

Curso: _____ Número: _____ Nome: _____

Leia, por favor, com atenção:

1. Este enunciado corresponde à segunda parte do exame (**R**).
2. Cada parte está cotada para 10 valores e tem a nota mínima de 4,0 valores.
3. Este exame deverá ser realizado no enunciado, sem acesso a um computador.
4. Poderá consultar o formulário dado em anexo ao exame.
5. É proibido o uso de qualquer outro material de apoio (livros, apontamentos, telemóvel), assim como a troca de qualquer informação com os colegas.
6. A entrega do exame e a saída da sala só são possíveis no final do exame.
7. As respostas às questões deverão ser dadas, exclusivamente, na folha do enunciado, no espaço reservado para tal. Estas respostas deverão ser apenas código em **R**.
8. Não é necessário escrever o resultado do código, mas apenas o código em si.
9. O não cumprimento de alguma das regras conduzirá à anulação do exame.
10. A duração do exame, considerando ambas as partes, é de **2 horas**.

1. Usando estruturas cíclicas (iterativas), escreva o código que permite calcular o resultado das seguintes expressões.

   (a) [0.5 val.]

$$\sum_{i=10}^{100} \left(i^2 + 4i^2\right) \tag{1}$$

Solução:

```
ua = c()
for (i in 10:100) {
        ua=c(ua, i ^ 2 + 4 * i ^ 2)
}
sum(ua)
```

(b) [0.5 val.]

$$\sum_{i=1}^{10} \frac{\sqrt{i}}{i+1} \tag{2}$$

Solução:

```
ub = c()
for (i in 1:10) {
        ub=c(ub, sqrt(i) / (i + 1))
}
sum(ub)
```

2. Sem usar estruturas cíclicas, escreva o código que permite chegar ao mesmo resultado das alíneas anteriores.

   (a) Equação 1. [0.5 val.]

   Solução:

   ```
   x = 10:100
   sum(x ^ 2 + 4 * x ^ 2)
   ```

   (b) Equação 2. [0.5 val.]

   Solução:

   ```
   x = 1:10
   sum(sqrt(x) / (x + 1))
   ```

3. O conjunto de dados (`airquality`) contém registos diários de qualidade do ar, na cidade de Nova Iorque, durante 153 dias, entre Maio e Setembro de 1973. Foram registadas as seguintes variáveis:

   Ozono Concentração de O_3 (ppm – partes por milhão)

   RadSolar Radiação solar (Ly – langley)

   Vento Velocidade do vento (mph – milhas por hora)

   Temp Temperatura do ar (°F – graus Farenheit)

   ArcoIris Presença do arco-íris (valor booleano)

   Mês Mês do ano (1-12)

   Dia Dia do mês (1-31)

   As seguintes instruções executadas em **R** mostram um resumo deste conjunto de dados.

```
> head(airquality)
 Ozono RadSolar Vento Temp ArcoIris Mes Dia
 1    41      190   7.4   67     TRUE   5   1
 2    36      118   8.0   72     TRUE   5   2
 3    12      149  12.6   74    FALSE   5   3
 4    18      313  11.5   62    FALSE   5   4
 5    NA       NA  14.3   56    FALSE   5   5
 6    28       NA  14.9   66    FALSE   5   6
```

```
> str(airquality)
'data.frame':        153 obs. of  7 variables:
$ Ozono   : int  41 36 12 18 NA 28 23 19 8 NA ...
$ RadSolar: int  190 118 149 313 NA NA 299 99 19 194 ...
$ Vento   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
$ Temp    : int  67 72 74 62 56 66 65 59 61 69 ...
$ ArcoIris: logi  TRUE TRUE FALSE FALSE FALSE FALSE ...
$ Mes     : int  5 5 5 5 5 5 5 5 5 5 ...
$ Dia     : int  1 2 3 4 5 6 7 8 9 10 ...
```

(a) Crie um `data.frame` contendo **apenas** os dias e valores de temperatura em que se registou uma temperatura superior ao respectivo terceiro quartil (percentil 75). [1.5 val.]

Solução:

```
subset(airquality, Temp > quantile(Temp, .75), c(Dia, Temp))
```

(b) Crie uma função que retorne o índice de conforto climatérico (I_{CC}), definido pela Equação 3. [0.5 val.]

$$I_{CC} = \frac{Temperatura + Vento}{Ozono \times 10} \tag{3}$$

Solução:

```
icc = function (temp, wind, ozone) {
        return ((temp + wind) / (ozone * 10))
}
```

(c) Escreva o código necessário para criar o gráfico na Figura 1. [0.5 val.]

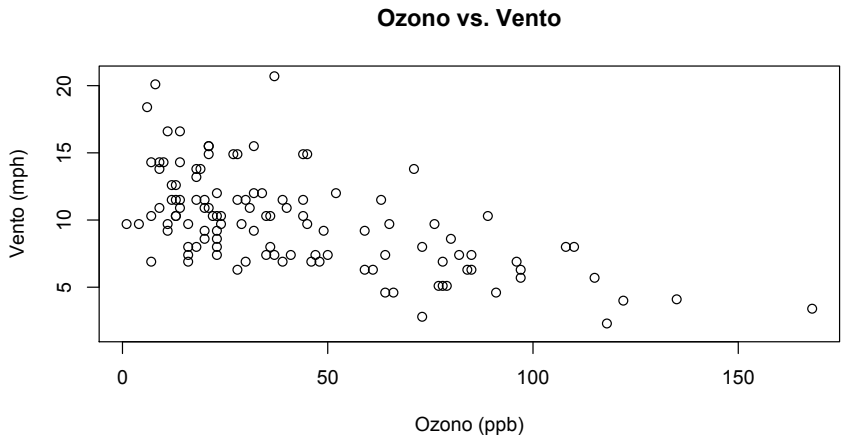

Figura 1

Solução:

```
plot(airquality$Ozono, airquality$Vento, main="Ozono vs. Vento",
        xlab="Ozono (ppb)", ylab="Vento (mph)")
```

(d) Escreva o código necessário para criar um gráfico de barras com a contagem do número de dias em que o arco-íris apareceu e não apareceu, para os dias em que a temperatura foi maior que 90°F (Figura 2). [1.5 val.]

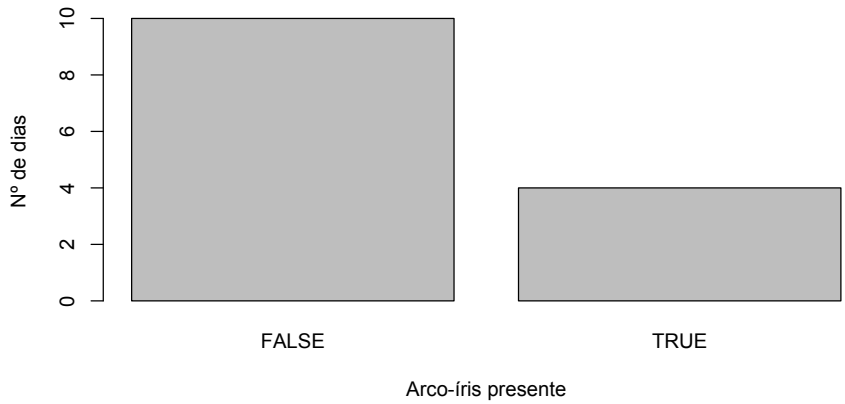

Figura 2

Solução:

```
barplot(table(subset(airquality, Temp > 90, ArcoIris)),
        xlab="Arco-íris presente", ylab="Nº de dias")
```

4. Considere que o `data.frame` `CEGI` contém a pauta com as notas dos alunos de CEGI. Atente às seguintes instruções executadas em R.

```
> head(CEGI)
R1aEpoca R2aEpoca SAS1aEpoca SAS2aEpoca
Ana        12.1     14.4       15.2       11.7
Pedro       8.2      6.6       12.3        9.9
Rui        13.3     13.3       13.3       13.3
Maria      17.2     15.1        8.2       12.2
Raquel     18.0     19.0       20.0       17.0
Duarte     10.1      7.1       12.0        8.9
> str(CEGI)
'data.frame':        131 obs. of  4 variables:
$ R1aEpoca  : num  12.1 8.2 13.3 17.2 18 10.1 ...
$ R2aEpoca  : num  14.4 6.6 13.3 15.1 19 7.1 ...
$ SAS1aEpoca: num  15.2 12.3 13.3 8.2 20 12 ...
$ SAS2aEpoca: num  11.7 9.9 13.3 12.2 17 8.9 ...
```

(a) Escreva o código necessário para guardar o conteúdo deste `data.frame` num ficheiro com nome à sua escolha. O ficheiro guardado deve ter um aspecto idêntico ao do seguinte excerto. [0.5 val.]

```
R1aEpoca|R2aEpoca|SAS1aEpoca|SAS2aEpoca
Ana|12,1|14,4|15,2|11,7
Pedro|8,2|6,6|12,3|9,9
Rui|13,3|13,3|13,3|13,3
Maria|17,2|15,1|8,2|12,2
```

Solução:

```
write.table(CEGI, 'pauta.dat', sep='|', dec=',', quote=FALSE)
```

(b) Crie uma função que permita calcular a nota final de cada aluno, sabendo que: [1.5 val.]

- R1aEpoca e R2aEpoca são as notas dos exames de 1ª e de 2ª épocas de R.
- SAS1aEpoca e SAS2aEpoca são as notas dos exames de 1ª e de 2º épocas de SAS.
- A nota final é dada pela melhor nota entre a 1ª e a 2ª época.
- A nota de cada época é dada pela média entre a nota de R e de SAS.
- Caso a nota do exame de R ou de SAS seja inferior a 8 valores o aluno reprova com 8 valores.

Depois de criar a sua função, use-a, para calcular a nota final de todos os alunos.

Solução:

```
nota_final = function (notas) {
        if (any(notas < 8)) {
                nota = 8
        } else {
                nota = max(mean(notas[1:2]), mean(notas[3:4]))
        }
        return(nota)
}
apply(CEGI, 1, nota_final)
```

5. O vector `precip` contém a precipitação (chuva) média anual, em polegadas (in), de 70 cidades norte americanas. As seguintes instruções executadas em **R** fornecem uma noção sobre a estrutura deste objecto. [2 val.]

```
> str(precip)
Named num [1:70] 67 54.7 7 48.5 14 17.2 20.7 13 43.4 40.2 ...
- attr(*, "names")= chr [1:70] "Mobile" "Juneau" "Phoenix" "Little Rock" ...
> head(precip)
Mobile      Juneau     Phoenix Little Rock Los Angeles  Sacramento
67.0        54.7         7.0        48.5        14.0        17.2
```

Explique, por palavras suas, o que se pretende encontrar com o seguinte bloco de código, e como isso é conseguido.

```
1 cidades = strsplit(names(precip), " ")
2 names(precip)[sapply(cidades, function(x) length(x) > 2)]
```

Solução:

Pretende-se encontrar os nomes das cidades, existentes no vector `precip`, cujos nomes são compostos por mais de duas palavras.

Na primeira linha, a função `strsplit` é usada para separar os nomes dos elementos do vector `precip` por palavras (espaços vazios existentes nos nomes). Esta função separa elementos do tipo caracter, dado um determinado separador, e recebe como argumentos os nomes dos elementos do vector, e o caracter correspondente ao separador, na primeira e na segunda posição, respectivamente. O resultado da utilização desta função é uma lista, guardada numa nova variável denominada `cidades`, em que cada atributo é um vector com o conjunto de palavras separadas, havendo tantos atributos quanto o comprimento do vector `precip`.

Na segunda linha, a função `sapply` é utilizada para percorrer cada atributo da nova lista, aplicando uma função anónima a cada atributo. Esta função devolve um valor lógico, conforme o número de elementos nesse atributo é ou não superior a dois. Este vector lógico resultante é então usado para indexar o vector contendo os nomes do vector `precip`.