Ejercicios opcionales tema 1

Juan Andrés Peraira Pérez

9 de abril de 2018

Ejercicio 1: Construye un vector de edades con los números 1, 3, 5, 2, 11, 9, 3, 9, 12, 3 y un vector de pesos con 4.4, 5.3, 7.2, 5.2, 8.5, 7.3, 6, 10.4, 10.2, 6.1. Consulta el largo de ambos vectores y, en caso de que sea posible, construye una matriz con los datos. Consulta los estadísticos de resumen de ambas variables. Guarda los datos en un archivo txt en la carpeta de trabajo y las órdenes con las que has trabajado

```
#-- Construcción de vectores
edades<-c(1,3,5,2,11,9,3,9,12,3)
edades
```

```
##  [1]  1  3  5  2 11  9  3  9 12  3
```

```
pesos<-c(4.4,5.3,7.2,5.2,8.5,7.3,6,10.4,10.2,6.1)
edades
```

```
##  [1]  1  3  5  2 11  9  3  9 12  3
```

```
#-- Consulatmos el largo de ambos vectores
length(edades)
```

```
## [1] 10
```

```
length(pesos)
```

```
## [1] 10
```

```
#-- Ambos tienen longitud 10
#-- Creamos la matriz y consultamos los estadísticos de resumen
datos1<-cbind(pesos,edades)
datos<-as.matrix(datos1)
datos
```

```
##       pesos edades
##  [1,]   4.4      1
##  [2,]   5.3      3
##  [3,]   7.2      5
##  [4,]   5.2      2
##  [5,]   8.5     11
##  [6,]   7.3      9
##  [7,]   6.0      3
##  [8,]  10.4      9
##  [9,]  10.2     12
## [10,]   6.1      3
```

```
summary(datos)
```

```
##      pesos            edades
##  Min.   : 4.400   Min.   : 1.0
##  1st Qu.: 5.475   1st Qu.: 3.0
```

```
##  Median : 6.650   Median : 4.0
##  Mean   : 7.060   Mean   : 5.8
##  3rd Qu.: 8.200   3rd Qu.: 9.0
##  Max.   :10.400   Max.   :12.0
```

```
## Guardamos los datos en la carpeta de trabajo en  formato txt
#-- Los datos se guardan en el siguiente directorio
getwd()
```

```
## [1] "D:/Entregas"
```

```
write.table(datos, file = "datos.txt", row.names = FALSE)
#-- Para guardar las ordenes
#savehistory (file = "datos.Rhistory")
```

Ejercicio 2: Instala el paquete vegan, actívalo y solicita ayuda para conocerlo. Realiza un ejemplo de lo que el paquete permite. Observa los datos que contiene el paquete, cárgalos y mira su encabezado.

```
#-- Instalamos el paquete
#install.packages("vegan")
#Cargamos el paquete "vegan"
library(vegan)
```

```
## Warning: package 'vegan' was built under R version 3.4.4
```

```
## Loading required package: permute
```

```
## Warning: package 'permute' was built under R version 3.4.4
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 3.4.4
```

```
## This is vegan 2.4-6
```

```
#-- Solicitamos ayuda para conocer el paquete
#help(vegan)

do <- get("{")
do(x <- 3, y <- 2*x-3, 6-x-y); x; y
```

```
## [1] 0
```

```
## [1] 3
```

```
## [1] 3
```

```
do
```

```
## .Primitive("{")
```

```
#-- Cargamos los datos del paquete
data("varespec")
#-- seleccionamos en encabezado
head(varespec)
```

```
##    Callvulg Empenigr Rhodtome Vaccmyrt Vaccviti Pinusylv Descflex Betupube
## 18     0.55    11.13     0.00     0.00    17.80     0.07     0.00        0
## 15     0.67     0.17     0.00     0.35    12.13     0.12     0.00        0
```

```
## 24     0.10     1.55     0.00     0.00    13.47     0.25     0.00        0
## 27     0.00    15.13     2.42     5.92    15.97     0.00     3.70        0
## 23     0.00    12.68     0.00     0.00    23.73     0.03     0.00        0
## 19     0.00     8.92     0.00     2.42    10.28     0.12     0.02        0
##    Vacculig Diphcomp Dicrsp Dicrfusc Dicrpoly Hylosple Pleuschr Polypili
## 18     1.60     2.07   0.00     1.62     0.00      0.0     4.67     0.02
## 15     0.00     0.00   0.33    10.92     0.02      0.0    37.75     0.02
## 24     0.00     0.00  23.43     0.00     1.68      0.0    32.92     0.00
## 27     1.12     0.00   0.00     3.63     0.00      6.7    58.07     0.00
## 23     0.00     0.00   0.00     3.42     0.02      0.0    19.42     0.02
## 19     0.00     0.00   0.00     0.32     0.02      0.0    21.03     0.02
##    Polyjuni Polycomm Pohlnuta Ptilcili Barbhatc Cladarbu Cladrang Cladstel
## 18     0.13     0.00     0.13     0.12     0.00    21.73    21.47     3.50
## 15     0.23     0.00     0.03     0.02     0.00    12.05     8.13     0.18
## 24     0.23     0.00     0.32     0.03     0.00     3.58     5.52     0.07
## 27     0.00     0.13     0.02     0.08     0.08     1.42     7.63     2.55
## 23     2.12     0.00     0.17     1.80     0.02     9.08     9.22     0.05
## 19     1.58     0.18     0.07     0.27     0.02     7.23     4.95    22.08
##    Cladunci Cladcocc Cladcorn Cladgrac Cladfimb Cladcris Cladchlo Cladbotr
## 18     0.30     0.18     0.23     0.25     0.25     0.23     0.00     0.00
## 15     2.65     0.13     0.18     0.23     0.25     1.23     0.00     0.00
## 24     8.93     0.00     0.20     0.48     0.00     0.07     0.10     0.02
## 27     0.15     0.00     0.38     0.12     0.10     0.03     0.00     0.02
## 23     0.73     0.08     1.42     0.50     0.17     1.78     0.05     0.05
## 19     0.25     0.10     0.25     0.18     0.10     0.12     0.05     0.02
##    Cladamau Cladsp Cetreric Cetrisla Flavniva Nepharct Stersp Peltapht
## 18     0.08   0.02     0.02     0.00     0.12     0.02   0.62     0.02
## 15     0.00   0.00     0.15     0.03     0.00     0.00   0.85     0.00
## 24     0.00   0.00     0.78     0.12     0.00     0.00   0.03     0.00
## 27     0.00   0.02     0.00     0.00     0.00     0.00   0.00     0.07
## 23     0.00   0.00     0.00     0.00     0.02     0.00   1.58     0.33
## 19     0.00   0.00     0.00     0.00     0.02     0.00   0.28     0.00
##    Icmaeric Cladcerv Claddefo Cladphyl
## 18        0        0     0.25        0
## 15        0        0     1.00        0
## 24        0        0     0.33        0
## 27        0        0     0.15        0
## 23        0        0     1.97        0
## 19        0        0     0.37        0
```

Ejercicio 3: Cuando trabajamos con factores, a veces nos interesa conocer cuantas observaciones hay de cada clase o categoría. La frecuencia de cada una de las categorías se puede obtener facilmente con la funcion table(). Crea un factor de 30 elementos con tres categorías (1, 2 y 3) asignadas al azar. Etiqueta las categorías como Castellón, Valencia y Alicante. Utiliza la funcion table() para calcular cuántas observaciones hay de cada categoría.

```
#-- Creamos un vector con los nombre
 x<-c("Castellon","Valencia","Alicante")
#-- Con la función sample generamos una muestra aleatoria de 30 observaciones del vector
#-- creado y lo transformamos a factor
muestra<-sample(as.factor(x),30,replace=T)
```

```r
table(muestra)
```

```
## muestra
##  Alicante Castellon  Valencia
##        14         9         7
```

```r
str(muestra)
```

```
##  Factor w/ 3 levels "Alicante","Castellon",..: 1 1 3 3 1 2 2 1 1 1 ...
```

Ejercicio 4: Con los siguientes numeros: 7.3, 6.8, 0.005, 9, 12, 2.4, 18.9, .9 a) Calcula la media. b) ¿Cuántos valores son mayores que 1?, c) Calcula la raíz cuadrada de los numeros. d) Obtén los numeros que son mayores que su raíz cuadrada. e) Redondea los datos de la raíz cuadrada para que tengan solo 1 cifra decimal.

```r
x<-c(7.3,6.8,0.005,9,12,2.4,18.9,0.9)
x
```

```
## [1]  7.300  6.800  0.005  9.000 12.000  2.400 18.900  0.900
```

```r
media<-mean(x)
media
```

```
## [1] 7.163125
```

```r
table(x>1)
```

```
##
## FALSE  TRUE
##     2     6
```

```r
#-- En este caso existen 6 valores mayores a 1
#-- Raiz cuadrada de  los números
raiz<-sqrt(x)
raiz
```

```
## [1] 2.70185122 2.60768096 0.07071068 3.00000000 3.46410162 1.54919334
## [7] 4.34741302 0.94868330
```

```r
#-- Números mayores a su raiz cuadrada
media_raiz<-function(vector){
  raiz_cuadrada<-sqrt(vector)
  for (i in 1:length(vector)) {
    if(vector[i]>raiz_cuadrada[i]) print(vector[i])

  }
}
media_raiz(x)
```

```
## [1] 7.3
## [1] 6.8
## [1] 9
## [1] 12
## [1] 2.4
## [1] 18.9
```

```r
#-- Redondeo de los datos de la raiz cuadrada para que solo tengan un decimal
round(raiz,1)
```

```
## [1] 2.7 2.6 0.1 3.0 3.5 1.5 4.3 0.9
```

Ejercicio 5: Utiliza las funciones rep() y seq() para producir un vector que contenga: a) los valores: 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, b) los valores: 4, 4, 4, 4, 3, 3, 3, 3, 2, 2, 2, 2, 1, 1, 1, 1, c) los valores: 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5.

```r
a<-rep(c(1,2,3,4),4)
a
```

```
##  [1] 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
```

```r
b<-c(rep(4,4),rep(3,4),rep(2,4),rep(1,4))
b
```

```
##  [1] 4 4 4 4 3 3 3 3 2 2 2 2 1 1 1 1
```

```r
c<-c(rep(1,1),rep(2,2),rep(3,3),rep(4,4),rep(5,5))
c
```

```
##  [1] 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
```

Ejercicio 6: Use la funcion sample() para obtener una muestra de tamaño 5 de un vector con valores 1:20 y probabilidades proporcionales.

```r
muestra<-sample(1:20,5)
muestra
```

```
## [1] 12 16 10 13  3
```

Ejercicio 7: Utiliza la función reshape() con los datos iris para poder realizar un gráfico con la función bwplot() del paquete lattice.

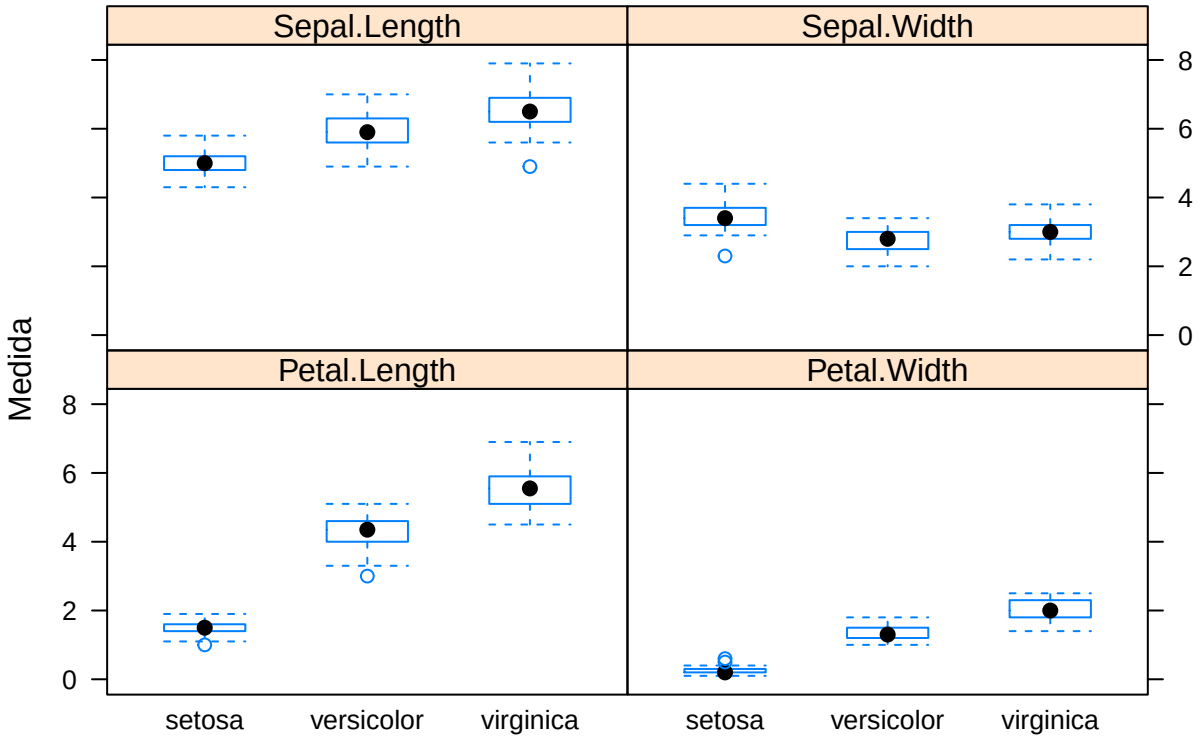
```
#-- Cargamos los datos de "iris"
datos<-iris
head(datos)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
#-- Instalamos y cargamos los paquetes necesarios
# install.packages("reshape")
# library(reshape)
#install.packages("lattice")
#library(lattice)
iris_reshape <- reshape(datos, varying=1:4, v.names="Medida",
                        timevar="Dimension", times=names(datos)[1:4],
                        idvar="ID medida", direction="long")
head(iris_reshape)
```

```
##                Species    Dimension Medida ID medida
## 1.Sepal.Length  setosa Sepal.Length    5.1         1
## 2.Sepal.Length  setosa Sepal.Length    4.9         2
## 3.Sepal.Length  setosa Sepal.Length    4.7         3
## 4.Sepal.Length  setosa Sepal.Length    4.6         4
## 5.Sepal.Length  setosa Sepal.Length    5.0         5
## 6.Sepal.Length  setosa Sepal.Length    5.4         6
```

```
bwplot(Medida ~ Species | Dimension, data=iris_reshape)
```



Ejercicio 8: Crea una función para calcular el salario de un trabajador que cobra 20 euros la hora y que trabaja 4 horas diarias. Calcula con esta función cuánto ha de ganar al mes si trabaja los 5 días de la semana.

```
sal<-function(horas,precio=20){
  salario<-horas*precio
  return(paste(salario,"euros"))
}
```

```
#-- Dado que trabaja 4 horas al día, a la semana son 20 horas
#-- y por consiguiente al més son 80 horas
salario<-sal(80)
salario
```

```
## [1] "1600 euros"
```

Ejercicio 9: Crea una nueva función que incluya la siguiente información: el trabajador cobra las horas extra a 40 euros. Calcula con esta función cuál será su salario si este mes ha trabajado 5 horas extras.

```
sal_2<-function(horas,hora_extra=NA){
  precio=20
  precio_hora_extra=40
```

```
  if(!is.null(hora_extra) & !is.na(hora_extra)){
  salario<-(horas*precio) + (hora_extra*precio_hora_extra)
  } else{salario<-(horas*precio)}

  return(paste(salario,"euros"))
}

#-- En la función aparece por defecto horas extras vacias, si se indican
#-- cuantas son la función las tiene encuenta.

#-- Ejemplo con horas extras.
salario<-sal_2(80,5)
salario
```

```
## [1] "1800 euros"
```

```
#-- Ejemplo sin horas extras.
salario<-sal_2(80)
salario
```

```
## [1] "1600 euros"
```

Ejercicio 10: Imagina que una empresa envía paquetes por correo a 10 euros el paquete, pero incluye una oferta donde si se entregan más de 50 paquetes se realiza un 10% de descuento en el precio final. Construye una función donde se calcule cada caso y ponlo a prueba.

```
paquete<-function(paquetes){
  precio=10
  if(paquetes>50) {
    resultado = (paquetes * precio) -(paquetes * precio *0.10)
  } else {resultado = paquetes * precio}
  return(paste(resultado,"euros"))
}

#-- Comprobación con menos de 50 paquetes
resultado<-paquete(30)
resultado
```

```
## [1] "300 euros"
```

```
#-- Comprobación con mas de 50 paquetes
resultado<-paquete(68)
resultado
```

```
## [1] "612 euros"
```