

FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE
DEPARTAMENTO DE MATEMÁTICAS

TITULO:

Practicas resueltas del CURSO de R

Fernando Ernesto Manzanares Morán

Docente:
Salvador Enrique Rodriguez Hernandez

Practica 2.

```
> v <- numeric(3);v
[1] 0 0 0
> v[3] <- 17; v
[1] 0 0 17
> x <- c(2, 4, 3.1, 8, 6)
> x
[1] 2.0 4.0 3.1 8.0 6.0
> is.integer(x)
[1] FALSE
> is.double(x)
[1] TRUE
> length(x)
[1] 5
> x <- edit(x)
> y = 1:4; y
[1] 1 2 3 4
> y[2] <- 5
> u <- 1:12
> u
[1] 1 2 3 4 5 6 7 8 9 10 11 12
> u1=u[2 * 1:5]
> u1
[1] 2 4 6 8 10
> assign("z", c(x, 0, x))
> z
[1] 2.0 4.0 3.1 8.0 6.0 0.0 2.0 4.0 3.1 8.0 6.0
> s1 <- seq(2, 10); s1
[1] 2 3 4 5 6 7 8 9 10
```

```

> s2 = seq(from=-1, to=5); s2

[1] -1  0  1  2  3  4  5

> s3<-seq(to=2, from=-2); s3

[1] -2 -1  0  1  2

> s4=seq(from=-3, to=3, by=0.2); s4

[1] -3.0 -2.8 -2.6 -2.4 -2.2 -2.0 -1.8 -1.6 -1.4 -1.2 -1.0 -0.8 -0.6 -0.4 -0.2
[16]  0.0  0.2  0.4  0.6  0.8  1.0  1.2  1.4  1.6  1.8  2.0  2.2  2.4  2.6  2.8
[31]  3.0

> s5 <- rep(s3, times=3); s5

[1] -2 -1  0  1  2 -2 -1  0  1  2 -2 -1  0  1  2

> 1/x

[1] 0.5000000 0.2500000 0.3225806 0.1250000 0.1666667

> v=2*x+z+1
> v

[1]  7.0 13.0 10.3 25.0 19.0  5.0 11.0 11.2 20.1 21.0 11.0

> e<- c(1, 2, 3, 4); e2<-c(4, 5, 6, 7); crossprod(e, e2)

      [,1]
[1,]    60

> xt = t(x)
> xt

      [,1] [,2] [,3] [,4] [,5]
[1,]     2     4  3.1     8     6

> u = exp(y);u

[1]  2.718282 148.413159  20.085537  54.598150

> options(digits=10); u

[1]  2.718281828 148.413159103  20.085536923  54.598150033

> resum <- c(length(y),sum(y), prod(y), min(y), max(y)); resum

[1]  4 13 60  1  5

> yo <- sort(y); yo

```

```

[1] 1 3 4 5

> deptos <- c("Santa Ana", "Sonsonate", "San Salvador"); deptos

[1] "Santa Ana"      "Sonsonate"      "San Salvador"

> deptos[4]="Ahuachap      "; deptos

[1] "Santa Ana"      "Sonsonate"      "San Salvador" "Ahuachap      "

> codDeptos <- c(11, 12, 13, 14)
> Oriente <- codDeptos [c("La Uni      ", "San Miguel")];Oriente

[1] NA NA

> etiqs<-paste(c("X", "Y"), 1:10, sep=""); etiqs

[1] "X1" "Y2" "X3" "Y4" "X5" "Y6" "X7" "Y8" "X9" "Y10"

> M <- matrix(numeric(), nrow = 3, ncol=4)
> M[2,3] <- 6
> M

      [,1] [,2] [,3] [,4]
[1,]    NA    NA    NA    NA
[2,]    NA    NA     6    NA
[3,]    NA    NA    NA    NA

> A <- matrix(c(2, 4, 6, 8, 10, 12), nrow=2, ncol=3)
> A

      [,1] [,2] [,3]
[1,]     2     6    10
[2,]     4     8    12

> mode(A)

[1] "numeric"

> dim(A)

[1] 2 3

> attributes(A)

$dim
[1] 2 3

> is.matrix(A)

```

```

[1] TRUE

> is.array(A)

[1] TRUE

> B <- matrix(1:12, nrow=3, ncol=4)
> B

      [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

> x1 <- seq(0, 10, 2); x1

[1]  0  2  4  6  8 10

> x2 <- seq(1, 11, 2); x2

[1]  1  3  5  7  9 11

> x3 <- runif(6); x3

[1] 0.42274589790 0.01166639803 0.30733486149 0.95921583986 0.47616313654
[6] 0.26931536943

> Xcol <- cbind(x1, x2, x3); Xcol

      x1 x2      x3
[1,]  0  1 0.42274589790
[2,]  2  3 0.01166639803
[3,]  4  5 0.30733486149
[4,]  6  7 0.95921583986
[5,]  8  9 0.47616313654
[6,] 10 11 0.26931536943

> Xfil <- rbind(x1, x2, x3); Xfil

      [,1]      [,2]      [,3]      [,4]      [,5]
x1 0.0000000000 2.0000000000 4.0000000000 6.0000000000 8.0000000000
x2 1.0000000000 3.0000000000 5.0000000000 7.0000000000 9.0000000000
x3 0.4227458979 0.0116663980 0.3073348615 0.9592158399 0.4761631365
      [,6]
x1 10.0000000000
x2 11.0000000000
x3  0.2693153694

> X <- Xfil[1:3, c(2, 3)]; X

```

```

      [,1]      [,2]
x1 2.000000000000 4.000000000000
x2 3.000000000000 5.000000000000
x3 0.01166639803 0.3073348615

> v<-c(1, 2); v %*%A

      [,1] [,2] [,3]
[1,]    10    22    34

> P <- A %*% B; P

      [,1] [,2] [,3] [,4]
[1,]    44    98   152   206
[2,]    56   128   200   272

> 2*A

      [,1] [,2] [,3]
[1,]     4    12    20
[2,]     8    16    24

> length(A)

[1] 6

> T=sqrt(B); T

      [,1]      [,2]      [,3]      [,4]
[1,] 1.000000000 2.000000000 2.645751311 3.162277660
[2,] 1.414213562 2.236067977 2.828427125 3.316624790
[3,] 1.732050808 2.449489743 3.000000000 3.464101615

> t(A)

      [,1] [,2]
[1,]     2     4
[2,]     6     8
[3,]    10    12

> C <- matrix(c(2, 1, 10, 12), nrow=2, ncol=2); C

      [,1] [,2]
[1,]     2    10
[2,]     1    12

> det(C)

[1] 14

```

```

> InvC <- solve(C)
> eigen(C)

$values
[1] 12.916079783 1.083920217

$vectors
      [,1]      [,2]
[1,] -0.6754894393 -0.99583021557
[2,] -0.7373696613 0.09122599279

> c(length(A), sum(A), prod(A), min(A), max(A))

[1] 6 42 46080 2 12

> nombres <- matrix(c("Carlos", "JosÃl'", "Caren", "RenÃl'", "Mar??a", "Mario"),
+                    nrow=3, ncol=2); nombres

      [,1] [,2]
[1,] "Carlos" "RenÃl'"
[2,] "JosÃl'" "Mar??a"
[3,] "Caren" "Mario"

> X <- array(c(1, 3, 5, 7, 9, 11), dim=c(2, 3)); X

      [,1] [,2] [,3]
[1,] 1 5 9
[2,] 3 7 11

> Z <- array(1, c(3, 3)); Z

      [,1] [,2] [,3]
[1,] 1 1 1
[2,] 1 1 1
[3,] 1 1 1

> W <- 2*Z+1
> W

      [,1] [,2] [,3]
[1,] 3 3 3
[2,] 3 3 3
[3,] 3 3 3

> TX <- t(X)
> TX

      [,1] [,2]
[1,] 1 3
[2,] 5 7
[3,] 9 11

```

```

> a <- c(2, 4, 6)
> a

[1] 2 4 6

> b <- 1:3
> b

[1] 1 2 3

> M <- a %o% b
> M

      [,1] [,2] [,3]
[1,]    2    4    6
[2,]    4    8   12
[3,]    6   12   18

> Arreglo3 <- array(c(1:8, 11:18, 111:118), dim = c(2, 4, 3))
> Arreglo3

, , 1

      [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2

      [,1] [,2] [,3] [,4]
[1,]   11   13   15   17
[2,]   12   14   16   18

, , 3

      [,1] [,2] [,3] [,4]
[1,]  111  113  115  117
[2,]  112  114  116  118

>
>
>
>
>
>
>
>
>
>
>

```


Practica 3.

```
> sexo <- c("M", "F", "F", "M", "F", "F", "M")
> sexo

[1] "M" "F" "F" "M" "F" "F" "M"

> edad <- c(19, 20, 19, 22, 20, 21, 19)
> edad

[1] 19 20 19 22 20 21 19

> FactorSexo = factor(sexo)
> FactorSexo

[1] M F F M F F M
Levels: F M

> mediaEdad <- tapply(edad, FactorSexo, mean)
> mediaEdad

  F  M
20 20

> is.vector(mediaEdad)

[1] FALSE

> is.matrix(mediaEdad)

[1] FALSE

> is.list(mediaEdad)

[1] FALSE

> is.table(mediaEdad)

[1] FALSE

> is.array(mediaEdad)

[1] TRUE

> factor()

factor(0)
Levels:
```

```
> lista1<-list(padre="Pedro", madre="Mar?a", no.hijos=3, edad.hijos=c(4,7,9))
> lista1
```

```

$padre
[1] "Pedro"

$madre
[1] "Mar?a"

$no.hijos
[1] 3

$edad.hijos
[1] 4 7 9

> is.matrix(lista1)

[1] FALSE

> is.vector(lista1$edad.hijos)

[1] TRUE

> lista1["madre"]

$madre
[1] "Mar?a"

> lista1[[4]][2]

[1] 7

> lista1["padre"]

$padre
[1] "Pedro"

> lista1$padre

[1] "Pedro"

> lista1$edad.hijos[2]

[1] 7

> lista1[[4]][2]

[1] 7

> lista1[["pedro"]]

NULL

```

```

> x <- "nombre"; lista1[x]

$<NA>
NULL

> subLista <- lista1[4]; subLista

$edad.hijos
[1] 4 7 9

> lista1[5] <- list(sexo.hijos=c("F", "M", "F")); lista1

$padre
[1] "Pedro"

$madre
[1] "Mar?a"

$no.hijos
[1] 3

$edad.hijos
[1] 4 7 9

[[5]]
[1] "F" "M" "F"

> lista1 <- edit(lista1)
> S <- matrix(c(3, -sqrt(2), -sqrt(2), 2), nrow=2, ncol=2);S

      [,1]      [,2]
[1,]  3.000000000 -1.414213562
[2,] -1.414213562  2.000000000

> autovS <- eigen(S); autovS

$values
[1] 4 1

$vectors
      [,1]      [,2]
[1,] -0.8164965809 -0.5773502692
[2,]  0.5773502692 -0.8164965809

> evals <- eigen(S)$values; evals

[1] 4 1

```

```

> Notas <- matrix(c(2, 5, 7, 6, 8, 2, 4, 9, 10), ncol=3,
+                 dimnames=list(c("Matem?tica", "?lgebra", "Geometr?a"),
+                               c("Juan", "Jos?", "Ren?"))); Notas

      Juan Jos? Ren?
Matem?tica    2    6    4
?lgebra       5    8    9
Geometr?a     7    2   10

> ncol=(3)
> log <- sample(c(TRUE, FALSE), size = 20, replace = TRUE)
> log

[1] TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE FALSE TRUE
[13] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE

> comp <- rnorm(20) + runif(20) * (1i)
> comp

[1] 0.3268162652+0.8201739625i 0.6551202575+0.2958150213i
[3] 1.5167326485+0.6684645915i 1.9904253553+0.7022224779i
[5] 0.2497033665+0.5298961878i -1.8313806466+0.0499122052i
[7] 0.2194819298+0.8149696330i 1.4849877983+0.7153704786i
[9] 0.2994140876+0.4602233020i 0.1071140547+0.6764138630i
[11] -0.7396235287+0.6143820933i -0.1632875248+0.6013764038i
[13] -1.0378319110+0.6787370704i 0.8639115601+0.6922727772i
[15] 0.7947920817+0.4594895071i 0.9555788704+0.8493550471i
[17] -0.2824005815+0.3403120476i 0.2384509463+0.3103362380i
[19] -1.8527352285+0.2581272351i -0.2563747839+0.6799715257i

> num <- rnorm(20, mean=0, sd=1)
> num

[1] 0.73024328080 -0.87366402412 -0.82766035445 -0.10465739694 0.56595154742
[6] 0.78526981986 -1.05677859471 0.45881856864 -0.02889866804 -0.33173140101
[11] -0.62539777495 0.60296756454 2.14749532606 0.66210444221 -1.09262944901
[16] 0.87588077884 0.97946220227 1.51209852228 0.67376478424 0.23545680037

> df1 <- data.frame(log, comp, num)
> df1

      log      comp      num
1  TRUE 0.3268162652+0.8201739625i 0.73024328080
2  TRUE 0.6551202575+0.2958150213i -0.87366402412
3 FALSE 1.5167326485+0.6684645915i -0.82766035445
4 FALSE 1.9904253553+0.7022224779i -0.10465739694
5  TRUE 0.2497033665+0.5298961878i 0.56595154742
6 FALSE -1.8313806466+0.0499122052i 0.78526981986

```

```

7 TRUE 0.2194819298+0.8149696330i -1.05677859471
8 FALSE 1.4849877983+0.7153704786i 0.45881856864
9 FALSE 0.2994140876+0.4602233020i -0.02889866804
10 FALSE 0.1071140547+0.6764138630i -0.33173140101
11 FALSE -0.7396235287+0.6143820933i -0.62539777495
12 TRUE -0.1632875248+0.6013764038i 0.60296756454
13 TRUE -1.0378319110+0.6787370704i 2.14749532606
14 TRUE 0.8639115601+0.6922727772i 0.66210444221
15 TRUE 0.7947920817+0.4594895071i -1.09262944901
16 TRUE 0.9555788704+0.8493550471i 0.87588077884
17 FALSE -0.2824005815+0.3403120476i 0.97946220227
18 FALSE 0.2384509463+0.3103362380i 1.51209852228
19 FALSE -1.8527352285+0.2581272351i 0.67376478424
20 FALSE -0.2563747839+0.6799715257i 0.23545680037

> nombres <- c("logico", "complejo", "numerico")
> names(df1) <- nombres; df1

  logico      complejo      numerico
1 TRUE 0.3268162652+0.8201739625i 0.73024328080
2 TRUE 0.6551202575+0.2958150213i -0.87366402412
3 FALSE 1.5167326485+0.6684645915i -0.82766035445
4 FALSE 1.9904253553+0.7022224779i -0.10465739694
5 TRUE 0.2497033665+0.5298961878i 0.56595154742
6 FALSE -1.8313806466+0.0499122052i 0.78526981986
7 TRUE 0.2194819298+0.8149696330i -1.05677859471
8 FALSE 1.4849877983+0.7153704786i 0.45881856864
9 FALSE 0.2994140876+0.4602233020i -0.02889866804
10 FALSE 0.1071140547+0.6764138630i -0.33173140101
11 FALSE -0.7396235287+0.6143820933i -0.62539777495
12 TRUE -0.1632875248+0.6013764038i 0.60296756454
13 TRUE -1.0378319110+0.6787370704i 2.14749532606
14 TRUE 0.8639115601+0.6922727772i 0.66210444221
15 TRUE 0.7947920817+0.4594895071i -1.09262944901
16 TRUE 0.9555788704+0.8493550471i 0.87588077884
17 FALSE -0.2824005815+0.3403120476i 0.97946220227
18 FALSE 0.2384509463+0.3103362380i 1.51209852228
19 FALSE -1.8527352285+0.2581272351i 0.67376478424
20 FALSE -0.2563747839+0.6799715257i 0.23545680037

> row.names(df1) <- letters[1:20]
> df1

  logico      complejo      numerico
a TRUE 0.3268162652+0.8201739625i 0.73024328080
b TRUE 0.6551202575+0.2958150213i -0.87366402412
c FALSE 1.5167326485+0.6684645915i -0.82766035445

```

```

d FALSE 1.9904253553+0.7022224779i -0.10465739694
e TRUE 0.2497033665+0.5298961878i 0.56595154742
f FALSE -1.8313806466+0.0499122052i 0.78526981986
g TRUE 0.2194819298+0.8149696330i -1.05677859471
h FALSE 1.4849877983+0.7153704786i 0.45881856864
i FALSE 0.2994140876+0.4602233020i -0.02889866804
j FALSE 0.1071140547+0.6764138630i -0.33173140101
k FALSE -0.7396235287+0.6143820933i -0.62539777495
l TRUE -0.1632875248+0.6013764038i 0.60296756454
m TRUE -1.0378319110+0.6787370704i 2.14749532606
n TRUE 0.8639115601+0.6922727772i 0.66210444221
o TRUE 0.7947920817+0.4594895071i -1.09262944901
p TRUE 0.9555788704+0.8493550471i 0.87588077884
q FALSE -0.2824005815+0.3403120476i 0.97946220227
r FALSE 0.2384509463+0.3103362380i 1.51209852228
s FALSE -1.8527352285+0.2581272351i 0.67376478424
t FALSE -0.2563747839+0.6799715257i 0.23545680037

```

```
> edad <- c(18, 21, 45, 54); edad
```

```
[1] 18 21 45 54
```

```
> datos <- matrix(c(150, 160, 180, 205, 65, 68, 65, 69), ncol=2, dimnames=list(c(),
+                                                                                   c("Estatura",
```

```

      Estatura Peso
[1,]      150   65
[2,]      160   68
[3,]      180   65
[4,]      205   69

```

```
> sexo <- c("F", "M", "M", "M"); sexo
```

```
[1] "F" "M" "M" "M"
```

```
> hojal <- data.frame(Edad=edad, datos, Sexo=sexo)
```

```
> hojal
```

```

  Edad Estatura Peso Sexo
1   18      150   65    F
2   21      160   68    M
3   45      180   65    M
4   54      205   69    M

```

```
> search()
```

```

[1] ".GlobalEnv"      "package:stats"    "package:graphics"
[4] "package:grDevices" "package:utils"    "package:datasets"
[7] "package:methods"  "Autoloads"        "package:base"

```

```

> attach(hoja1)
> search()

[1] ".GlobalEnv"          "hoja1"          "package:stats"
[4] "package:graphics"    "package:grDevices" "package:utils"
[7] "package:datasets"    "package:methods"  "Autoloads"
[10] "package:base"

> Edad

[1] 18 21 45 54

> hoja1$Peso <- Peso+1
> hoja1

  Edad Estatura Peso Sexo
1   18      150   66    F
2   21      160   69    M
3   45      180   66    M
4   54      205   70    M

> detach(hoja1)
> edad

[1] 18 21 45 54

>
>
>

```

Practica 4.

```

Entrada1 <- read.table("datos01.txt", header=TRUE) Entrada1 Edat1 <-
scan("datos01.txt", list(X1=0, X2=0), skip = 1, flush = TRUE, quiet = TRUE)
Edat1 pp <- scan("datos02.txt", skip = 1, quiet= TRUE) pp
library(foreign) baseproductos <-read.table("productos.csv",header=TRUE,sep
= ";") baseproductos
library(Hmisc) Baseimportante<-spss.get("Mundo.sav",use.value.labels =TRUE)
Baseimportante

```

Practica 5.

```

> x <- c(6:10)
> x

[1] 6 7 8 9 10

> sqrt(x)

[1] 2.449489743 2.645751311 2.828427125 3.000000000 3.162277660

```

```

> sqrt(ifelse(x >= 0, x, NA))

[1] 2.449489743 2.645751311 2.828427125 3.000000000 3.162277660

> x <- c(2, 6, 4, 7, 5, 1)
> x

[1] 2 6 4 7 5 1

> suma<-0; for(i in 1:3) suma = suma+x[i]
> suma

[1] 12

> media <- function(x)
+ {
+   n = length(x)
+   suma <- 0.0
+   for(i in 1:n) suma = suma + x[i]
+   media = suma/n
+ }
> func.cuadratica <- function(x)
+ {
+   3*x^2-5*x+2
+ }
> y <- func.cuadratica(2)
> y

[1] 4

> save(media, file= "media.RData")
> rm(list=ls(all=TRUE))
> load("media.RData")
> media <- function(x)
+ {
+   n = length(x)
+   suma <- 0.0
+   for(i in 1:n) suma = suma + x[i]
+   media = suma/n
+ }
> x <- 1:5
> (media(x))

[1] 3

> y <- c(5, 8 , 4, 9)
> (media(y))

```



```
[1] 6.5
```

```
> Seno <- function(x)
+ {
+ y = sin(x)
+ plot(x, y, main="Ejemplo de gr?ficos en R",
+ xlab="x", ylab="y = Seno(x)", col="blue", pch=1)
+ }
> x<-seq(-pi, pi, len=100)
> Seno(x)
> func.cuadratica <- function(x)
+ {
+ 3*x^2-5*x+2
+ }
> y <- func.cuadratica(2)
> y
```

```
[1] 4
```

```
> media <- function(x)
+ {
+ n = length(x)
+ suma <- 0.0
+ for(i in 1:n) suma = suma + x[i]
+ media = suma/n
+ }
>
```

```
> library(splines)
> library( RcmdrMisc)
> library(car)
> library(sandwich)
> library(relimp, pos=15)
>
>
>
```

Practica 6.

```
> #"CC"=Coca_Cola
> #"PC"=Pepsi_Cola
> #"SC"=Salva_Cola
> Tipo<-c("CC", "PC", "SC");Tipo
```

```
[1] "CC" "PC" "SC"
```

```
> Consumo<-sample(Tipo,20,replace=TRUE);Consumo
```

```

[1] "PC" "CC" "SC" "PC" "PC" "CC" "PC" "SC" "PC" "CC" "SC" "CC" "SC" "SC" "PC"
[16] "CC" "CC" "CC" "PC" "SC"

> data.entry(Consumo)
> write(Consumo, "Consumo.txt")
> frec <- table(Consumo); frec

Consumo
CC PC SC
 7  7  6

> prop <- table(Consumo)/length(Consumo); prop

Consumo
  CC  PC  SC
0.35 0.35 0.30

> summary(Consumo)

      Length      Class      Mode
      20 character character

> barplot(frec, main="Gráfico de barras", xlab=" Consumo", col=c("yellow", "white", "red"),
+ sub="Agosto-2012")
> barplot(prop, main="Gráfico de barras", xlab=" Consumo\n", col=c("yellow", "white",
+ "red"), sub="Agosto-2012")
> pie(frec, main="Gráfico de pastel", xlab="Tipo de Consumo", col=c("yellow", "white",
+ "cyan"), sub="Agosto-2012")
> names(frec) = c("Coca Cola", "Pepsi", "Salva Cola")
> pie(frec, main="Gráfico de pastel", xlab=" Consumo", radius=1, col=c("red", "gray",
+ "cyan"), sub="Agosto-2012")
> n <- length(frec)
> hoja <- data.frame(frec); hoja

      Var1 Freq
1 Coca Cola   7
2 Pepsi       7
3 Salva Cola   6

> etiq <- c(paste(hoja$Var1, "-", hoja$Freq)); etiq

[1] "Coca Cola - 7" "Pepsi - 7"      "Salva Cola - 6"

> pie(frec, main="Gráfico de pastel", labels=etiq, col=rainbow(n), border=TRUE)

```

Practica 7.

```

> Hijos<-c(2,1,2,1,4,2,3,0,2,3,3,2,1,0,2,4,1,2,1,3,4,1,2,3,1,5,2,3,1,2)
> data.entry(Hijos)
> Hijos

```

```

[1] 2 1 2 1 4 2 3 0 2 3 3 2 1 0 2 4 1 2 1 3 4 1 2 3 1 5 2 3 1 2

> length(Hijos)

[1] 30

> write(Hijos, "Hijos.txt")
> ls()

[1] "Consumo"          "etiq"              "frec"              "func.cuadratica"
[5] "Hijos"            "hoja"              "media"             "n"
[9] "prop"             "Seno"              "Tipo"              "x"
[13] "y"

> rm(list=ls(all=TRUE)); ls()

character(0)

> X <- scan("Hijos.txt", what = integer(0), na.strings = "NA", flush=FALSE)
> ls()

[1] "X"

> stripchart(X, method="stack", vertical=FALSE, col="blue", pch=1, main="Gráfico de\n
+ puntos", xlab="Número de hijos")
> fab <- table(X); fab

X
 0  1  2  3  4  5
2  8 10  6  3  1

> fre <- fab/length(X); fre

X
      0      1      2      3      4
0.06666666667 0.26666666667 0.33333333333 0.20000000000 0.10000000000
      5
0.03333333333

> Fac <- cumsum(fab); Fac

 0  1  2  3  4  5
2 10 20 26 29 30

> Far <- Fac/length(X); Far

      0      1      2      3      4
0.06666666667 0.33333333333 0.66666666667 0.86666666667 0.96666666667
      5
1.00000000000

```

```

> options(digits=2)
> tabla <- data.frame(fab=fab, fre=fre, Fac=Fac, Far=Far)
> names(tabla) <- c("X", "fab", "free.X", "fre", "Fac", "Far")
> tabla

  X fab free.X  fre Fac  Far
0 0  2      0 0.067  2 0.067
1 1  8      1 0.267 10 0.333
2 2 10      2 0.333 20 0.667
3 3  6      3 0.200 26 0.867
4 4  3      4 0.100 29 0.967
5 5  1      5 0.033 30 1.000

> tfre <- data.frame(X=tabla$X, fab=tabla$fab, fre=tabla$fre, Fac=tabla$Fac, Far=tabla$Far)
> tfre

  X fab  fre Fac  Far
1 0  2 0.067  2 0.067
2 1  8 0.267 10 0.333
3 2 10 0.333 20 0.667
4 3  6 0.200 26 0.867
5 4  3 0.100 29 0.967
6 5  1 0.033 30 1.000

> media <- mean(X, na.rm = FALSE); media

[1] 2.1

> for(i in 1:length(X)) if (fab[i] == max(fab)) break()
> moda <- names(fab[i]); moda # R no tiene incorporada una funci3n para la moda

[1] "2"

> mediana <- median(X); mediana

[1] 2

> range(X)

[1] 0 5

> cuasivar <- var(X); cuasivar

[1] 1.5

> s <- sd(X); s

[1] 1.2

```

```

> quantile(X,c(0.25, 0.5, 0.75))

25% 50% 75%
 1    2    3

> quantile(X, 0.6)

60%
 2

> resumen <- summary(X); resumen

      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0.0     1.0     2.0     2.1     3.0     5.0

> fivenum(X)

[1] 0 1 2 3 5

> barplot(tfre[[2]], main="Gráfico de barras", xlab="X = Número Hijos\n", ylab="frecuencia",
+ col=c("yellow", "blue", "white", "orange", "cyan", "red"), sub="Agosto-2012")
> pie(tfre[[2]], main="Gráfico de pastel", xlab="Número Hijos\n", col=c("yellow", "blue",
+ "white", "orange", "cyan", "red"), sub="Agosto-2012")
> names(fab) = c("Cero", "Uno", "Dos", "Tres", "Cuatro", "Cinco")
> pie(fab, main="Gráfico de pastel", xlab="X = Número Hijos\n", col=c("yellow", "blue",
+ "white", "orange", "cyan", "red"), sub="Agosto-2012")
> boxplot(X, main="Gráfico de caja", ylab="Número de hijos\n")
> boxplot(X, main="Gráfico de caja", xlab=" Número de hijos\n", plot=TRUE, border="red", col="red",
+
+
+

```

Practica 8.

```

Notas<-c(4.47,4.47,3.48,5.0,3.42,3.78,3.1,3.57,4.2,4.5,3.6,3.75,4.5,2.85,3.7,4.2,3.2,4.05,4.9,5.1,5.3,4.16,4.56,3.
data.entry(Notas)
write(Notas, "Notas.txt")
ls() rm(list=ls(all=TRUE)) ls()
X <- scan("Notas.txt", what = double(0), na.strings = "NA", flush=FALSE)
ls()

n <- length(X); n k <- 1+3.322*logb(60, 10); k k <- round(k); k rango <-
max(X)-min(X); rango a=rango/k; a a <- round(a, 3); a
Calcula el ancho o amplitud a de cada intervalo a=rango/k rango <- max(X)-
min(X); rango a=rango/k; a a <- round(a, 3); a
Define los límites y puntos medios de cada uno de los k intervalos limites <-
seq(from=min(X)-0.01/2, to=max(X)+0.01/2, by=a); limites options(digits=4)
ci <- cbind(1:k); ci for(i in 2:length(limites)) ci[i-1, 1] <- (limites[i] + limites[i-
1])/2 ci
Encuentra las frecuencias absolutas fi para cada intervalo

```

```

options(digits=2) fi <- cbind(table(cut(X, breaks = limites, labels=NULL,
include.lowest=FALSE, right=FALSE, dig.lab=4))); fi
Encuentra las frecuencias relativas o proporciones fri
options(digits=4) fri <- fi/n; fri
Encuentra las frecuencias acumuladas ascendentes Fi options(digits=2) Fi
<- cumsum(fi); Fi
Encuentra las frecuencias relativas acumuladas Fri options(digits=4) Fri <-
Fi/n; Fri
Completa la tabla de frecuencias. tablaFrec <- data.frame(ci=ci, fi=fi,
fri=fri, Fi=Fi, Fri=Fri); tablaFrec
h <- hist(X, breaks=c(limites[1]-a, limites, limites[k+1]+a), freq = TRUE,
probability = FALSE, include.lowest = FALSE, right = TRUE, main = "His-
tograma de frecuencias", col="lightyellow", lty=1, border="purple", xlab=" No-
tas de aspirantes", ylab="Frecuencia (fi)", axes=TRUE, labels=FALSE) text(hmids, hdensity,
hcounts, adj = c(0.5, -0.5), col = "red") rug(jitter(X)) adicionamarcasdelosdatos
h es un objeto del tipo lista que contiene atributos del histograma is.list(h);
h
h <- hist(X, breaks=c(limites[1]-a, limites, limites[k+1]+a), freq = FALSE,
probability = TRUE, include.lowest = FALSE, right = TRUE, main="AproximaciÃ³n
a una Normal", col="lightyellow", lty=1, border="purple", xlab="Notas de aspi-
rantes", ylab="Frecuencia relativa (fri)", axes=TRUE, labels=FALSE) text(hmids, hdensity,
hcounts, adj = c(0.5, 0.2), col = "red") rug(jitter(X)) adicionamarcasdelosdatos curve(dnorm(x, mean =
mean(X), sd = sd(X)), col = 2, lty = 2, lwd = 2, add = TRUE)
Crea el polÃgono de frecuencias h <- hist(X, breaks=c(limites[1]-a, limites,
limites[k+1]+a), freq = TRUE, probability=FALSE, include.lowest=FALSE, right=TRUE,
main = "PolÃgono de frecuencias", col="lightyellow", lty=1, border="purple",
xlab=" Notas de aspirantes", ylab="Frecuencia (fi)", axes=TRUE, labels=FALSE)
text(hmids, hdensity, hcounts, adj = c(0.5, -0.5), col = "red") rug(jitter(X)) adicionamarcasdelosdatos vCi <-
-c(hmids[1]-a, hmids, hmids[k+1]+a); vCi vfi <- c(0, hcounts, 0); vfilines(vCi, vfi, col =
"blue", type = "l")
Crea la Ojiva ascendente o polÃgono de frecuencias acumuladas ascen-
dentes Fia <- c(0, Fi); Fia plot(limites, Fia, type = "p", pch=1, col = "blue",
main="Ojiva ascendente", xlab="Notas de aspirantes", ylab="Frecuencia acu-
mulada (Fi)") text(limites, hdensity, Fia, adj = c(0.5, -0.5), col = "red") lines(limites, Fia, col =
"black", type = "l")
Calcula los principales estadÃsticos descriptivos de la variable Calcula la
moda, ya que el R no proporciona una funciÃ³n para eso. options(digits=4)
for(i in 1:k) if (fi[i] == max(fi)) break() if(i > 1) moda <- limites[i]+((fi[i]-
fi[i-1])/((fi[i]-fi[i-1])+(fi[i]-fi[i+1])))*a else moda <- limites[i]+(fi[i]/(fi[i]+(fi[i]-
fi[i+1])))*a moda
Calcula los cuartiles: Q1, Q2, Q3 Q <- 1:3 for(v in 1:3) for(i in 1:k) if (Fi[i]
> (v*25*n)/100) Q[v] <- limites[i]+(((25*v*n)/100)-Fi[i-1])/fi[i])*a break Q
Calcula los principales estadÃsticos. estadisticos <- rbind(media=sum(tabEstadci fi)/n, moda =
moda, Q1 = Q[1], Q2 = Q[2], Q3 = Q[3], rango = max(X)-min(X), varianza =
sum(tabEstadciMedia2fi)/n, Desviacion=sqrt(sum(tabEstadciMedia2fi)/n), CoeficienteVariacion =

```

$\sqrt{\text{sum}(\text{tabEstadciMedia2fi})/n} / (\text{sum}(\text{tabEstadci}fi)/n), CAfisher = (\text{sum}(\text{tabEstadciMedia3fi})/n) / \sqrt{\text{sum}(\text{sum}(\text{tabEstadciMedia4fi})/n) / \sqrt{\text{sum}(\text{tabEstadciMedia2fi})/n}^4 - 3) estadisticos$

Gráfico de cajas `boxplot(X, main="Gráfico de caja", xlab="Notas", notch=FALSE, data=parent.frame(), plot=TRUE, border="red", col="yellow", horizontal=TRUE)`
 Observación: en la función `boxplot()`, si `plot` es `FALSE` se produce un resumen de los valores (los cinco números).

`windows() boxplot(X, main="Gráfico de caja", xlab="X = Notas", notch=TRUE, data=parent.frame(), plot=TRUE, border="red", col="yellow", horizontal=TRUE)`
`par(mfrow=c(1,2))` Divide la ventana gráfica en dos partes (1 fila, 2 columnas)
`mtext(side=3, line=0, cex=2, outer=T, "Titulo para Toda la Página")`
`hist(X); boxplot(X)`

Calcula los principales estadísticos descriptivos de la variable. Calcula la moda, ya que el R no proporciona una función para eso.
`options(digits=4)`
`for(i in 1:k) if (fi[i] == max(fi)) break() if(i > 1) moda <- limites[i] + ((fi[i]-fi[i-1]) / ((fi[i]-fi[i-1]) + (fi[i]-fi[i+1])))) * a`
`moda <- limites[i] + (fi[i] / (fi[i] + (fi[i]-fi[i+1]))) * a`
`moda`

Varios gráficos en una misma ventana `par(mfrow=c(1,2))` Divide la ventana gráfica en dos partes (1 fila, 2 columnas)
`mtext(side=3, line=0, cex=2, outer=T, "Titulo para Toda la Página")`
`hist(X); boxplot(X)`

Practica 9.

`library(foreign)` `HojaCat <- read.table("HojaCat.txt", header=TRUE)` `HojaCat`

Conecta la hoja de datos a la segunda ruta o lista de búsqueda.
`attach(HojaCat, pos=2)` `pos` especifica la posición donde buscar la conexión
`search()`

Crea una tabla de contingencia o de doble entrada `tablaCont <- table(HojaCat)`
`tablaCont` `length(HojaCat)`

Encuentra la suma de cada fila de la tabla de contingencia. Distribución marginal de `X=Estado civil`
`suma.filas <- apply(tablaCont, 1, sum)`
`suma.filas`
 El 1 indica que son totales por fila

Gráficos de barras para tabla de contingencia. Barras apiladas `barplot(t(tablaCont), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil", ylab="Ocupación", legend.text=TRUE)`

`barplot(t(tablaCont), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil", ylab="Ocupación", beside=TRUE, legend.text=TRUE)`

Guardar las todas las opciones iniciales y modificar número de decimales
`options(digits=3)` solo imprime 3 lugares decimales `options('digits')`

Proporciones basadas en el total de la muestra, la suma de filas y columnas suman 1

`propTotal <- prop.table(tablaCont)`
`propTotal`
`barplot(t(propTotal), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil", ylab="Ocupación", beside=TRUE, legend.text=TRUE)`

Proporciones basadas en el total por fila, cada fila suma 1.

`propFila <- prop.table(tablaCont, 1)`
`propFila` Total por fila se indica en 1
`barplot(t(propFila), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil", ylab="Ocupación", beside=TRUE, legend.text=TRUE)`

```
propFila <- prop.table(tablaCont, 1); propFila
# Total por fila se indica en 1
barplot(t(propFila), main="Gráfico de barras (Estado, Ocupación)",
xlab="Estado civil", ylab="Ocupación", beside=TRUE, legend.text=TRUE)
# Realizar la prueba o contraste Chi-cuadrado de independencia
prueba <- chisq.test(tablaCont); prueba
# Frecuencias absolutas esperadas para la prueba Chi-cuadrada
prueba$expectedfij = fi./No.column
```

Practica 10.

```
> A <- c(100,96,92,96,92); A
[1] 100 96 92 96 92
> B <- c(76,80,75,84,82); B
[1] 76 80 75 84 82
> C <- c(108,100,96,98,100); C
[1] 108 100 96 98 100
> Baterias <- data.frame(procesoA=A, procesoB=B, procesoC=C); Baterias
  procesoA procesoB procesoC
1      100       76      108
2       96       80      100
3       92       75       96
4       96       84       98
5       92       82      100
> # Para editar los datos puede utilizar la función fix()
> fix(Baterias)
> write.table(Baterias, file="Baterias.txt", append=FALSE, quote=TRUE, sep=" ", na="NA",
+ col.names=TRUE)
> ls(); rm(list=ls(all=TRUE)); ls()

[1] "A"          "B"          "Baterias"   "C"          "cuasivar"   "fab"
[7] "Fac"        "Far"        "fre"        "i"          "media"      "mediana"
[13] "moda"       "resumen"    "s"          "tabla"      "tfre"       "X"
character(0)
> Baterias <- read.table("Baterias.txt", header=TRUE); Baterias
  procesoA procesoB procesoC
1      100       76      108
2       96       80      100
3       92       75       96
4       96       84       98
5       92       82      100
```



```

> attach(Baterias, pos=2)
> search()

[1] ".GlobalEnv"          "Baterias"            "package:RcmdrMisc"
[4] "package:sandwich"    "package:car"         "package:splines"
[7] "package:stats"       "package:graphics"    "package:grDevices"
[10] "package:utils"       "package:datasets"    "package:methods"
[13] "Autoloads"          "package:relimp"      "package:base"

> stripchart(Baterias, main="Gr?fico de puntos para los tres procesos", method = "stack", ve
+ FALSE, col="blue", pch=1, xlab="Duraci?n (semanas)", ylab="Proceso")
> #Muestra un resumen estad?stico para los tres procesos.
> summary(Baterias)

      procesoA      procesoB      procesoC
Min.   : 92   Min.   :75   Min.   : 96
1st Qu.: 92   1st Qu.:76   1st Qu.: 98
Median : 96   Median :80   Median :100
Mean    : 95   Mean    :79   Mean    :100
3rd Qu.: 96   3rd Qu.:82   3rd Qu.:100
Max.    :100   Max.    :84   Max.    :108

> # Horizontal
> boxplot(Baterias, width=NULL, varwidth=TRUE, names, add= FALSE, horizontal = TRUE,
+ main="Gr?fico de caja por proceso", border=par("fg"), col=c("yellow", "cyan", "red"), xlab
+ "Duraci?n (semanas)", ylab="Proceso")
> # Vertical
> boxplot(Baterias, width=NULL, varwidth=TRUE, names, add= FALSE, horizontal = FALSE,
+ main="Gr?fico de caja por proceso", border=par("fg"), col=c("yellow", "cyan", "red"), xlab
+ "Duraci?n (semanas)", ylab="Proceso")
> #Presenta la matriz de covarianzas muestral.
> options(digits=3) # s?lo imprime 3 lugares decimales
> S <- var(Baterias); S

      procesoA procesoB procesoC
procesoA    11.2    -1.6    12.4
procesoB    -1.6    14.8    -4.7
procesoC    12.4    -4.7    20.8

> Baterias <- stack(Baterias); Baterias

      values      ind
1      100 procesoA
2       96 procesoA
3       92 procesoA
4       96 procesoA
5       92 procesoA

```

```

6      76 procesoB
7      80 procesoB
8      75 procesoB
9      84 procesoB
10     82 procesoB
11    108 procesoC
12    100 procesoC
13     96 procesoC
14     98 procesoC
15    100 procesoC

> names(Baterias) # Muestra los encabezados de los vectores

[1] "values" "ind"

> #Desconecta la hoja de datos de la segunda ruta o lista de b?squeda.
> detach(Baterias, pos=2); search()

[1] ".GlobalEnv"      "package:RcmdrMisc" "package:sandwich"
[4] "package:car"      "package:splines"   "package:stats"
[7] "package:graphics" "package:grDevices" "package:utils"
[10] "package:datasets" "package:methods"   "Autoloads"
[13] "package:relimp"   "package:base"

> #An?lisis de una variable bidimensional
>
> Fuma = c("Si", "No", "No", "Si", "No", "Si", "Si", "Si", "No", "Si"); Fuma

[1] "Si" "No" "No" "Si" "No" "Si" "Si" "Si" "No" "Si"

> Cantidad = c(1,2,2,3,3,1,2,1,3,2); Cantidad

[1] 1 2 2 3 3 1 2 1 3 2

> Estudia <- data.frame(Fuma=Fuma, Cantidad=Cantidad); Estudia

  Fuma Cantidad
1   Si         1
2   No         2
3   No         2
4   Si         3
5   No         3
6   Si         1
7   Si         2
8   Si         1
9   No         3
10  Si         2

```

```

> fix(Estudia)
> write.table(Estudia, file="Estudia.txt", append=FALSE, quote=TRUE, sep=" ", na="NA",
+ col.names=TRUE)
> write.table

function (x, file = "", append = FALSE, quote = TRUE, sep = " ",
  eol = "\n", na = "NA", dec = ".", row.names = TRUE, col.names = TRUE,
  qmethod = c("escape", "double"), fileEncoding = "")
{
  qmethod <- match.arg(qmethod)
  if (is.logical(quote) && (length(quote) != 1L || is.na(quote)))
    stop("'quote' must be 'TRUE', 'FALSE' or numeric")
  quoteC <- if (is.logical(quote))
    quote
  else TRUE
  qset <- is.logical(quote) && quote
  if (!is.data.frame(x) && !is.matrix(x))
    x <- data.frame(x)
  makeRownames <- isTRUE(row.names)
  makeColnames <- is.logical(col.names) && !identical(FALSE,
    col.names)
  if (is.matrix(x)) {
    p <- ncol(x)
    d <- dimnames(x)
    if (is.null(d))
      d <- list(NULL, NULL)
    if (is.null(d[[1L]]) && makeRownames)
      d[[1L]] <- seq_len(nrow(x))
    if (is.null(d[[2L]]) && makeColnames && p > 0L)
      d[[2L]] <- paste0("V", 1L:p)
    if (qset)
      quote <- if (is.character(x))
        seq_len(p)
      else numeric()
  }
  else {
    if (qset)
      quote <- if (length(x))
        which(unlist(lapply(x, function(x) is.character(x) ||
          is.factor(x))))
      else numeric()
    if (any(sapply(x, function(z) length(dim(z)) == 2 &&
      dim(z)[2L] > 1))) {
      c1 <- names(x)
      x <- as.matrix(x, rownames.force = makeRownames)
      d <- dimnames(x)

```

```

        if (qset) {
            ord <- match(c1, d[[2L]], 0L)
            quote <- ord[quote]
            quote <- quote[quote > 0L]
        }
    }
    else d <- list(if (makeRownames) row.names(x), if (makeColnames) names(x))
    p <- ncol(x)
}
nocols <- p == 0L
if (is.logical(quote))
    quote <- NULL
else if (is.numeric(quote)) {
    if (any(quote < 1L | quote > p))
        stop("invalid numbers in 'quote'")
}
else stop("invalid 'quote' specification")
rn <- FALSE
rnames <- NULL
if (is.logical(row.names)) {
    if (row.names) {
        rnames <- as.character(d[[1L]])
        rn <- TRUE
    }
}
else {
    rnames <- as.character(row.names)
    rn <- TRUE
    if (length(rnames) != nrow(x))
        stop("invalid 'row.names' specification")
}
if (!is.null(quote) && rn)
    quote <- c(0, quote)
if (is.logical(col.names)) {
    if (!rn && is.na(col.names))
        stop("'col.names = NA' makes no sense when 'row.names = FALSE'")
    col.names <- if (is.na(col.names) && rn)
        c("", d[[2L]])
    else if (col.names)
        d[[2L]]
    else NULL
}
else {
    col.names <- as.character(col.names)
    if (length(col.names) != p)
        stop("invalid 'col.names' specification")
}

```

```

}
if (file == "")
  file <- stdout()
else if (is.character(file)) {
  file <- if (nzchar(fileEncoding))
    file(file, ifelse(append, "a", "w"), encoding = fileEncoding)
  else file(file, ifelse(append, "a", "w"))
  on.exit(close(file))
}
else if (!isOpen(file, "w")) {
  open(file, "w")
  on.exit(close(file))
}
if (!inherits(file, "connection"))
  stop("'file' must be a character string or connection")
qstring <- switch(qmethod, escape = "\\\"\\\"", double = "\"\"")
if (!is.null(col.names)) {
  if (append)
    warning("appending column names to file")
  if (quoteC)
    col.names <- paste("\"", gsub("\"", qstring, col.names),
      "\"", sep = "")
  writeLines(paste(col.names, collapse = sep), file, sep = eol)
}
if (nrow(x) == 0L)
  return(invisible())
if (ncols && !rn)
  return(cat(rep.int(eol, NROW(x)), file = file, sep = ""))
if (is.matrix(x) && !is.atomic(x))
  mode(x) <- "character"
if (is.data.frame(x)) {
  x[] <- lapply(x, function(z) {
    if (is.object(z) && !is.factor(z))
      as.character(z)
    else z
  })
}
invisible(.External2(C_writetable, x, file, nrow(x), p, rnames,
  sep, eol, na, dec, as.integer(quote), qmethod != "double"))
}
<bytecode: 0x00000000af82710>
<environment: namespace:utils>

> ls()

[1] "Baterias" "Cantidad" "Estudia"  "Fuma"    "S"

```

```

> rm(list=ls(all=TRUE))
> ls()

caracter(0)

> Estudia <- read.table("Estudia.txt", header=TRUE)
> Estudia

      Fuma Cantidad
1      Si         1
2      No         2
3      No         2
4      Si         3
5      No         3
6      Si         1
7      Si         2
8      Si         1
9      No         3
10     Si         2

> tablaCont <- table(Estudia)
> tablaCont

      Cantidad
Fuma 1 2 3
No 0 2 2
Si 3 2 1

> options(digits=3) # s?lo imprime 3 lugares decimales
> propTotal <- prop.table(tablaCont); propTotal

      Cantidad
Fuma   1   2   3
No 0.0 0.2 0.2
Si 0.3 0.2 0.1

> propFila <- prop.table(tablaCont, 1)
> propFila

      Cantidad
Fuma      1      2      3
No 0.000 0.500 0.500
Si 0.500 0.333 0.167

> propCol <- prop.table(tablaCont, 2)
> propCol

```

	Cantidad		
Fuma	1	2	3
No	0.000	0.500	0.667
Si	1.000	0.500	0.333

```
> barplot(table(Estudia$Cantidad, Estudia$Fuma), beside = FALSE, horizontal=FALSE, main="Gr?fico de barras (Cantidad de horas de estudio,Fuma)", legend.text =T, xlab="Cantidad de horas-es
+ de barras (Cantidad de horas de estudio,Fuma)", legend.text =T, xlab="Cantidad de horas-es
+ ylab="Fuma")
> Fuma=factor(Estudia$Fuma); Fuma
```

```
[1] Si No No Si No Si Si Si No Si
Levels: No Si
```

```
> barplot(table(Estudia$Cantidad, Estudia$Fuma), main="Gr?fico de barras (Fuma, Cantidad de
+ de estudio)", xlab="Fuma", ylab="Cantidad de horas-estudio", beside=TRUE, legend.text=T)
> barplot(table(Estudia$Cantidad, Estudia$Fuma), main="Gr?fico de barras (Fuma, Cantidad de
+ de estudio)", xlab="Fuma", ylab="Cantidad de horas-estudio", beside=TRUE, legend.text=c("
+ que 5", "5-10", "mayor que 10"))
>
>
>
```

Practica 11.

```
> usuarios <- c(10, 15, 20, 20, 25, 30, 30); usuarios
```

```
[1] 10 15 20 20 25 30 30
```

```
> tiempo = c(1.0, 1.2, 2.0, 2.1, 2.2, 2.0, 1.9); tiempo
```

```
[1] 1.0 1.2 2.0 2.1 2.2 2.0 1.9
```

```
> Sistema <- data.frame(Usuarios=usuarios, Tiempo=tiempo);Sistema
```

	Usuarios	Tiempo
1	10	1.0
2	15	1.2
3	20	2.0
4	20	2.1
5	25	2.2
6	30	2.0
7	30	1.9

```
> fix(Sistema)
> write.table(Sistema, file="Sistema.txt", append=FALSE, quote=TRUE, sep=" ", na="NA",
+ col.names = TRUE)
> ls(); rm(list=ls(all=TRUE)); ls()
```

```

[1] "Estudia"    "Fuma"       "propCol"    "propFila"   "propTotal" "Sistema"
[7] "tablaCont" "tiempo"     "usuarios"

character(0)

> #Recupera la hoja de datos.
> Sistema <- read.table("Sistema.txt", header=TRUE); Sistema

  Usuarios Tiempo
1       10    1.0
2       15    1.2
3       20    2.0
4       20    2.1
5       25    2.2
6       30    2.0
7       30    1.9

> #Conecta la hoja de datos a la segunda ruta o lista de búsqueda.
> attach(Sistema, pos=2); search()

[1] ".GlobalEnv"      "Sistema"        "package:RcmdrMisc"
[4] "package:sandwich" "package:car"     "package:splines"
[7] "package:stats"    "package:graphics" "package:grDevices"
[10] "package:utils"    "package:datasets" "package:methods"
[13] "Autoloads"       "package:relimp"  "package:base"

> #Muestra un resumen de principales estadísticos de las variables.
> summary(Sistema)

  Usuarios      Tiempo
Min.   :10.0   Min.    :1.00
1st Qu.:17.5   1st Qu.:1.55
Median :20.0   Median  :2.00
Mean   :21.4   Mean    :1.77
3rd Qu.:27.5   3rd Qu.:2.05
Max.   :30.0   Max.    :2.20

> cov(Sistema) # Matriz de covarianzas

      Usuarios Tiempo
Usuarios  55.95  2.714
Tiempo    2.71  0.222

> cor(Sistema, use = "all.obs", method="pearson") # Matriz de correlaciones

      Usuarios Tiempo
Usuarios  1.000  0.769
Tiempo    0.769  1.000

```



```

> #Elabora un gráfico de dispersión para analizar alguna relación entre las variables.
> plot(Usuarios, Tiempo, xlim= c(5, 35), ylim= c(0.0, 2.5), type = "p", pch=1, col = "blue",
+ "Gráfico de dispersión (Usuarios, Tiempo)", xlab="Número de usuarios", ylab="Tiempo de
+ ejecución")
> #Sin cerrar la ventana del gráfico anterior, ejecuta la siguiente instrucción
> identify(Usuarios, Tiempo, n=1) # n=1 indica que solamente será un punto seleccionado

integer(0)

> reg.Y.X <- lm(Tiempo ~ -1 + Usuarios, Sistema, na.action=NULL, method="qr", model=TRUE)
> #-1 indica que no se toma en cuenta la constante en el modelo.
> summary(reg.Y.X)

Call:
lm(formula = Tiempo ~ -1 + Usuarios, data = Sistema, na.action = NULL,
    method = "qr", model = TRUE)

Residuals:
    Min       1Q   Median       3Q      Max
-0.483 -0.187  0.206  0.313  0.511

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
Usuarios    0.0794     0.0065    12.2 1.8e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.387 on 6 degrees of freedom
Multiple R-squared:  0.961,    Adjusted R-squared:  0.955
F-statistic: 150 on 1 and 6 DF,  p-value: 1.82e-05

> lines(Usuarios, 0.079437*Usuarios)
> reg.anova <- anova(reg.Y.X); reg.anova

Analysis of Variance Table

Response: Tiempo
      Df Sum Sq Mean Sq F value    Pr(>F)
Usuarios  1   22.4   22.40    150 1.8e-05 ***
Residuals  6    0.9    0.15
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

>
>
>

```

Practica 13.

```

> moneda <- c("C", "+"); moneda

[1] "C" "+"

> n <- 10; n

[1] 10

> espacio <- 1:54;espacio

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
[26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
[51] 51 52 53 54

> # se define el tamaño de la muestra
> n <- 6; n

[1] 6

> muestra <- sample(espacio, n); muestra

[1] 47 23 42 40 33 1

> # genera el espacio muestral del lanzamiento de los dos dados
> espacio = as.vector(outer(1:6, 1:6, paste)); espacio

[1] "1 1" "2 1" "3 1" "4 1" "5 1" "6 1" "1 2" "2 2" "3 2" "4 2" "5 2" "6 2"
[13] "1 3" "2 3" "3 3" "4 3" "5 3" "6 3" "1 4" "2 4" "3 4" "4 4" "5 4" "6 4"
[25] "1 5" "2 5" "3 5" "4 5" "5 5" "6 5" "1 6" "2 6" "3 6" "4 6" "5 6" "6 6"

> # se define el tamaño de la muestra
> n <- 4; n

[1] 4

> # finalmente se selecciona la muestra
> muestra <- sample(espacio, n, replace=TRUE); muestra

[1] "1 2" "5 1" "3 3" "6 6"

> #genera el espacio muestral de las 52 cartas
> naipe = paste(rep(c("A", 2:10, "J", "Q", "K"), 4), c("OROS", "COPAS", "BASTOS",
+ "ESPADAS"));naipe

[1] "A OROS" "2 COPAS" "3 BASTOS" "4 ESPADAS" "5 OROS"
[6] "6 COPAS" "7 BASTOS" "8 ESPADAS" "9 OROS" "10 COPAS"
[11] "J BASTOS" "Q ESPADAS" "K OROS" "A COPAS" "2 BASTOS"
[16] "3 ESPADAS" "4 OROS" "5 COPAS" "6 BASTOS" "7 ESPADAS"
[21] "8 OROS" "9 COPAS" "10 BASTOS" "J ESPADAS" "Q OROS"
[26] "K COPAS" "A BASTOS" "2 ESPADAS" "3 OROS" "4 COPAS"

```

```

[31] "5 BASTOS"      "6 ESPADAS"    "7 OROS"       "8 COPAS"      "9 BASTOS"
[36] "10 ESPADAS"    "J OROS"       "Q COPAS"      "K BASTOS"     "A ESPADAS"
[41] "2 OROS"        "3 COPAS"      "4 BASTOS"     "5 ESPADAS"    "6 OROS"
[46] "7 COPAS"       "8 BASTOS"     "9 ESPADAS"    "10 OROS"      "J COPAS"
[51] "Q BASTOS"      "K ESPADAS"

> # se define el tamaño de la muestra
> n <- 5; n

[1] 5

> # se obtiene la muestra sin reemplazo (aunque no se especifique con replace=FALSE)
> cartas <- sample(naipes, n) ; cartas

[1] "2 BASTOS" "4 COPAS"  "6 COPAS"  "A COPAS"  "J BASTOS"

> espacio <- function(num)
+ {
+   numDiv7 <- numeric(0)
+   ind <- 0
+   for(i in 1:length(num))
+     if ((num[i] %% 7)==0)
+     {
+       ind <- ind+1
+       numDiv7[ind]=num[i]
+     }
+   return(numDiv7)
+ }
> numeros <- 1:500
> # generando el espacio muestral
> s <- espacio(numeros); s

[1] 7 14 21 28 35 42 49 56 63 70 77 84 91 98 105 112 119 126 133
[20] 140 147 154 161 168 175 182 189 196 203 210 217 224 231 238 245 252 259 266
[39] 273 280 287 294 301 308 315 322 329 336 343 350 357 364 371 378 385 392 399
[58] 406 413 420 427 434 441 448 455 462 469 476 483 490 497

> # seleccionando la muestra
> muestra <- sample(s, 12, replace=TRUE); muestra

[1] 70 189 91 203 259 280 21 105 161 350 273 462

>
>
>
>
>
>
>

```

Practica 14.

```
> dbinom(4,8,0.5)

[1] 0.273

> x <- 2; n=8; p=1/2
> pbinom(x, size = n, prob = p, lower.tail=TRUE)

[1] 0.145

> x <- 4; n=8; p=1/2
> #primera forma
> F <- 1 - pbinom(x, n, p, lower.tail=TRUE); F

[1] 0.363

> #segunda forma
> pbinom(4, size=8, prob=0.5, lower.tail=FALSE)

[1] 0.363

> x <- 3; mu <- 6
> ppois(x, lambda = mu, lower.tail=TRUE)

[1] 0.151

> #primera forma
>
> sum(dpois(c(6,7,8),lambda = 6))

[1] 0.402

> # segunda forma
> F8 <- ppois(8, lambda = 6, lower.tail=TRUE)
> F5 <- ppois(5,lambda = 6, lower.tail=TRUE)
> F8 - F5

[1] 0.402

> n <- 30
> #genera 30 valores de una distribución de Poisson con  $\lambda = 6$ 
> x <- rpois(n, lambda=mu)
> #calcula las probabilidades para cada valor generado
>
>
> y <- dpois(x, lambda=mu)
> #genera el gráfico de distribución
> plot(x, y, xlab="x", ylab="Función de probabilidad", main="Distribución de Poisson: lambda")
```

```

+ type="h")
> #une los puntos a las líneas
> points(x, y, pch=21)
> x <- 0:2
> m = 11
> n <- 4; k=2
> # x define el número de globos con premio
>
>
>
>

```

Practica 15 y 16.

```

> #ejemplo 1
> x <-55
> a=0
> b<-90
> punif(x,min=a,max=b,lower.tail=TRUE)

[1] 0.611

> ## [1] 0.6111111
> F55=punif(55,min=a,max=b,lower.tail=TRUE)
> F15=punif(15,min=a,max=b,lower.tail=TRUE)
> F55-F15

[1] 0.444

> ## [1] 0.4444444
> (1-F55)*( F55-F15)

[1] 0.173

> ## [1] 0.1728395
> #ejemplo 2
> p<-c(0.80)
> media=5;
> d.t=1
> qnorm(p, mean=media,sd=d.t,lower.tail=TRUE)

[1] 5.84

> ## [1] 5.841621
> p<-c(0.80)
> g.l<-10
> qt(p,df=g.l,lower.tail=TRUE)

[1] 0.879

```

```

> ## [1] 0.8790578
> n<-16
> x<-4.5
> mu=5
> sigma=1
> d.t=sigma/sqrt(n)
> pnorm(x,mean=mu,sd=d.t,lower.tail=FALSE)

[1] 0.977

> ## [1] 0.9772499
> #ejemplo 3
> x<-5
> teta=7
> pexp(x,rate=1/teta,lower.tail=FALSE)

[1] 0.49

> ## [1] 0.4895417
> x<-3
> teta=7
> pexp(x,rate=1/teta,lower.tail=TRUE)

[1] 0.349

> ## [1] 0.3485609
> pexp(4,rate=1/teta,lower.tail=FALSE)

[1] 0.565

> ## [1] 0.5647181
> p<-0.9
> teta<-7
> qexp(p,rate=1/teta,lower.tail=TRUE)

[1] 16.1

> ## [1] 16.1181
> qexp(0.5,rate=1/teta,lower.tail=TRUE)

[1] 4.85

> ## [1] 4.85203
> qexp(0.68,rate=1/teta,lower.tail=TRUE)

[1] 7.98

> ## [1] 7.97604
> qexp(0.32,rate=1/teta,lower.tail=FALSE)

```

```

[1] 7.98

> ## [1] 7.97604
>
> #3. gneracion de muestras aleatorias de las distribuciones
> #ejemplo 1
> min<--2
> max<-4
> x=runif(100,min,max)
> x

[1] 0.09281 0.90708 -1.74780 -1.53806 -0.56655 2.36463 3.85881 1.47281
[9] -0.62901 1.63006 0.03023 0.67599 3.94219 3.50579 0.72304 3.51455
[17] 1.05715 -1.64025 3.56361 -1.14833 0.76841 -0.61599 -1.92115 -1.73778
[25] 0.87761 -0.84723 3.12214 -0.47305 3.64834 -1.46681 3.56717 -0.64985
[33] 2.46512 -0.17994 2.42383 -1.08043 0.05757 -0.51051 3.32363 2.85467
[41] -1.37410 0.26612 1.61592 2.35435 -1.88999 -1.00243 -0.68710 2.27279
[49] 0.87760 2.31962 2.71081 2.85696 -1.91322 2.14857 0.72150 -1.76869
[57] 3.09602 2.39829 1.89095 -0.68058 -0.50968 -1.23135 -0.81127 -1.59447
[65] 0.50505 0.00895 -0.98713 -1.57089 1.02366 0.92767 -0.93465 0.77132
[73] 1.80557 0.17802 2.34470 -1.32995 -0.94281 0.54813 2.67076 -0.82417
[81] -0.17639 0.79446 3.25581 1.75195 0.25800 0.58062 -0.98158 3.55403
[89] 1.27279 0.18575 -0.78087 -1.98356 3.90709 0.14140 0.85519 1.75686
[97] -1.60973 -1.80033 2.17625 1.02154

> hist(x,main="X ~Uniforme(min=-2,max=4",xlab="X",ylab="densidad de probabilidad",
+ probability=TRUE,col="cyan")
> curve(dunif(x,min,max),col="blue",add=TRUE)
> #ejemplo 2
> x.norm<-rnorm(n=200,mean=10,sd=2)
> x.norm

[1] 11.45 12.25 10.57 9.72 11.95 8.50 9.97 8.40 7.62 10.11 9.09 10.16
[13] 12.76 12.55 9.10 8.15 7.97 8.22 11.47 11.27 9.32 13.16 11.67 8.14
[25] 9.36 10.73 8.42 7.73 9.50 10.01 9.82 8.17 7.41 6.80 8.68 11.39
[37] 10.51 12.70 10.44 7.77 13.72 8.97 14.73 11.51 9.55 9.56 9.77 11.12
[49] 9.49 11.43 10.43 10.77 11.70 7.13 9.93 10.98 9.21 8.84 9.03 15.19
[61] 6.22 9.63 9.88 8.13 9.45 7.48 11.51 13.59 11.29 9.71 8.36 10.23
[73] 7.87 8.90 9.42 11.63 8.13 8.58 7.85 11.51 12.49 7.50 9.26 12.00
[85] 7.86 7.71 12.58 8.43 8.56 9.39 11.47 6.74 11.56 11.41 11.20 12.62
[97] 9.06 10.00 6.96 10.36 12.36 8.63 12.51 11.51 9.68 14.98 11.01 7.65
[109] 13.60 9.80 10.26 9.75 14.45 7.80 11.86 8.95 9.23 11.19 10.83 9.82
[121] 8.96 10.63 14.62 8.17 13.01 9.03 10.00 7.37 8.63 10.26 6.98 11.53
[133] 7.70 7.98 9.61 11.57 5.82 9.59 8.68 14.73 6.70 7.30 12.94 11.16
[145] 11.13 8.97 10.37 11.21 9.19 9.21 7.17 9.27 10.69 10.79 11.95 9.51
[157] 9.35 8.11 9.06 8.70 11.35 13.25 8.23 12.05 13.56 12.51 8.61 8.32
[169] 14.70 11.23 10.71 8.49 8.11 8.02 6.25 6.69 9.84 8.70 9.11 8.39

```

```
[181] 11.02  7.97  7.81 10.39 11.02  7.39 12.57  9.02  8.82 11.55  8.95  7.68
[193] 12.97  7.88 10.22  7.18  9.84 13.88  9.76 10.34
```

```
> hist(x.norm,breaks="Sturges",freq=TRUE,probability=FALSE,include.lowest=TRUE,right
+ =TRUE,density=NULL,angle=45,col="steelblue1",border = NULL, main = "Histograma de
+ datos observados",axes=TRUE,plot=TRUE,labels=FALSE)
> plot(ecdf(x.norm),main="Funcion de distribucion acumulada teorica")
```

Practica 17.

este es un ejemplo

```
> simulIntProp <- function(m=5, n=1, p, nivel.conf=0.95)
+ {
+ X <- rbinom(m, n, p)
+ # Matriz con 1000 valores aleatorios binomial(n,p), 50 muestras cada una de tama?o 20
+ pe <- X/n
+ # Calcula la proporci?n estimada en cada una de las muestras.
+ SE <- sqrt(pe*(1-pe)/n)
+ # Calcula la desviaci?n est?ndar estimada en cada una de las muestras.
+ alfa <- 1-nivel.conf
+ z <- qnorm(1-alfa/2)
+ Intervalo <- cbind(pe - z*SE, pe + z*SE)
+ # genera los extremos del intervalo de confianza
+ nInter <- 0
+ # un contador para conocer en cu?ntos intervalos se encuentra la verdadera proporci?n.
+ for(i in 1:m)
+ if ((p >= Intervalo[i, 1]) && (p <= Intervalo[i, 2]))
+ nInter <- nInter + 1
+ # funci?n que cuenta cu?ntos intervalos contienen el verdadero valor del par?metro.
+ return(nInter)
+ }
> n=20; m= 50; p=0.5; nivel.conf=0.95
> simulIntProp(m, n, p, nivel.conf)
```

```
[1] 46
```

```
> Intervalo # para visualizar cada uno de los intervalos generados
```

```
      [,1] [,2]
[1,] 0.3853 0.815
[2,] 0.3320 0.768
[3,] 0.2809 0.719
[4,] 0.2809 0.719
[5,] 0.4992 0.901
[6,] 0.3320 0.768
[7,] 0.2809 0.719
[8,] 0.0602 0.440
```



```

[9,] 0.0247 0.375
[10,] 0.3320 0.768
[11,] 0.1853 0.615
[12,] 0.4410 0.859
[13,] 0.2809 0.719
[14,] 0.1853 0.615
[15,] 0.3320 0.768
[16,] 0.2320 0.668
[17,] 0.1853 0.615
[18,] 0.3320 0.768
[19,] 0.2809 0.719
[20,] 0.4410 0.859
[21,] 0.2320 0.668
[22,] 0.3853 0.815
[23,] 0.3320 0.768
[24,] 0.4992 0.901
[25,] 0.2320 0.668
[26,] 0.1853 0.615
[27,] 0.4410 0.859
[28,] 0.3853 0.815
[29,] 0.2320 0.668
[30,] 0.3853 0.815
[31,] 0.6247 0.975
[32,] 0.3853 0.815
[33,] 0.4410 0.859
[34,] 0.2809 0.719
[35,] 0.3853 0.815
[36,] 0.3320 0.768
[37,] 0.3320 0.768
[38,] 0.2809 0.719
[39,] 0.2809 0.719
[40,] 0.4410 0.859
[41,] 0.0992 0.501
[42,] 0.3853 0.815
[43,] 0.3853 0.815
[44,] 0.1853 0.615
[45,] 0.4410 0.859
[46,] 0.3320 0.768
[47,] 0.2320 0.668
[48,] 0.2320 0.668
[49,] 0.0602 0.440
[50,] 0.3853 0.815

```

```
> nInter
```

```
[1] 46
```

```
> # para visualizar en cuantos de estos intervalos se encuentra la verdadera proporci?n.
> #Gráfico que muestra los intervalos de confianza de 95% que contienen y no contienen el v
> matplot(rbind(pe - z*SE, pe + z*SE), rbind(1:m, 1:m), type="l", lty=1)
> abline(v=p)
```

Practica 18.

```
> intervaloProp <- function(x, n, nivel.conf=0.95)
+ {
+ pe <- x/n
+ alfa <- 1-nivel.conf
+ z <- qnorm(1-alfa/2)
+ SE <- sqrt(pe*(1-pe)/n)
+ print(rbind(pe, alfa, z, SE))
+ LInf <- pe-z*SE
+ LSup <- pe+z*SE
+ print(" ")
+ print(paste("Intervalo para p es: [", round(LInf, 2), ",", round(LSup, 2), "]"))
+ }
> x=360; n=1200; nivel.conf=0.95
> intervaloProp(x, n, nivel.conf)

      [,1]
pe  0.3000
alfa 0.0500
z    1.9600
SE   0.0132
[1] " "
[1] "Intervalo para p es: [ 0.27 , 0.33 ]"
```

Practica 21. article

PRUEBAS DE NORMALIDAD DE UNA MUESTRA

Se digitan los datos del grupo de control

```
> IMC_Control <- c(23.6, 22.7, 21.2, 21.7, 20.7, 22.0, 21.8, 24.2, 20.1, 21.3,
+                 20.5, 21.1, 21.4, 22.2, 22.6, 20.4, 23.3, 24.8)
> par(mfrow=c(1,2))
```

Se genera el histograma de la variables de interÃls

```
> hist(IMC_Control,main="A",xlab="IMC (kg/m2)",ylab="Frecuencia")
```

Se genera el diagrama de caja de la variable de interÃls y se muestra en la misma ventana

```
> boxplot(IMC_Control,main="B", lab="IMC (kg/m2)",ylim=c(20,25))
```

Los comandos para contrastar normalidad son los siguientes

```
> sw <- shapiro.test(IMC_Control)
> sw
```

Shapiro-Wilk normality test

```
data: IMC_Control
W = 1, p-value = 0.5
```

```
> ks <- ks.test(IMC_Control,"pnorm",mean=mean(IMC_Control),sd=sd(IMC_Control))
> ks
```

One-sample Kolmogorov-Smirnov test

```
data: IMC_Control
D = 0.1, p-value = 1
alternative hypothesis: two-sided
```

Luego se digitan los datos para pacientes y se ejecutan las mismas instrucciones

```
> IMC_Pacientes <- c(25.6, 22.7, 25.9, 24.3, 25.2, 29.6, 21.3, 25.5, 27.4,
+                   22.3, 24.4, 23.7, 20.6, 22.8)
> par(mfrow=c(1,2))
> hist(IMC_Pacientes,main="A",xlab="IMC (kg/m2)",ylab="Frecuencia")
> boxplot(IMC_Pacientes,main="B", lab="IMC (kg/m2)",ylim=c(20,30))
> sw <- shapiro.test(IMC_Pacientes)
> sw
```

Shapiro-Wilk normality test

```
data: IMC_Pacientes
W = 1, p-value = 0.9
```

```
> ks <- ks.test(IMC_Pacientes,"pnorm",mean=mean(IMC_Pacientes),sd=sd(IMC_Pacientes))
> ks
```

One-sample Kolmogorov-Smirnov test

```
data: IMC_Pacientes
D = 0.1, p-value = 1
alternative hypothesis: two-sided
```