Guias Practicas de R

Lilian Martínez

November 26, 2015

```r
#1 CREACIÓN Y MANEJO DE VECTORES DE DATOS.
#1.1 VECTORES NUMÉRICOS

#FORMA 1-Crear un vector numérico vacío y añadirle luego sus elementos.

#Ejemplo 1:
v <- numeric(3)
v
```

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

```r
#Ejemplo 2:
v[3] <- 17
v
```

```
## [1]  0  0 17
```

```r
#FORMA 2-Crear un vector numérico asignándole todos sus elementos o valores.
#Ejemplo 1:
x <- c(2, 4, 3.1, 8, 6)
length(x)
```

```
## [1] 5
```

```r
#Ejemplo 2:
x <- edit(x)

#FORMA 3-Crear un vector numérico dando un rango de valores.
#Ejemplo 1:
y = 1:4; y
```

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

```r
#Ejemplo 2:
y[2] <- 5
y
```

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

```r
# Ejemplo 3:
u <- 1:12
u1=u[2 * 1:5]
u
```

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

```r
#FORMA 4-Crear un vector numérico utilizando la función assign()
#Ejemplo 1:
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
```

```r
#FORMA 5-Crear un vector numérico generando una sucesión de valores.
#Ejemplo 1:
s1 <- seq(2, 10)
s1
```

```
## [1]  2  3  4  5  6  7  8  9 10
```

```r
#Ejemplo 2:
s2 = seq(from=-1, to=5)
s2
```

```
## [1] -1  0  1  2  3  4  5
```

```r
#Ejemplo 3:
s3<-seq(to=2, from=-2)
s3
```

```
## [1] -2 -1  0  1  2
```

```r
#Ejemplo 4:
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
## [15] -0.2  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
## [29]  2.6  2.8  3.0
```

```r
#Ejemplo 5.
s5 <- rep(s3, times=3)
s5
```

```
##  [1] -2 -1  0  1  2 -2 -1  0  1  2 -2 -1  0  1  2
```

```r
#1.1.1  OPERACIONES CON VECTORES NUMÉRICOS
#Ejemplo
1: 1/x
```

```
## [1] 0.5000000 0.2500000 0.3225806 0.1250000 0.1666667
```

```r
#Ejemplo 2:
v=2*x+z+1
```

```
## Warning in 2 * x + z:  longitud de objeto mayor no es múltiplo de
la longitud de uno menor
```

```r
v
```

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

```r
#Ejemplo 3:
e1 <- c(1, 2, 3, 4)
e2<-c(4, 5, 6, 7)
crossprod(e1, e2)
```

```
##      [,1]
## [1,]   60

t(e1)%*%e2

##      [,1]
## [1,]   60

#1.1.2  OPERACIONES DE FUNCIONES SOBRE VECTORES NUMÉRICOS.
#Ejemplo 1:
xt = t(x)
xt

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

#Ejemplo 2:
u = exp(y)
u

## [1]   2.718282 148.413159  20.085537  54.598150

#OTRAS OPERACIONES:
#Ejemplo 1:
resum <- c(length(y), sum(y), prod(y), min(y), max(y))
resum

## [1]  4 13 60  1  5

#Ejemplo 2:
yo <- sort(y)
yo

## [1] 1 3 4 5

#1.2 VECTORES DE CARACTERES
#FORMA 1-Crear un vector de caracteres vacío y añadirle luego sus elementos.
#Ejemplo 1:
S<-character()

#FORMA 2-Crear un vector de caracteres asignándole todos sus elementos.
#Ejemplo 1: Crear el vector de caracteres:
deptos <- c("Santa Ana", "Sonsonate", "San Salvador")
deptos

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

#Ejemplo 2: Agregue el elemento "Ahuachapán" en la cuarta posición.
deptos[4]="Ahuachapán"
deptos

## [1] "Santa Ana"    "Sonsonate"    "San Salvador" "Ahuachapán"
```

```r
#FORMA 3-Crear un vector de caracteres dándole nombres a los elementos para identificarlos
#Ejemplo 1:
codDeptos <- c(11, 12, 13, 14)
names(codDeptos) <- c("Usulután", "San Miguel", "Morazán", "La Unión")
codDeptos
```

```
##   Usulután San Miguel    Morazán   La Unión
##         11         12         13         14
```

```r
Oriente <- codDeptos [c("La Unión", "San Miguel")]
Oriente
```

```
##   La Unión San Miguel
##         14         12
```

```r
#Ejemplo 2: Crear un vector con las etiquetas X1, Y2, ... , X9, Y10
etiqs<-paste(c("X", "Y"), 1:10, sep="")
etiqs
```

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

```r
#2.  CREACIÓN Y MANEJO DE MATRICES.
#2.1 CREACIÓN DE MATRICES NUMÉRICAS.
#FORMA 1-Crear una matriz numérica vacía y añadirle luego sus elementos.
#Ejemplo 1:
M <- matrix(numeric(), nrow = 3, ncol=4)

#Ejemplo 2:
M[2,3] <- 6
M
```

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

```r
#FORMA 2-Crear una matriz numérica asignándole todos sus elementos o valores.
#Ejemplo 1:
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
```

```r
#FORMA 3-Crear una matriz numérica dando un rango de valores
#Ejemplo 1:
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
```

```r
#FORMA 4-Crear una matriz a partir de la unión de vectores
# Crear tres vectores
x1 <- seq(0, 10, 2); x1
```

```
## [1]  0  2  4  6  8 10
```

```r
x2 <- seq(1, 11, 2); x2
```

```
## [1]  1  3  5  7  9 11
```

```r
x3 <- runif(6); x3
```

```
## [1] 0.3597401 0.9102380 0.3020148 0.5028854 0.5070357 0.6354945
```

```r
# Unir los tres vectores en una matriz por columnas.
Xcol <- cbind(x1, x2, x3)
Xcol
```

```
##      x1 x2        x3
## [1,]  0  1 0.3597401
## [2,]  2  3 0.9102380
## [3,]  4  5 0.3020148
## [4,]  6  7 0.5028854
## [5,]  8  9 0.5070357
## [6,] 10 11 0.6354945
```

```r
#Unir los tres vectores en una matriz por filas.
Xfil <- rbind(x1, x2, x3)
Xfil
```

```
##         [,1]     [,2]      [,3]      [,4]      [,5]       [,6]
## x1 0.0000000 2.000000 4.0000000 6.0000000 8.0000000 10.0000000
## x2 1.0000000 3.000000 5.0000000 7.0000000 9.0000000 11.0000000
## x3 0.3597401 0.910238 0.3020148 0.5028854 0.5070357  0.6354945
```

```r
# Acceso a las filas y columnas de una matriz.
X <- Xfil[1:3, c(2, 3)]
X
```

```
##        [,1]      [,2]
## x1 2.000000 4.0000000
## x2 3.000000 5.0000000
## x3 0.910238 0.3020148
```

```r
#2.2 OPERACIONES CON MATRICES NUMÉRICAS.
#MULTIPLICACIÓN DE MATRICES MATRICES NUMÉRICAS:
#Ejemplo 1:
v<-c(1, 2)
v %*%A
```

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

```r
#Ejemplo 2:
P <- A %*% B
P
```

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

```r
#OPERACIONES DE FUNCIONES SOBRE MATRICES NUMÉRICAS:
#Ejemplo 1:
length(A)
```

```
## [1] 6
```

```r
#Ejemplo 2:
T=sqrt(B)
T
```

```
##          [,1]     [,2]     [,3]     [,4]
## [1,] 1.000000 2.000000 2.645751 3.162278
## [2,] 1.414214 2.236068 2.828427 3.316625
## [3,] 1.732051 2.449490 3.000000 3.464102
```

```r
#Ejemplo 3: Transpuesta de una matriz
t(B)
```

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

```r
#Ejemplo 4: Determinante de una matriz:
C <- matrix(c(2, 1, 10, 12), nrow=2, ncol=2)
C
```

```
##      [,1] [,2]
## [1,]    2   10
## [2,]    1   12
```

```r
det(C)
```

```
## [1] 14
```

```r
#Ejemplo 5: Inversa de una matriz
InvC <- solve(C)
InvC
```

```
##             [,1]       [,2]
## [1,]  0.85714286 -0.7142857
## [2,] -0.07142857  0.1428571
```

```r
b=diag(2)
InvC<-solve(C, b)
InvC
```

```
##             [,1]       [,2]
## [1,]  0.85714286 -0.7142857
## [2,] -0.07142857  0.1428571
```

```r
#Ejemplo 6: Autovalores y autovectores de uma matriz simétrica
eigen(C)
```

```
## $values
## [1] 12.91608  1.08392
##
## $vectors
##            [,1]        [,2]
## [1,] -0.6754894 -0.99583022
## [2,] -0.7373697  0.09122599
```

```r
#Ejemplo 7:
diag(C)
```

```
## [1]  2 12
```

```r
#Ejemplo 8:
diag(C)
```

```
## [1]  2 12
```

```r
#Ejemplo 9:
diag(C)
```

```
## [1]  2 12
```

```r
#OTRAS OPERACIONES:
#Ejemplo 1:
c(length(C), sum(C), prod(C), min(C), max(C))
```

```
## [1]   4  25 240   1  12
```

```r
#Ejemplo 2:
O <- matrix(sort(C), nrow=2, ncol=2)
O
```

```
##      [,1] [,2]
## [1,]    1   10
## [2,]    2   12
```

```r
#CREACIÓN DE UNA MATRIZ DE CADENAS
#Ejemplo 1:
nombres <- matrix(c("Carlos", "José", "Ana", "René", "María", "Mario"),
nrow=3, ncol=2);nombres
```

```
##      [,1]     [,2]
## [1,] "Carlos" "René"
## [2,] "José"   "María"
## [3,] "Ana"    "Mario"

#3.  CREACIÓN Y MANEJO DE MATRICES INDEXADAS (ARRAY).
#Ejemplo 1:
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

#Ejemplo 2:
Z <- array(1, c(3, 3))
Z

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

#Ejemplo 3:
W <- 2*Z+1
W

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

#Ejemplo 4:
TX <- t(X)
TX

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

#Ejemplo 5:
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

#Ejemplo 6:
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

#Práctica 03 - Tipos de objetos: factores, listas y hojas de datos, operadores y funciones

#1.FACTORES NOMINALES Y ORDINALES.
#Ejemplo 1:
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
```

```r
#2. CREACIÓN Y MANEJO DE LISTAS.
#Ejemplo 1:
lista1<-list(padre="Pedro", madre="María", no.hijos=3, edad.hijos=c(4,7,9))
lista1
```

```
## $padre
## [1] "Pedro"
##
## $madre
## [1] "María"
##
## $no.hijos
## [1] 3
##
## $edad.hijos
## [1] 4 7 9
```

```r
#Ejemplo 3:
lista1[[4]][2]
```

```
## [1] 7
```

```r
#jemplo 4:
lista1["padre"]
```

```
## $padre
## [1] "Pedro"
```

```r
#Ejemplo 5:
lista1[["nombre"]]
```

```
## NULL
```

```r
x <- "nombre"
lista1[x]
```

```
## $<NA>
## NULL
```

```r
#Ejemplo 6:
subLista <- lista1[4]
subLista
```

```
## $edad.hijos
## [1] 4 7 9
```

```r
#Ejemplo 7:
lista1[5] <- list(sexo.hijos=c("F", "M", "F"))
lista1
```

```
## $padre
## [1] "Pedro"
##
## $madre
## [1] "María"
##
## $no.hijos
## [1] 3
##
## $edad.hijos
## [1] 4 7 9
##
## [[5]]
## [1] "F" "M" "F"
```

```
#Ejemplo 8: Funciones que devuelven una lista.
S <- matrix(c(3, -sqrt(2), -sqrt(2), 2), nrow=2, ncol=2)
S
```

```
##           [,1]      [,2]
## [1,]  3.000000 -1.414214
## [2,] -1.414214  2.000000
```

```
autovS <- eigen(S)
autovS
```

```
## $values
## [1] 4 1
##
## $vectors
##            [,1]       [,2]
## [1,] -0.8164966 -0.5773503
## [2,]  0.5773503 -0.8164966
```

```
evals <- eigen(S)$values
evals
```

```
## [1] 4 1
```

```
#Ejemplo 9: Crear una matriz dando nombres a las filas y columnas
Notas <- matrix(c(2, 5, 7, 6, 8, 2, 4, 9, 10),  ncol=3,
dimnames=list(c("Matemática","Álgebra","Geometría"),
c("Juan","José","René"))); Notas
```

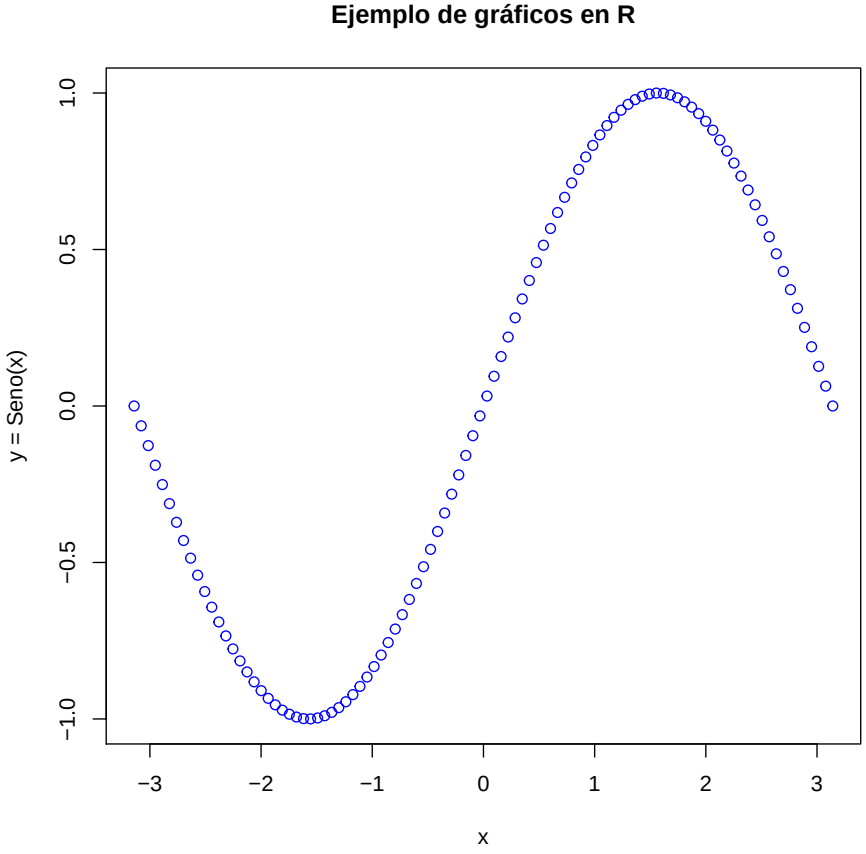
```
##            Juan José René
## Matemática    2    6    4
## Álgebra       5    8    9
## Geometría     7    2   10
```

```
#3.  CREACIÓN Y MANEJO DE HOJAS DE DATOS (DATA FRAME).
#Ejemplo 1: Creación de un data frame teniendo como columnas tres vectores:
```

```r
#log <- sample(c(TRUE, FALSE), size = 20, replace = T); log

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

##  [1]  0.2812909+0.1872242i  0.5076552+0.8397858i  1.0067152+0.8415720i
##  [4] -0.2706487+0.9415722i  0.3834390+0.6420772i  1.0147361+0.1468087i
##  [7]  0.7741639+0.0323138i -0.4229943+0.2316601i  0.5817461+0.7160524i
## [10]  1.7260241+0.1783340i -0.1399647+0.5533374i  1.3204574+0.2810097i
## [13]  1.9052410+0.7503338i  1.3780097+0.0684782i  0.0474793+0.9109889i
## [16] -0.2122710+0.7597308i  1.7548401+0.5396302i  1.6001864+0.4463336i
## [19] -1.2880362+0.9172408i -0.5516972+0.8530917i

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

##  [1] -0.41617703  1.01059198  0.61470031 -0.17979940  0.10932679
##  [6] -0.37896961 -0.62293985  1.20664229 -1.16390363 -0.77037984
## [11] -1.61239068  1.49925264  0.06317187 -1.19546125  0.17817468
## [16]  0.47729211  0.32177586  0.66147017  0.11282361  0.29310450

#Data frame compuesto por los tres vectores anteriores
#df1 <- data.frame(log, comp, num); df1

#Crear un vector de nombres de los tres vectores anteriores
nombres <- c("logico", "complejo", "numerico")
nombres

## [1] "logico"   "complejo" "numerico"

#Define los nombres de las filas del data frame asignándoles un vector de 20 #elementos co
#row.names(df1) <- letters[1:20]; df1

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","Peso"))); datos

##      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"

hoja1 <- data.frame(Edad=edad, datos, Sexo=sexo); hoja1
```

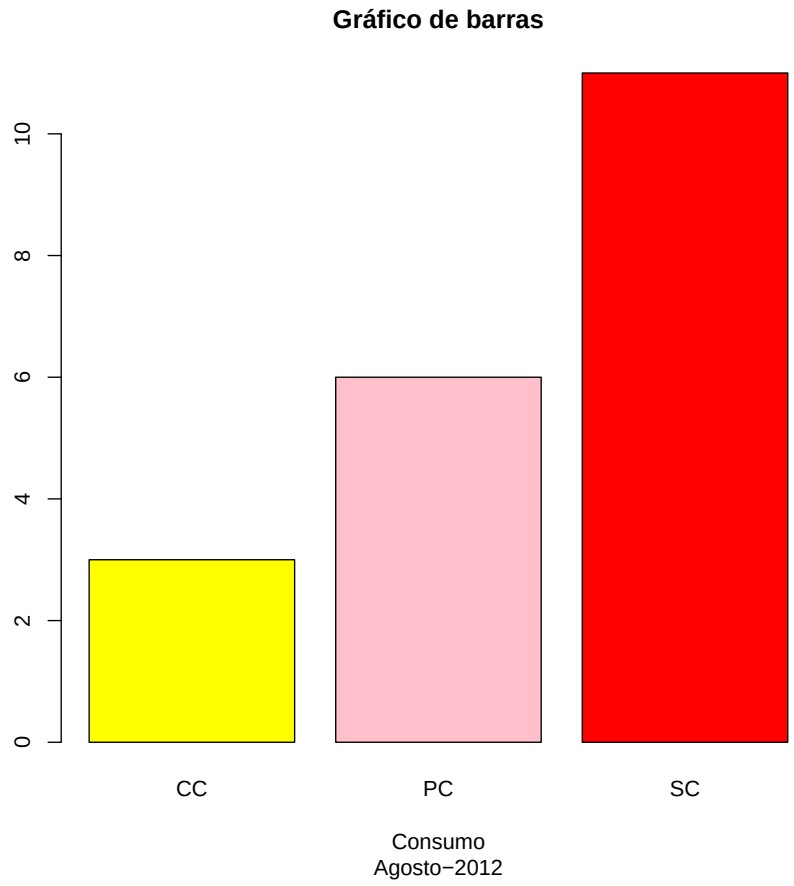
```
##   Edad Estatura Peso Sexo
## 1   18      150   65    F
## 2   21      160   68    M
## 3   45      180   65    M
## 4   54      205   69    M

#Práctica 05-Estructuras de control y definición de función en R.
#1.  ESTRUCTURA CONDICIONAL: LA ORDEN IF() Y IFELSE().

#Ejemplo 1:
if(x>0) y<-1 else y<-0

#Ejemplo 2:
x <- c(6:-4); x

##  [1]  6  5  4  3  2  1  0 -1 -2 -3 -4

sqrt(x)

## Warning in sqrt(x):  Se han producido NaNs

##  [1] 2.449490 2.236068 2.000000 1.732051 1.414214 1.000000 0.000000
##  [8]      NaN      NaN      NaN      NaN

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

##  [1] 2.449490 2.236068 2.000000 1.732051 1.414214 1.000000 0.000000
##  [8]       NA       NA       NA       NA

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

## Warning in sqrt(x):  Se han producido NaNs

##  [1] 2.449490 2.236068 2.000000 1.732051 1.414214 1.000000 0.000000
##  [8]       NA       NA       NA       NA

#2. ESTRUCTURAS ITERATIVAS O DE REPETICIÓN: FOR(), WHILE() Y REPEAT()

#Ejemplo 1:
x <- c(2, 6, 4, 7, 5, 1)
suma<-0; for(i in 1:3) suma = suma+x[i]; suma

## [1] 12

#3.  FUNCIONES ESCRITAS POR EL USUARIO.
#Ejemplo 1: Definir en R la función cuadrática  y=f(x)3x^2-5x+2
func.cuadratica <- function(x)
{
  3*x^2-5*x+2
}

#Ejemplo: calcular f(2)
y <- func.cuadratica(2);y
```

```
## [1] 4

#Ejemplo 2: Se quiere definir una función para calcular la media de un vector de datos.

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

#Guarde este objeto
save(media, file= "media.RData")

#Borre todos los objetos del área de trabajo
rm(list=ls(all=TRUE))

#Cargue el objeto con
load("media.RData")

#Pruebe la función media() con los siguientes vectores:
x <- 1:5;(media(x))

## [1] 3

y <- c(5, NA , 4, 9)


#Ejemplo 3: Se quiere definir una función para graficar la función seno de x.

Seno <- function(x)
{
  y = sin(x)
  plot(x, y, main="Ejemplo de gráficos en R",
       xlab="x", ylab="y = Seno(x)", col="blue", pch=1)
}
#Pruebe la función con el siguiente vector:
  x<-seq(-pi, pi, len=100)
Seno(x)
```

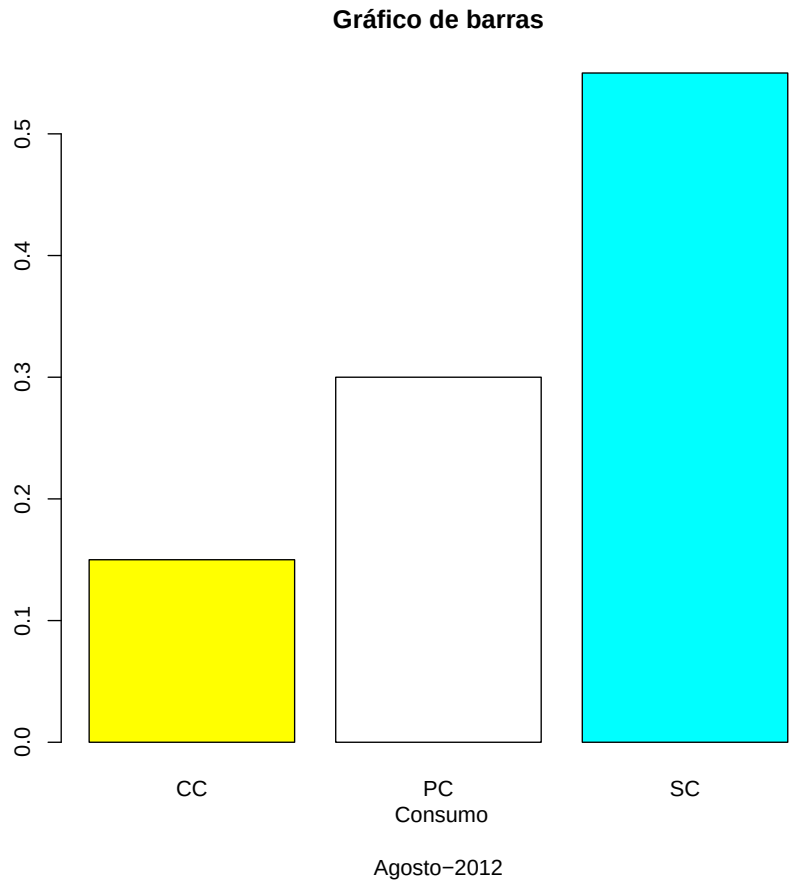



```
#Práctica 07 Análisis de datos categóricos.
#1.ANÁLISIS ESTADÍSTICO DE DATOS CATEGÓRICOS.
#Preferencias sobre el tipo de gaseosa que se consume:
#"CC"=Coca Cola, "PC"=Pepsi Cola, "SC"=Salva Cola

#Ejemplo 1: Crear un vector con el tipo de gaseosa y otro con la muestra generada #aleator
  Tipo <- c("CC", "PC", "SC"); Tipo

## [1] "CC" "PC" "SC"

#Crea un vector en las que contiene los tres tipos de refrescos
Consumo <- sample(Tipo, 20, replace=TRUE); Consumo

##  [1] "SC" "CC" "SC" "SC" "SC" "PC" "SC" "SC" "SC" "PC" "PC" "PC" "SC" "CC"
## [15] "SC" "SC" "SC" "PC" "CC" "PC"

#Editar o agregar datos
data.entry(Consumo)

#Guarde el vector en un archivo de datos
write(Consumo, "Consumo.txt")
```

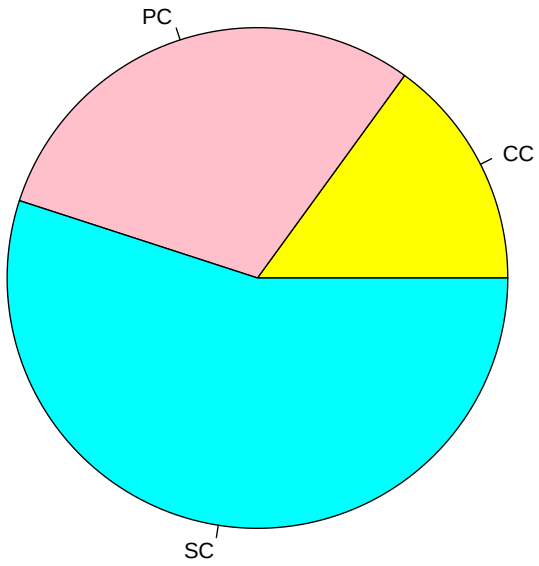
```r
#Eliminar los objetos que existen enel espacio de trabajo (Workspace)
ls()

## [1] "Consumo" "media"   "Seno"    "Tipo"    "x"       "y"

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

## character(0)

#Leer o recuperar el vector de datos o archivo de texto
Consumo <- scan("Consumo.txt", what= character(), na.strings = "NA",flush=FALSE)
Consumo

##  [1] "SC" "CC" "SC" "SC" "SC" "PC" "SC" "SC" "SC" "PC" "PC" "PC" "SC" "CC"
## [15] "SC" "SC" "SC" "PC" "CC" "PC"

ls()

## [1] "Consumo"

#Crear la tabla de distribuciónde frecuencias y proporciones
frec <- table(Consumo); frec

## Consumo
## CC PC SC
##  3  6 11

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

## Consumo
##   CC   PC   SC
## 0.15 0.30 0.55

#Conocer un resumen de los datos
summary(Consumo)

##    Length     Class      Mode
##        20 character character

#Realizar un gráfico de barras
# Para las frecuencias absolutas
barplot(frec, main="Gráfico de barras", xlab="Consumo", col=c("yellow", "pink", "red"), su
```



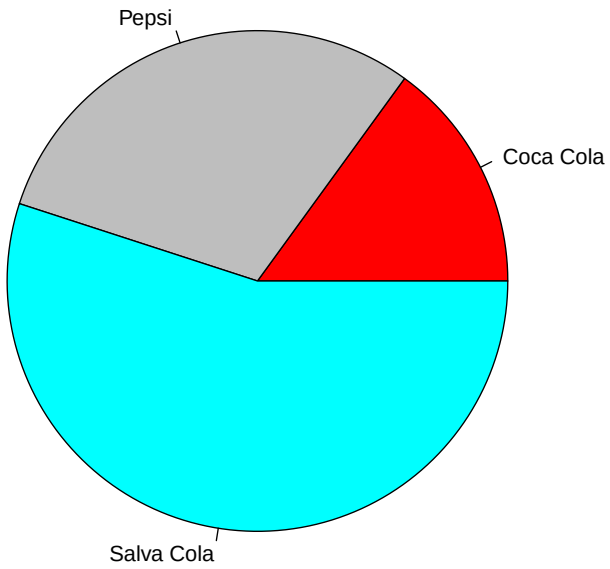

```
# Para las frecuencias relativas
barplot(prop, main="Gráfico de barras", xlab=" Consumo\n", col=c("yellow",
"white","Cyan"), sub="Agosto-2012")
```



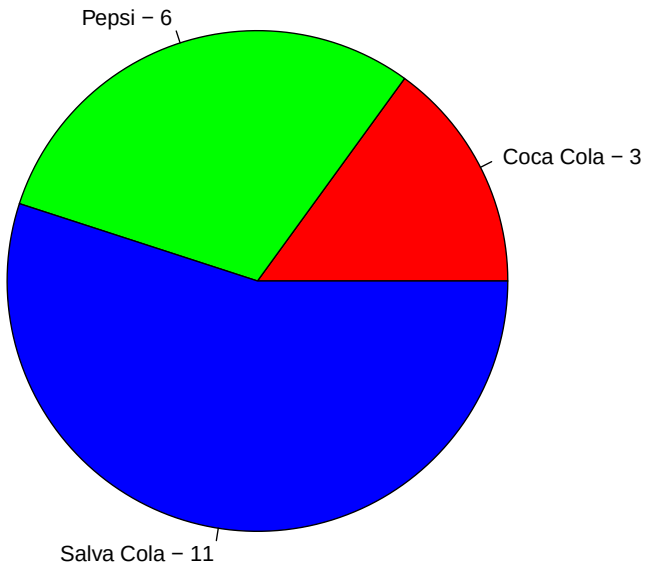

```
#Realizar un gráfico de pastel
pie(frec, main="Gráfico de pastel", xlab="Tipo de Consumo", col=c("yellow",
"pink", "cyan"), sub="Agosto-2012")
```




```
#Especificar nombres para las categorías y el color de los sectores
names(frec) = c("Coca Cola", "Pepsi", "Salva Cola")
pie(frec, main="Gráfico de pastel", xlab=" Consumo", radius=0.8, col=c("red",
"gray","cyan"), sub="Agosto-2012")
```

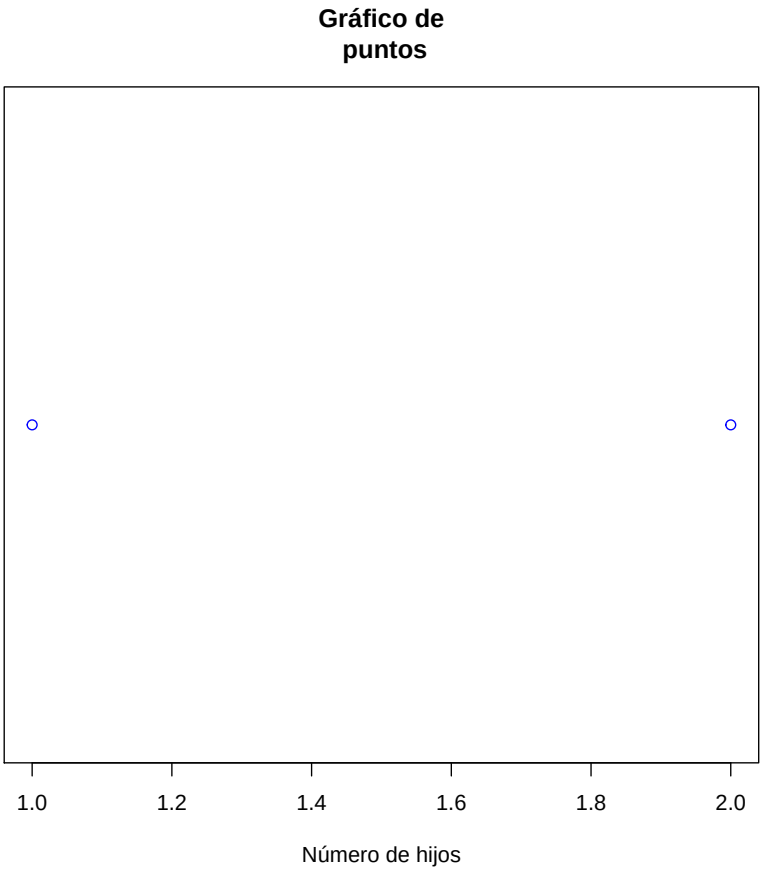



```
#Colocar valores numéricos en los sectores del gráfico
n <- length(frec)
hoja <- data.frame(frec); hoja

##         Var1 Freq
## 1  Coca Cola    3
## 2      Pepsi    6
## 3 Salva Cola   11

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

## [1] "Coca Cola - 3"   "Pepsi - 6"       "Salva Cola - 11"

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




```
#UNIDAD 2: Práctica 07-Análisis estadístico de datos univariados discretos con R.
# Crear el vector de datos.
Hijos <- c(2, 1)
data.entry(Hijos)
Hijos

## [1] 2 1

length(Hijos)

## [1] 2

# Guardar el vector de datos en un archivo de texto.
write(Hijos, "Hijos.txt")

# Leer o recuperar el vector de datos o archivo de texto
X <- scan("Hijos.txt", what = integer(0), na.strings = "NA", flush=FALSE)
ls()

## [1] "Consumo" "etiq"    "frec"    "Hijos"   "hoja"    "n"       "prop"
## [8] "X"
```

```r
#Elaborar el gráfico de puntos y diagrama de tallo-hojas (stem-and-leaf)
stripchart(Hijos, method="stack", vertical=FALSE, col="blue", pch=1,
main="Gráfico de\n puntos", xlab="Número de hijos")
```




```r
# Crear la tabla de frecuencias completa
# frecuencias individuales
fab <- table(Hijos); fab # frecuencias absolutas
```

```
## Hijos
## 1 2
## 1 1
```

```r
fre <- fab/length(Hijos); fre # frecuencias relativas
```

```
## Hijos
##   1   2
## 0.5 0.5
```

```r
Fac <- cumsum(fab); Fac # frecuencias acumuladas
```

```
## 1 2
## 1 2
```

```r
Far <- Fac/length(Hijos); Far # frecuencias acumuladas relativas
```

```
##   1   2
## 0.5 1.0
```

```r
# tabla de frecuencias completa
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
## 1 1   1      1 0.5   1 0.5
## 2 2   1      2 0.5   2 1.0
```

```r
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 1   1 0.5   1 0.5
## 2 2   1 0.5   2 1.0
```

```r
# Calcular los estadísticos descriptivos de la variable
# Estadísticos de tendencia central de los datos
media <- mean(Hijos, na.rm = FALSE); media
```

```
## [1] 1.5
```

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

```
## [1] "1"
```

```r
#Función para la moda
mediana <- median(Hijos); mediana
```

```
## [1] 1.5
```

```r
# Estadísticos de dispersión o variabilidad de los datos
# Devuelve el valor mínimo y máximo del conjunto de datos.
range(Hijos)
```

```
## [1] 1 2
```

```r
# Devuelve la cuasivarianza y la cuasivarianza muestral
cuasivar <- var(Hijos); cuasivar
```

```
## [1] 0.5
```

```r
s <- sd(Hijos); s
```

```
## [1] 0.71
```

```r
# Cálculo de Q1, Q2, Q3
quantile(Hijos,c(0.25, 0.5, 0.75))
```

```
## 25% 50% 75%
## 1.2 1.5 1.8
```

```r
# En general se pueden encontrar cualquier percentil
quantile(Hijos, 0.6)
```

```
## 60%
## 1.6
```

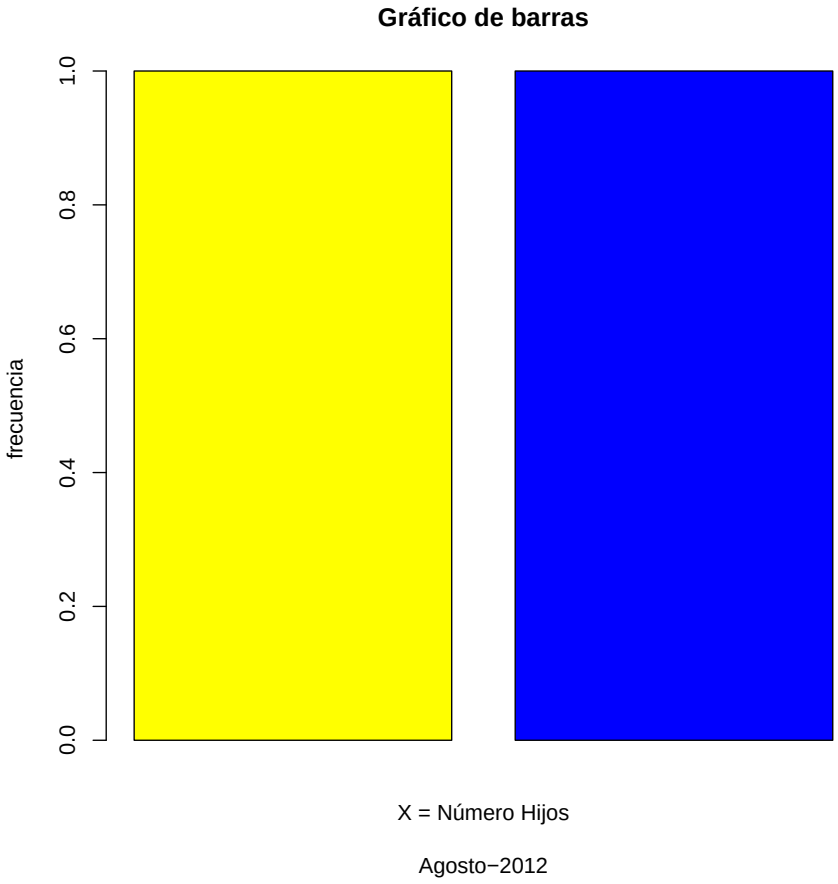
```r
# Conocer un resumen de los datos Min, Q1, Median, Mean, Q3, Max
resumen <- summary(Hijos); resumen
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     1.2     1.5     1.5     1.8     2.0
```

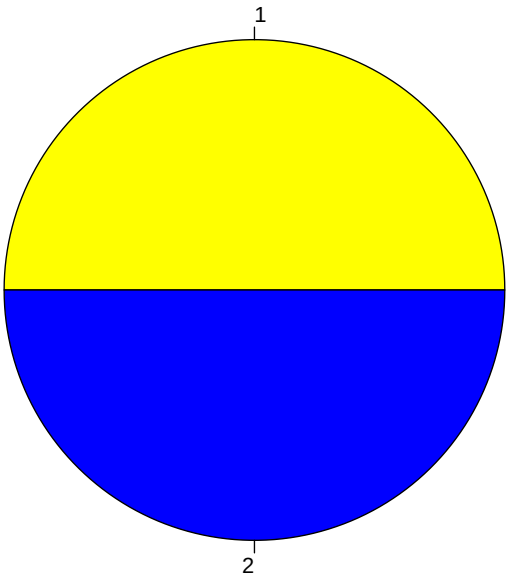
```r
# min, cuartil menor, mediana, cuartil mayor, max
fivenum(Hijos)
```

```
## [1] 1.0 1.0 1.5 2.0 2.0
```

```r
#Elaborar los gráficos que se le pueden aplicar a la variable discreta
# Gráfico de barras (por ser pocos valores)
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")
```

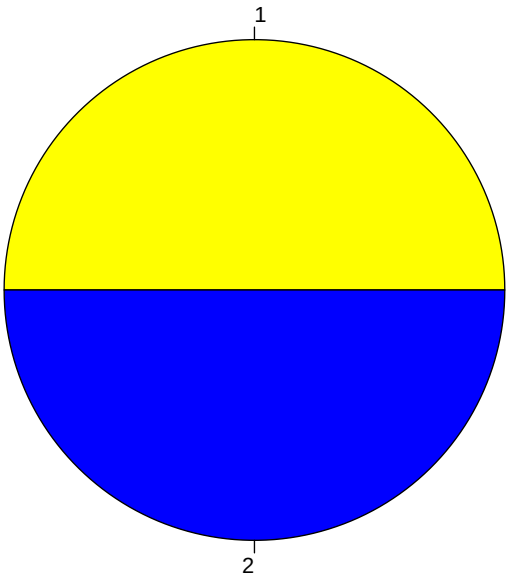



```
# Gráfico de pastel (por ser pocos valores)
pie(tfre[[2]], main="Gráfico de pastel", xlab="Número Hijos \n",
col=c("yellow", "blue","white", "orange", "cyan", "red"), sub="Agosto-2012")
```




```
# Se puede especificar nombres para las categorías
names(fab) = c("Cero", "Uno", "Dos", "Tres", "Cuatro", "Cinco")

## Error in names(fab) = c("Cero", "Uno", "Dos", "Tres", "Cuatro",
"Cinco"):  el atributo 'names' [6] debe tener la misma longitud que
el vector [2]

pie(fab, main="Gráfico de pastel", xlab="X = Número Hijos\n",
col=c("yellow", "blue","white", "orange", "cyan", "red"), sub="Agosto-2012")
```

Gráfico de pastel

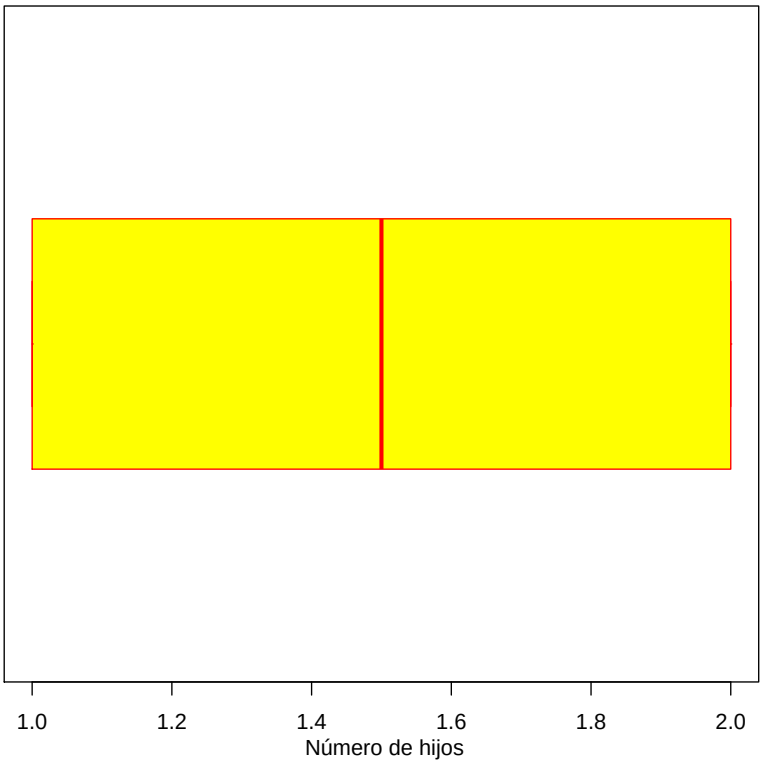


X = Número Hijos

Agosto−2012

```
# Gráfico de cajas (box-plot) es la representación gráfica de los cinco números
# Horizontal
boxplot(Hijos, main="Gráfico de caja", ylab="Número de hijos\n")
```


```
# Vertical
boxplot(Hijos, main="Gráfico de caja", xlab=" Número de hijos\n",
plot=TRUE, border="red",col="yellow", horizontal=TRUE)
```




```
#UNIDAD 2: Práctica 08 ANÁLISIS ESTADÍSTICO DE LOS DATOS.
#Crea el vector que contendrá los datos.
Notas <- c(4.47, 5.43); Notas

## [1] 4.5 5.4

data.entry(Notas)
Notas

## [1] 4.5 5.4

length(Notas)

## [1] 2

#Guarda el vector de datos en un archivo.
write(Notas, "Notas.txt")

#Lee o recupera el vector de datos desde el archivo de texto.
X <- scan("Notas.txt", what = double(0), na.strings = "NA", flush=FALSE)
ls()
```

```
##  [1] "Consumo"  "cuasivar" "etiq"     "fab"      "Fac"      "Far"
##  [7] "fre"      "frec"     "Hijos"    "hoja"     "i"        "media"
## [13] "mediana"  "moda"     "n"        "Notas"    "prop"     "resumen"
## [19] "s"        "tabla"    "tfre"     "X"
```

```
#Crea la tabla de frecuencias.
# Usa el Método de Herbert A. Sturges para determinar dicho número.
n <- length(X); n
```

```
## [1] 2
```

```
k <- 1+3.322*logb(n, 10); k
```

```
## [1] 2
```

```
k <- round(k); k
```

```
## [1] 2
```

```
# Calcula el ancho o amplitud a de cada intervalo a=rango/k
rango <- max(X)-min(X); rango
```

```
## [1] 0.9
```

```
a=rango/k; a
```

```
## [1] 0.45
```

```
a <- round(a, 3); a
```

```
## [1] 0.45
```

```
# Define los límites y puntos mediosde cada uno de los k intervalos
limites <- seq(from=min(X)-0.01/2,to=max(X)+0.01/2, by=a); limites
```

```
## [1] 4.5 4.9 5.4
```

```
options(digits=4)
ci <- cbind(1:k); ci
```

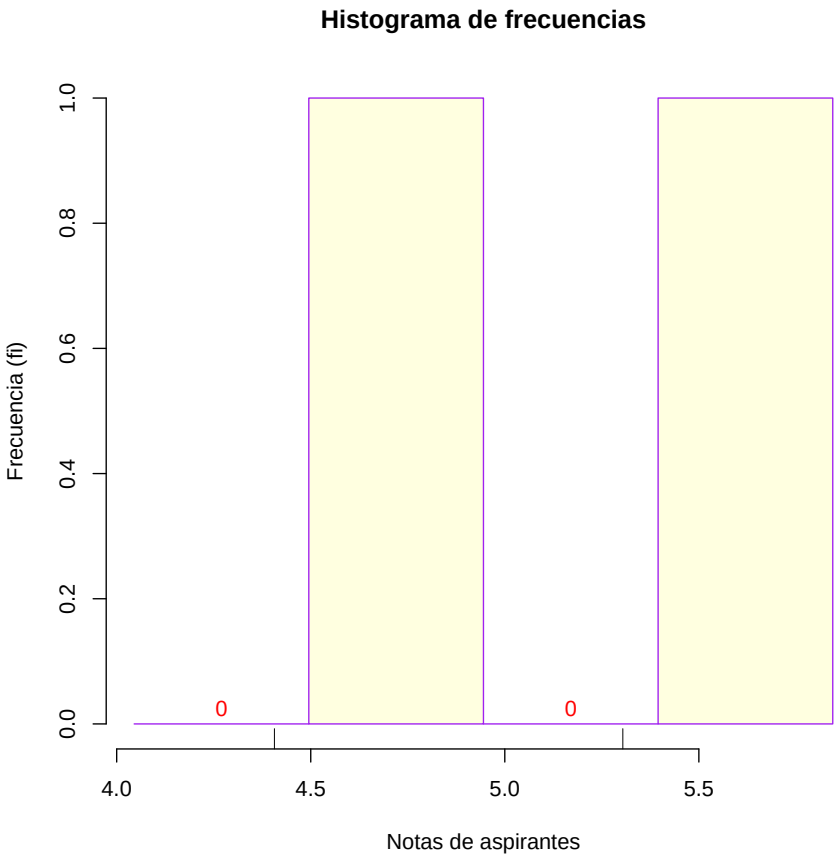
```
##      [,1]
## [1,]    1
## [2,]    2
```

```
for(i in 2:length(limites)) ci[i-1, 1] <- (limites[i] + limites[i-1])/2
ci
```

```
##      [,1]
## [1,] 4.72
## [2,] 5.17
```

```
# Encuentra las frecuencias absolutas fi para cada intervalo.
options(digits=2)
fi <- cbind(table(cut(Notas, breaks = limites, labels=NULL, include.lowest=FALSE,
right=FALSE, dig.lab=4))); fi
```

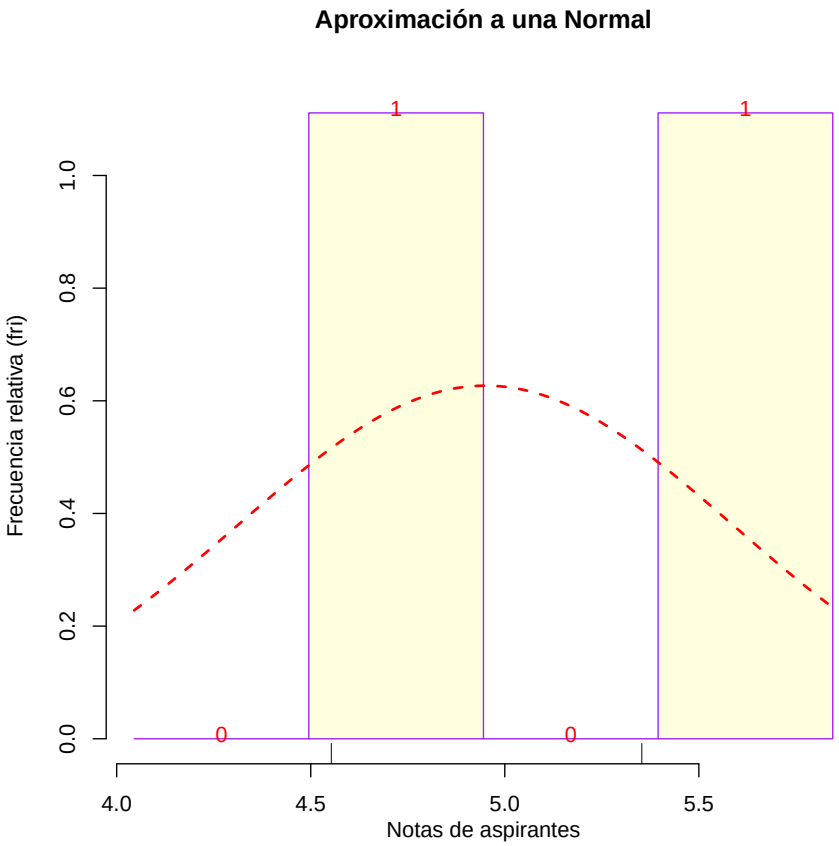
```
##               [,1]
## [4.495,4.945)    0
## [4.945,5.395)    0

# Encuentra las frecuencias relativas o proporciones fri.
options(digits=4)
fri <- fi/n; fri

##               [,1]
## [4.495,4.945)    0
## [4.945,5.395)    0

# Encuentra las frecuencias acumuladas ascendentes Fi
options(digits=2)
Fi <- cumsum(fi); Fi

## [1] 0 0

# Encuentra las frecuencias relativas acumuladas Fri
options(digits=4)
Fri <- Fi/n; Fri

## [1] 0 0

#Completa la tabla de frecuencias.
tablaFrec <- data.frame(ci=ci, fi=fi, fri=fri, Fi=Fi, Fri=Fri); tablaFrec

##                  ci fi fri Fi Fri
## [4.495,4.945) 4.72  0   0  0   0
## [4.945,5.395) 5.17  0   0  0   0

#Crea el histograma 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 = "Histograma de frecuencia

text(h$mids, h$density, h$counts, adj=c(0.5, -0.5), col="red")
rug(jitter(Notas))
```




```
is.list(h); h
```

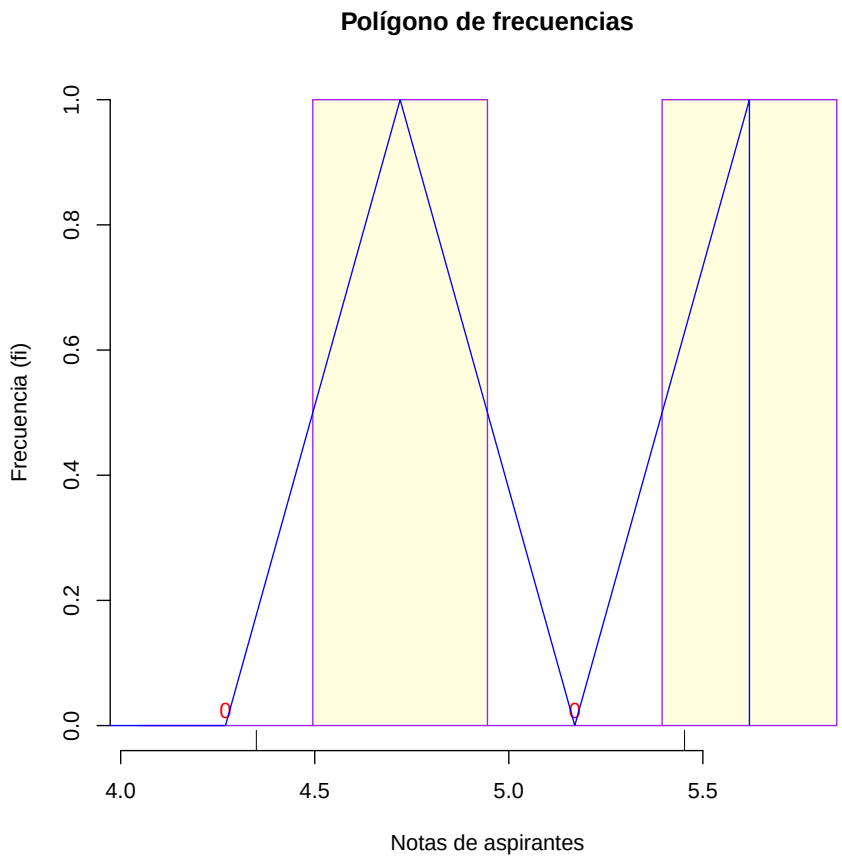
```
## [1] TRUE
## $breaks
## [1] 4.045 4.495 4.945 5.395 5.845
##
## $counts
## [1] 0 1 0 1
##
## $density
## [1] 0.000 1.111 0.000 1.111
##
## $mids
## [1] 4.27 4.72 5.17 5.62
##
## $xname
## [1] "X"
##
## $equidist
## [1] TRUE
```

```
##
## attr(,"class")
## [1] "histogram"

#Aproxima al histograma la función de densidad normal
h <- hist(X, breaks=c(limites[1]-a, limites, limites[k+1]+a), freq = FALSE,
probability = TRUE, include.lowest = FALSE, right = TRUE,
main="Aproximación a una Normal\n", col="lightyellow",lty=1,border="purple",
xlab="Notas de aspirantes\n", ylab="Frecuencia relativa (fri)",
axes=TRUE, labels=FALSE)
text(h$mids, h$density, h$counts, adj=c(0.5, 0.2), col="red")
rug(jitter(X)) # adiciona marcas de los datos
curve(dnorm(x, mean=mean(X), sd=sd(X)), col = 2, lty = 2,lwd = 2, add = TRUE)
```

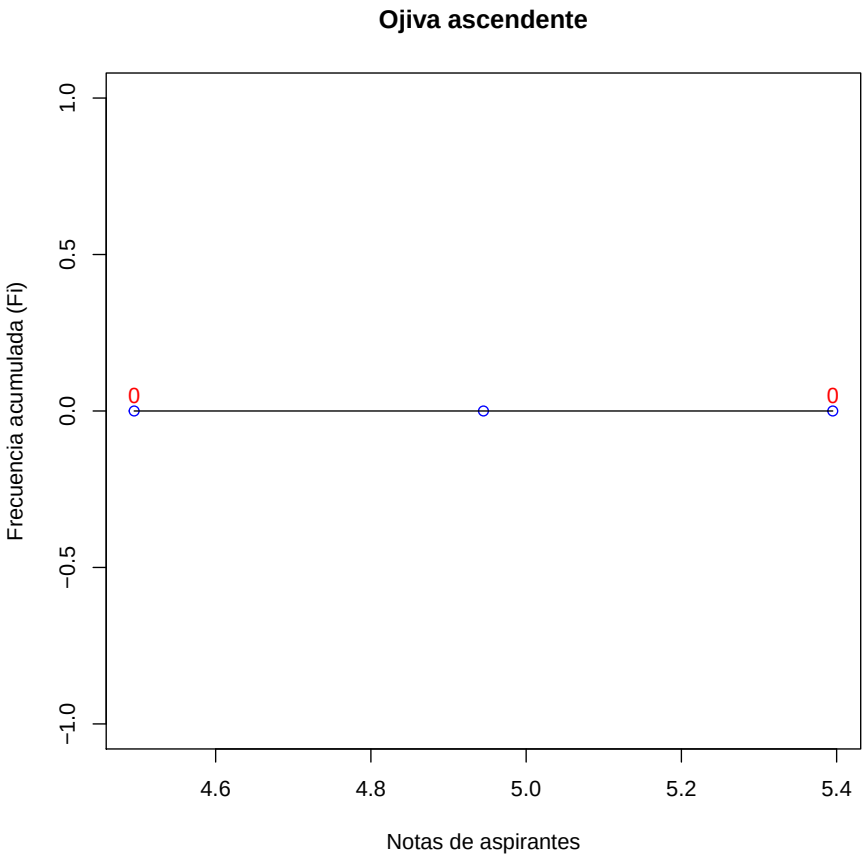



```
#Crea el polígono de frecuencias
h <- hist(X, breaks=c(limites[1]-a, limites, limites[k+1]+a), freq = TRUE,
probability=FALSE, include.lowest=FALSE,right=TRUE,
main = "Polígono de frecuencias",col="lightyellow", lty=1, border="purple", xlab="
Notas de aspirantes", ylab="Frecuencia (fi)", axes=TRUE, labels=FALSE)
text(h$mids, h$density, h$counts, adj=c(0.5, -0.5), col="red")
```

```
rug(jitter(X)) # adiciona marcas de los datos
vCi <- c(h$mids[1]-a, h$mids, h$mids[k+1]+a); vCi

## [1] 3.82 4.27 4.72 5.17 5.62 5.62

vfi <- c(0, h$counts, 0); vfi

## [1] 0 0 1 0 1 0

lines(vCi, vfi, col="blue", type="l")
```




```
#Crea la Ojiva ascendente o polígono de frecuencias acumuladas ascendentes
Fia <- c(0, Fi); Fia

## [1] 0 0 0

plot(limites, Fia, type = "p", pch=1, col = "blue", main="Ojiva ascendente",
xlab="Notas de aspirantes",ylab="Frecuencia acumulada (Fi)")
text(limites, h$density, Fia, adj=c(0.5, -0.5), col="red")
lines(limites, Fia, col="black", type="l")
```



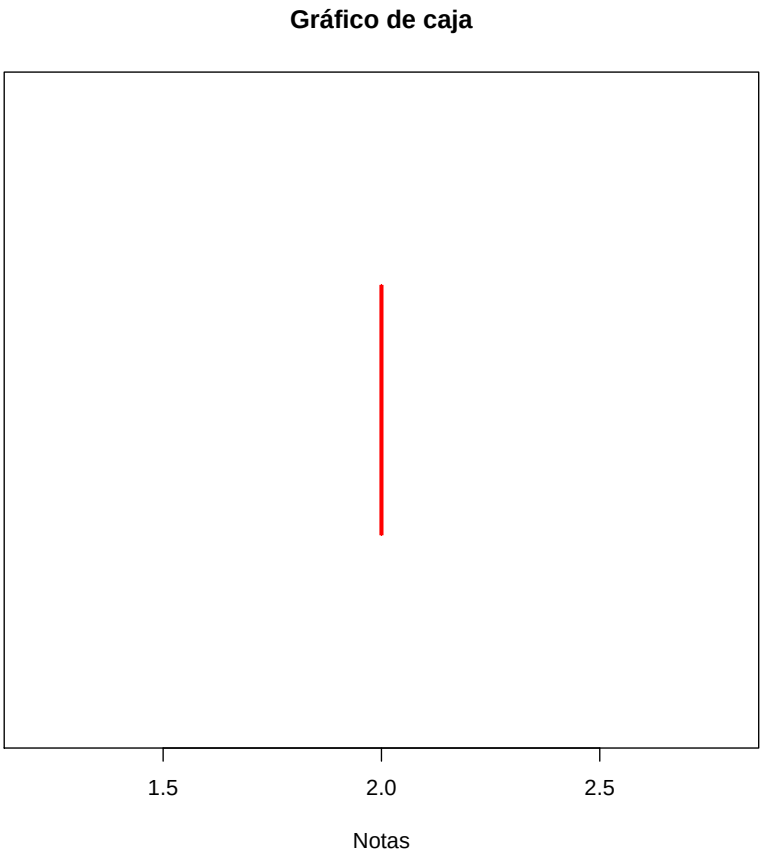

```r
#Calcula los principales estadísticos descriptivos de la variable
# Calcula la moda, ya que el R no proporciona una función para eso.
options(digits=4)
for(i in 1:k) if (fi[i] == max(fi)) break()
if(i > 1) moda <- limites[i]+((fi[i]-fi[i-1])/((fi[i]-fi[i-1])+(fi[i]-fi[i+1]) ))*a
moda <- limites[i]+(fi[i]/(fi[i]+(fi[i]-fi[i+1])))*a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

```
## [1] NaN
```

```r
# 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
```

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

```r
#Otros gráficos:
# Gráfico de cajas
boxplot(2, main="Gráfico de caja", xlab="Notas", notch=FALSE,
data=parent.frame(), plot=TRUE, border="red", col="yellow",horizontal=TRUE)
```




```r
#Práctica 09-Análisis de unavariable bidimensional categórica.

#REALICE UN ANÁLISIS ESTADÍSTICO DE LOS DATOS.
#Leer o recuperar este archivo con la función read.table()
HojaCat <- read.csv("HojaCat.csv", strip.white=TRUE);HojaCat

##    Estado.Civil  Ocupación
## 1        Casado Desocupado
## 2       Soltero    Estudia
## 3       Soltero    Trabaja
## 4        Casado    Estudia
## 5    Acompañado    Trabaja
## 6       Soltero Desocupado
## 7        Casado    Trabaja
## 8        Casado    Estudia
```

```
## 9    Acompañado Desocupado
## 10   Acompañado    Estudia
## 11       Casado    Trabaja
## 12      Soltero    Estudia
## 13   Acompañado Desocupado
## 14       Casado Desocupado
## 15      Soltero    Estudia
## 16      Soltero    Trabaja
## 17       Casado Desocupado
## 18      Soltero    Trabaja
```

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

```
##  [1] ".GlobalEnv"        "HojaCat"           "package:knitr"
##  [4] "package:stats"     "package:graphics"  "package:grDevices"
##  [7] "package:utils"     "package:datasets"  "package:methods"
## [10] "Autoloads"         "package:base"
```

```
#Crea una tabla de contigencia o de doble entrada
tablaCont <- table(HojaCat); tablaCont
```

```
##             Ocupación
## Estado.Civil Desocupado Estudia Trabaja
##   Acompañado          2       1       1
##   Casado              3       2       2
##   Soltero             1       3       3
```

```
length(HojaCat)
```

```
## [1] 2
```

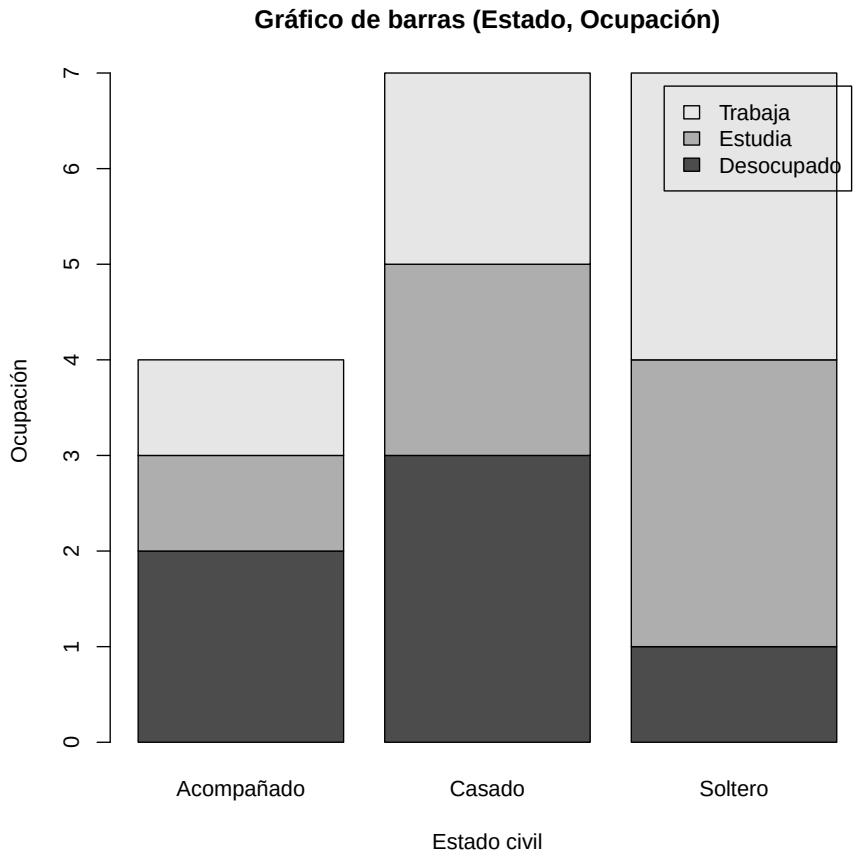
```
# Encuentra la suma de cada filade la tabla de contingencia
# Distribución marginal de X=Estado civil
suma.filas <- apply(tablaCont, 1, sum); suma.filas
```

```
## Acompañado     Casado    Soltero
##          4          7          7
```

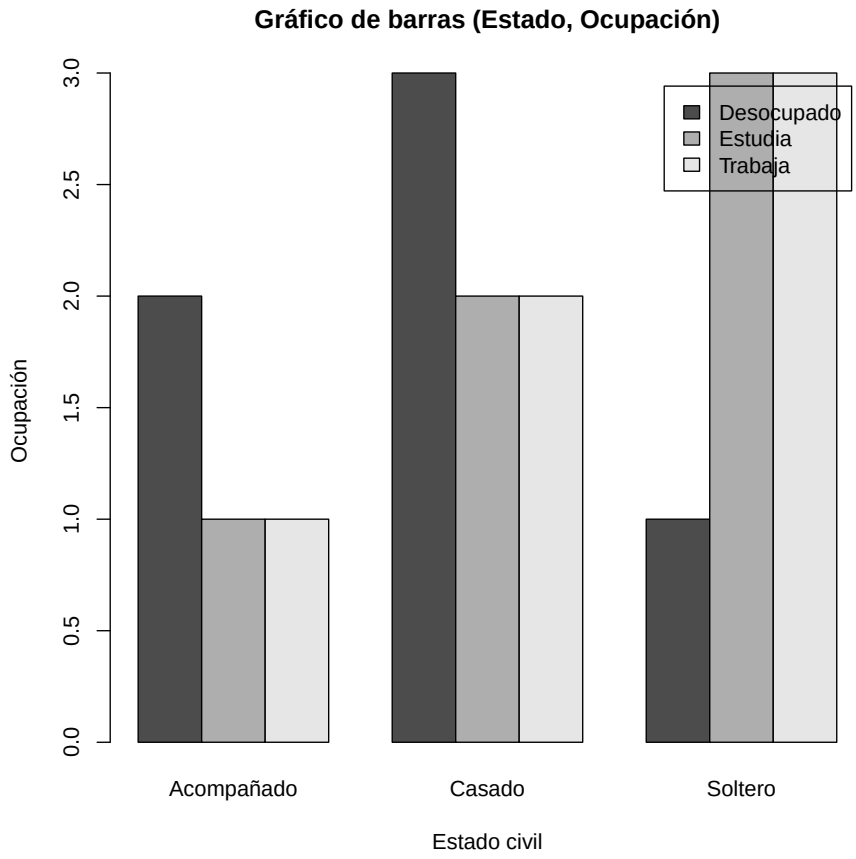
```
# Encuentra la suma de cada filade la tabla de contingencia
# distribución marginal de Y=Ocupación
suma.columnas <- apply(tablaCont,2,sum); suma.columnas
```

```
## Desocupado    Estudia    Trabaja
##          6          6          6
```

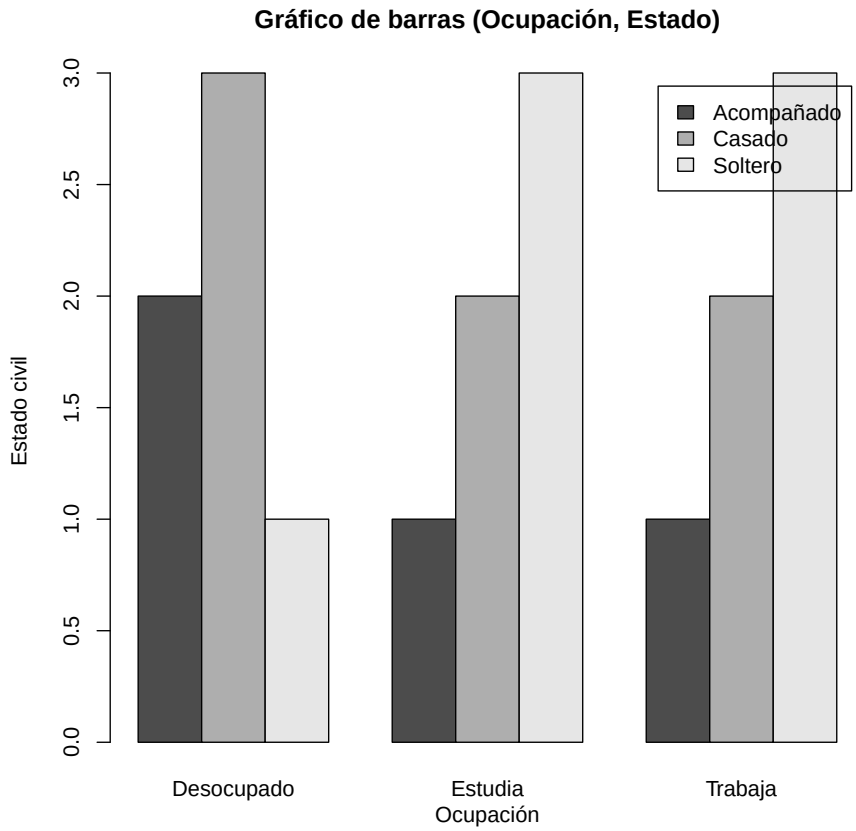
```
# Gráficos de barras para tabla de contingencia.
# Barras apiladas
barplot(t(tablaCont), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil",
ylab="Ocupación", legend.text=TRUE)
```



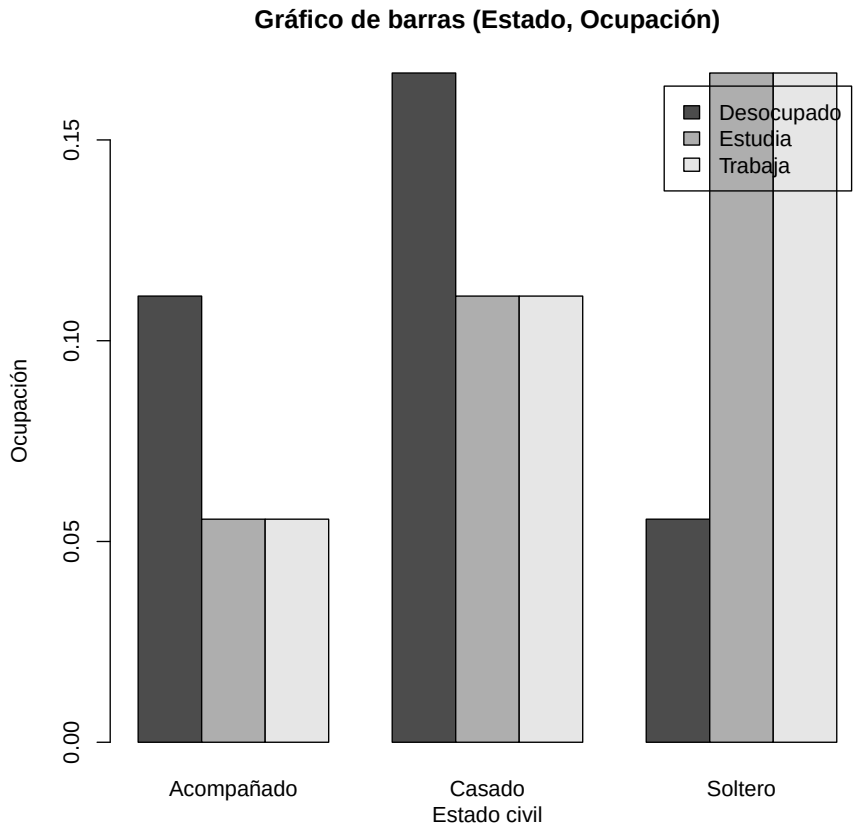

```
# Barras agrupadas
barplot(t(tablaCont), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil",
ylab="Ocupación", beside=TRUE, legend.text=TRUE)
```



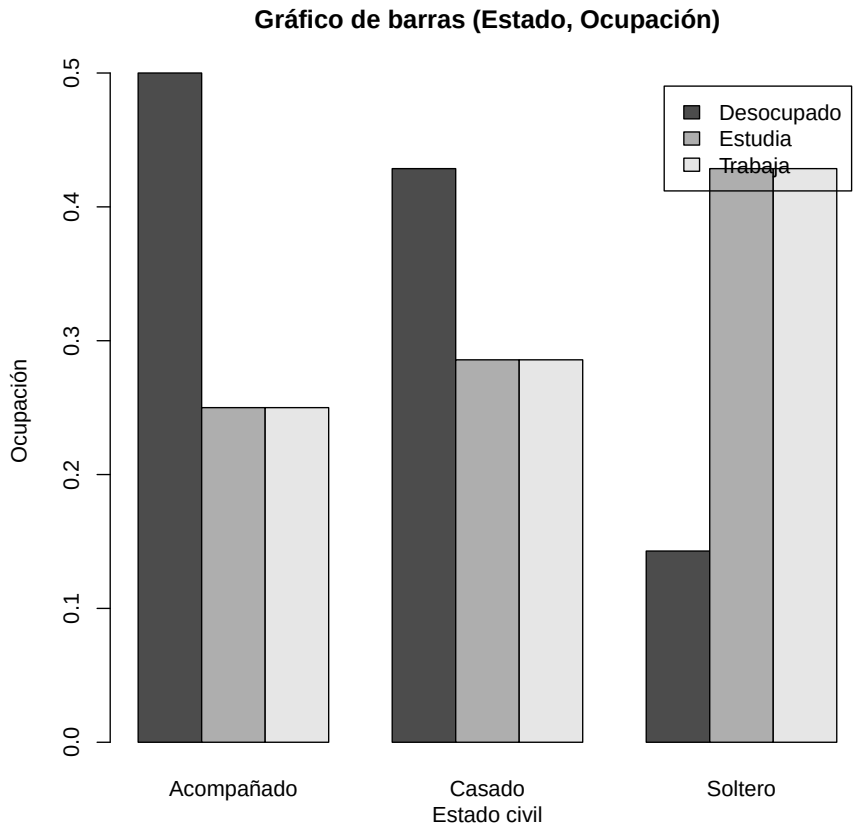

```
#Al usar beside =FALSE se obtiene el mismo gráfico de la instrucción anterior.
barplot(tablaCont, main="Gráfico de barras (Ocupación, Estado)",
xlab="Ocupación\n", ylab="Estado civil", beside=TRUE, legend.text=TRUE)
```



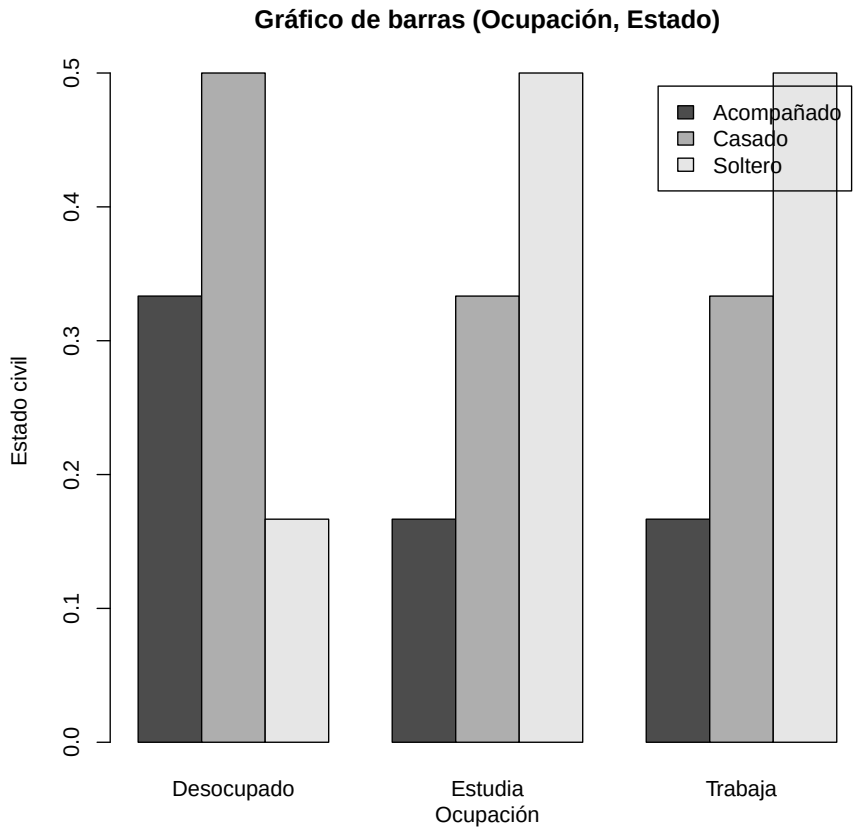

```
#Calcula tablas de proporciones o de probabilidades.
op <- options()
options(digits=3)
options('digits')

## $digits
## [1] 3

# Proporciones basadas en el total de la muestra, la suma de filas y columnas suman 1.
propTotal <- prop.table(tablaCont); propTotal

##             Ocupación
## Estado.Civil Desocupado Estudia Trabaja
##   Acompañado     0.1111  0.0556  0.0556
##   Casado         0.1667  0.1111  0.1111
##   Soltero        0.0556  0.1667  0.1667

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




```
# Proporciones basadas en el total por fila, cada fila suma 1.
propFila <- prop.table(tablaCont, 1); propFila

##             Ocupación
## Estado.Civil Desocupado Estudia Trabaja
##   Acompañado       0.500   0.250   0.250
##   Casado           0.429   0.286   0.286
##   Soltero          0.143   0.429   0.429

# Total por fila se indica en 1
barplot(t(propFila), main="Gráfico de barras (Estado, Ocupación)", xlab="Estado civil\n",
ylab="Ocupación", beside=TRUE, legend.text=TRUE)
```




```
# Proporciones basadas en el total por columna, cada columna suma 1.
propColum <- prop.table(tablaCont, 2); propColum

##             Ocupación
## Estado.Civil Desocupado Estudia Trabaja
##   Acompañado      0.333   0.167   0.167
##   Casado          0.500   0.333   0.333
##   Soltero         0.167   0.500   0.500

# Total por columna se indica en 2
barplot(propColum, main="Gráfico de barras (Ocupación, Estado)",
xlab="Ocupación\n", ylab="Estado civil", beside=TRUE, legend.text=TRUE)
```




```
#Realizar la prueba o contraste Chi-cuadrado de independencia
prueba <- chisq.test(tablaCont); prueba

## Warning in chisq.test(tablaCont):  Chi-squared approximation may
be incorrect

##
##  Pearson's Chi-squared test
##
## data:  tablaCont
## X-squared = 2, df = 4, p-value = 0.7

# Frecuencias absolutas esperadas para la prueba Chi-cuadrada
# fij = fi./No. column
prueba$expected

##             Ocupación
## Estado.Civil Desocupado Estudia Trabaja
##   Acompañado       1.33    1.33    1.33
##   Casado           2.33    2.33    2.33
##   Soltero          2.33    2.33    2.33
```