Principios y Herramientas de Programación

Dra. Jessica Andrea Carballido

jac@cs.uns.edu.ar

Dpto. de Ciencias e Ingeniería de la Computación

UNIVERSIDAD NACIONAL DEL SUR


```
     5
ames(sort(apply(ejemplo,1,sum)))[1]
1] "d"
> apply(ejemplo,1,sum)
a b c d e f g h
7 6 5 4 5 6 7 8
> sort(apply(ejemplo,1,sum))
d c e b f a g h
4 5 5 6 6 7 7 8
sort(apply(ejemplo,1,sum))[1]


y(ejemplo,1,sum))
```

Porqué usar R?

"Gnu is Not Unix"

- Es libre. Se distribuye bajo licencia GNU, lo cual significa que lo puedes utilizar gratis y ¡mejorar!
- Es multiplataforma, hay versiones para Linux, Windows, Mac, iPhone… ¡web!
- Se puede analizar en R cualquier tipo de datos.
- Es muy potente.
- Su capacidad gráfica difícilmente es superada por ningún otro paquete estadístico.
- Es compatible con 'todos' los formatos de datos (`.csv`, `.xls`, `.sav`, `.sas`…)
- Es ampliable, si quieres añadir algo: ¡empaquétalo!
- Hay miles de técnicas estadísticas implementadas, cada día hay más.

Instalación

- Se obtiene de http://cran.r-project.org
- Windows: Bajarse el ejecutable e instalar.
- Una vez instalada la distribución (y paquetes básicos) se pueden instalar paquetes adicionales.

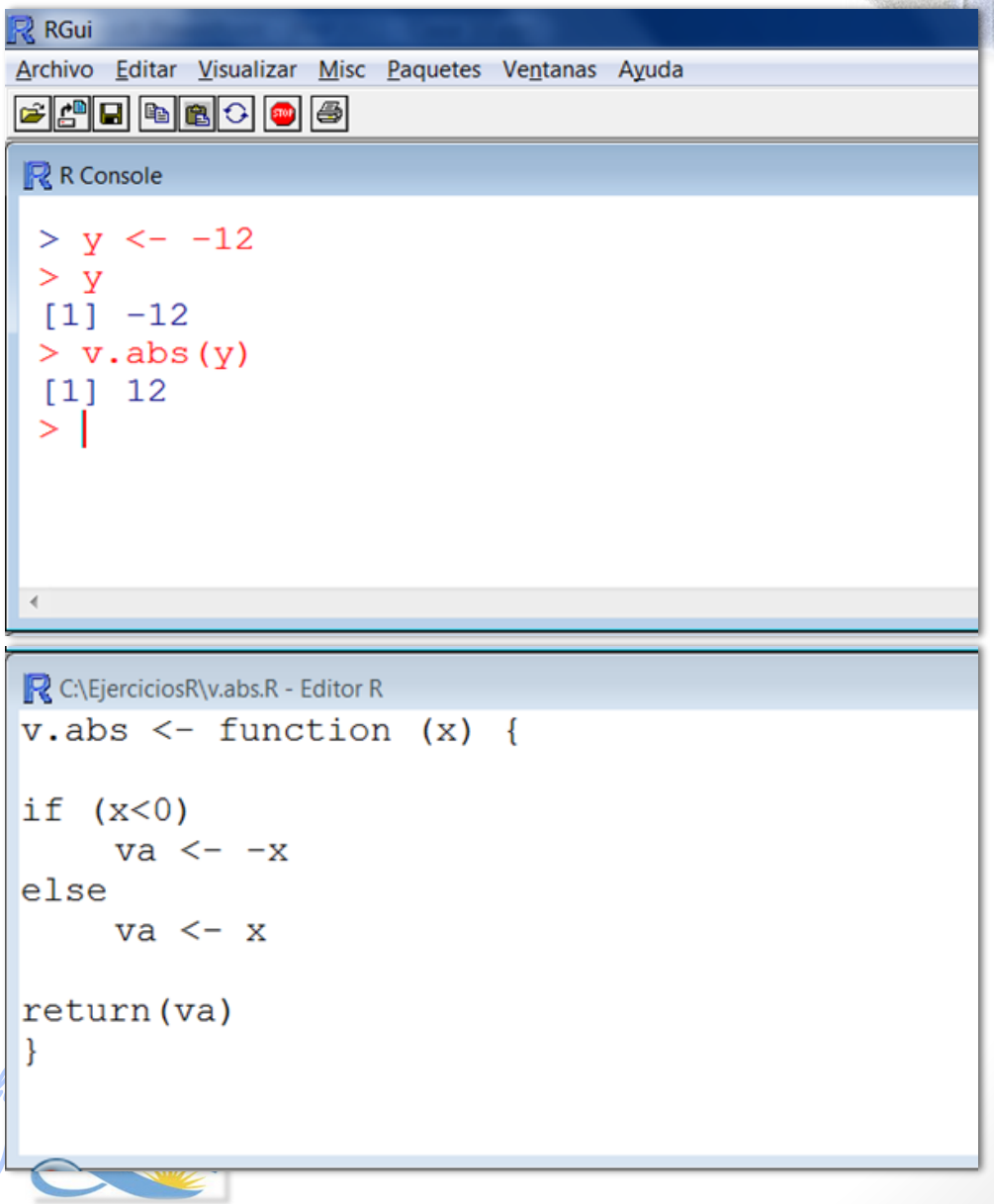
www.rstudio.com

Ambiente de Trabajo

RGui

Consola

Editor de Scripts



RStudio

Ambiente de Trabajo

~/workbench – RStudio

userTrend.R* | Q1Report.Rnw | userData

Source on Save | Run | Source

```
1  # User Trend Analysis
2  # Breakdown of active and non-active users
3
4  library(plyr)
5  library(ggplot2)
6
7  userData <- read.csv("userDataTrends.csv")
8  userData <- subset(userData, select = -c(id, group))
9  userData$active <- as.factor(userData[,1])
10
11 states <- levels(userData$state)
12
13 names(userData)
14 count(userData, "active == 1")
15 View(userData)
16
17 summary(subset(userData, active == 1)$state)
18 summary(subset(userData, active == 0)$state)
19
20 qplot(state, age, color = active, data = userData,
21       main = "Breakdown of Users by Age and State") +
22       opts(plot.title = theme_text(size = 19))
23
```

23:1 (Top Level) R Script

Console ~/

```
  active....1 freq
1       FALSE  310
2        TRUE  270
> View(userData)
> summary(subset(userData, active == 1)$state)
IA IL IN KS MI MN MO ND NE OH SD
19 26 21 21 27 49 22 26 19 16 24
> summary(subset(userData, active == 0)$state)
IA IL IN KS MI MN MO ND NE OH SD
26 27 18 31 27 49 22 32 19 33 26
> qplot(state, age, color = active, data = userData,
+       main = "Breakdown of Users by Age and State") +
+       opts(plot.title = theme_text(size = 19))
>
```

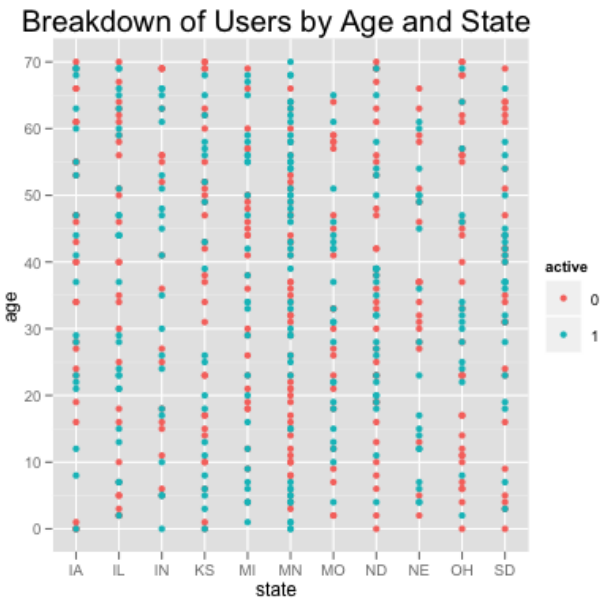
Workspace | History

Load | Save | Import Dataset | Clear All

Data	
userData	580 obs. of 5 variables
Values	
active	integer[270]
states	character[11]
Functions	
split(group, location, ...)	

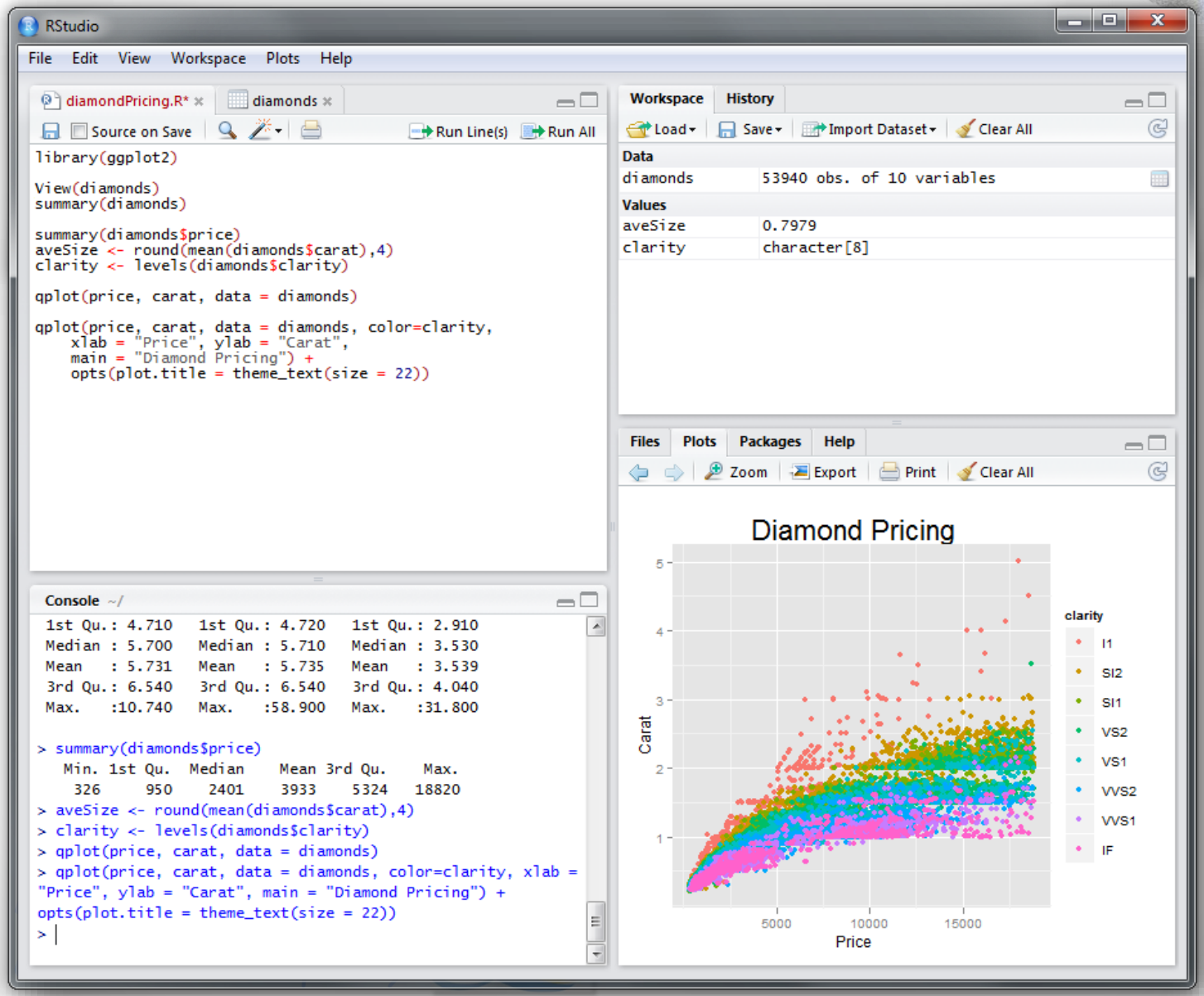
Files | Plots | Packages | Help

Zoom | Export | Clear All



RStudio

Ambiente de Trabajo



Objetos

Los objetos primitivos son:

Variables:

- Numéricas
- De caracteres
- Lógicas

Funciones

Objetos más complejos:

- Vectores, matrices, listas, data frames, entre otros.

Objetos

```
>
> a=10
> b=1.5
> c="hola"
> d=TRUE
> a
[1] 10
> b
[1] 1.5
> c
[1] "hola"
> d
[1] TRUE
> a+b
[1] 11.5
>
```

Creación

Los objetos no se declaran,
se crean la primera vez que son utilizados.

En la línea de comandos se evalúan expresiones, se ejecutan asignaciones y se definen funciones, entre otras cosas.

R es “case sensitive”. **A** y **a** son dos objetos distintos.
Los identificadores pueden contener letras, números, “.”,” _”, pero no pueden empezar por número!

Objetos

Tienen *nombre, contenido* y *atributos.*
Todo objeto tiene dos atributos:
tipo y *longitud*.

- Tipo: *clase* básica de los elementos en el objeto (numeric, character, logic).
- Longitud: número de elementos en el objeto.

Objetos

Tipos Simples:

- Numeric
  - x = 10.8
- Character
  - nombre = “pepe lopez”
- Logical
  - es.primo = TRUE

Objetos

```
>
> a <- 49 # <- operador de asignación
> # no se pueden poner espacios entre el "<" y el "-"
> # se asigna el valor 49 a la variable a
> # desde la versión 1.4.0 de R puede usarse el "="
> class(a)
[1] "numeric"
> a
[1] 49
> sqrt(a)
[1] 7
>
```

Con # se ingresan comentarios
que son ignorados por el intérprete.

Objetos

```
>
> a <- (1+1==3) # operador de comparación " == "
> class(a)
[1] "logical"
> a
[1] FALSE
>
>
```

```
>
> a <- "El perro se comió mi informe"
> class(a)
[1] "character"
> a
[1] "El perro se comió mi informe"
> b = sub("perro", "gato", a)
> b
[1] "El gato se comió mi informe"
>
```

La función sub(arg1, arg2, arg3) busca la palabra almacenada en arg1 en la cadena arg3 y la reemplaza por arg2 (si la encuentra).

Objetos

Tipos estructurados:

Los objetos se crean usando funciones predefinidas.

- **Vector**: sucesión de elementos del mismo tipo
  - > a <- **c(**1,2,3**)** ; b <- **c(**“uno”, “dos”, “tres”**)**
  - > l <- **c(**FALSE, FALSE, TRUE**)**
- **Matrix**: es una disposición ordenada de elementos del mismo tipo organizados en filas y columnas:
  - > A <- **matrix(**c(1,2,3,4,5,6), nrow = 2, ncol = 3**)**
    # matriz 2x3
    # por defecto se rellena por columnas

```
> a
[1] 1 2 3
> b
[1] "uno"  "dos"  "tres"
> l
[1] FALSE FALSE  TRUE
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

c(..) y matrix(..) son constructores de los tipos arreglo y matriz

Objetos

- **Data frame:** es una concatenación de columnas que pueden ser de distinto tipo.
  > X <- **data.frame(**nums = c(1,2,3), cads = c("uno", "dos", "tres")**)**
- **Lista**: colección de elementos que puede ser homogénea o heterogénea (eg. Un vector + 2 matrices)
  > List1 <- **list(**a, A, X**)**
- **Factor:** Es un vector para datos categóricos. Identifica las clases distintas.
  > Fact <- **factor(**c("fem", "masc", "fem", "fem")**)**

```
> X
  nums cads
1    1  uno
2    2  dos
3    3 tres
> List1
[[1]]
[1] 1 2 3

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

[[3]]
  nums cads
1    1  uno
2    2  dos
3    3 tres

> Fact
[1] fem  masc fem  fem
Levels: fem masc
>
```

Objetos

objeto	tipos	varios tipos posibles en el mismo objeto?
vector	numérico, caracter, complejo *o* lógico	No
factor	numérico *o* caracter	No
arreglo	numérico, caracter, complejo *o* lógico	No
matriz	numérico, caracter, complejo *o* lógico	No
data.frame	numérico, caracter, complejo *o* lógico	Si
ts	numérico, caracter, complejo *o* lógico	Si
lista	numérico, caracter, complejo, lógico función, expresión, . . .	Si

Objetos

Casos particulares: Objetos NA, NULL y NaN

```
is.na(NA)      # TRUE
is.na(5)       # FALSE
5 == NA        # NA

is.null(NULL) # TRUE
is.nan(NaN)   # TRUE

# Nota:
# - NA   : "Not Available", no disponible.
# - NULL : objeto "vacío"
# - NaN  : "Not a Number", no es un número (ej.: log(-1), sqrt(-1))
```

Operadores
LOGICOS y RELACIONALES

Operadores					
Aritméticos		Comparativos		Lógicos	
+	adición	<	menor que	! x	NO lógico
-	substracción	>	mayor que	x & y	Y lógico
*	multiplicación	<=	menor o igual que	x && y	id.
/	división	>=	mayor o igual que	x | y	O lógico
^	potencia	==	igual	x || y	id.
%%	módulo	!=	diferente de	xor(x, y)	O exclusivo
%/%	división de enteros				

Cortocircuito
Operadores || y &&


```
> a
Error: object 'a' not found
> TRUE || a
[1] TRUE
> FALSE && a
[1] FALSE
> TRUE | a
Error: object 'a' not found
> FALSE & a
Error: object 'a' not found
```

Operadores ARITMETICOS

```
>
> 2+3
[1] 5
> 3/2
[1] 1.5
> 2**3
[1] 8
> 4**2-3*2
[1] 10
> (56-14)/6-4*7*10/(5^2-5)
[1] -7
> 1<4
[1] TRUE
> "casa"<"arbol"
[1] FALSE
> !"dos">"tres"
[1] TRUE
> (5>10)&&(4==4)
[1] FALSE
> TRUE || FALSE
[1] TRUE
> 3&&4
[1] TRUE
> 3&&0
[1] FALSE
>
```

Ctrl+L
Para limpiar
la consola

```
> a=c(1,2)
> b=c(5,7)
> a+b
[1] 6 9
> a<b
[1] TRUE TRUE
>
```

Operadores polimórficos

Operador de suma:

- ✓ Suma enteros
- ✓ Suma reales
- ✓ Suma vectores! (elem a elem)
- ✓ Suma matrices! (elem a elem)

Algunas funciones estándar

Nombre	Operación
`sqrt(x)`	raíz cuadrada
`abs(x)`	valor absoluto
`sin(x)`, `cos(x)`, `tan(x)`	funciones trigonométricas (radianes) escriba `?Trig` para hallar otras
`pi`	número π= 3.1415926..
`exp(x)`, `log(x)`	exponencial, logaritmo
`gamma(x)`	función gamma de Euler
`factorial(x)`	función factorial
`choose(n,k)`	función que calcula el número combinatorio

```
> a=10
> sqrt(a)
[1] 3.162278
> a=c(1,5,2)
> sqrt(a)
[1] 1.000000 2.236068 1.414214
```

Funciones polimórficas también!

Aritmética vectorial

Los vectores pueden usarse en expresiones, en cuyo caso las operaciones se realizan elemento a elemento.
Dos vectores que se utilizan en la misma expresión no necesitan ser de la misma longitud. Si no lo son, el resultado será un vector de la longitud del más largo, y el más corto será "reciclado", repitiéndolo tantas veces como sea necesario hasta que coincida con el más largo. En particular, cualquier constante será simplemente repetida.

```
> c(4,3,9)+4
[1]  8  7 13
> c(4,3,9)*4
[1] 16 12 36
> c(4,3,9)+c(3,8,5)
[1]  7 11 14
> c(4,3,9)-c(3,8,5)
[1]  1 -5  4
> c(4,3,9)^2
[1] 16  9 81
> c(4,3,9)+c(3,6,8,5,2,7)
[1]  7  9 17  9  5 16
```

RECICLADO

Aritmética vectorial

```
> a=c(1,2,3)
> b=c(2,5,2)
> a&b
[1] TRUE TRUE TRUE
> a&&b
[1] TRUE
> b=c(2,5,0)
> a&b
[1]  TRUE  TRUE FALSE
> a&&b
[1] TRUE
> b=c(0,5,0)
> a&b
[1] FALSE  TRUE FALSE
> a&&b
[1] FALSE
```

CUIDADO!

Cuando usamos && en lugar de & con operandos que son vectores, R sólo evalúa el PRIMER ELEMENTO de cada vector e ignora el resto.

Objetos

Objetos lógicos: valores TRUE o T, FALSE o F.

```
> a=TRUE
> a
[1] TRUE
> a=T
> a
[1] TRUE
> a=t
> a
standardGeneric for "t" defined from package "base"

function (x)
standardGeneric("t")
<environment: 0x0000015576199ce8>
Methods may be defined for arguments: x
Use  showMethods("t")  for currently available ones.
```

```
> m=matrix(1:8, nrow=2)
> m
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
> a(m)
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
```

R es sensible a las mayúsculas.

Sus objetos predefinidos tienen nombres en minúscula en general, por lo que es una buena práctica dar nombres en MAYUSCULAS a nuestras variables.

Definición de funciones

Las funciones son objetos, como todo en R.
Por lo tanto para definir una función indicamos su nombre,
le asignamos un objeto de tipo *function* (**constructor**)
y a continuación definimos los argumentos
y la expresión que computará la función.

La mayor potencia de R es que podemos definir
funciones para extenderlo.
Es un lenguaje netamente funcional.

Condicional

- Suponga que se desea calcular el valor absoluto de un número:

```
valorAbs = function (x) {
    if (x<0)
        va = -x
    else
        va = x
    return(va)
}
```

```
va = ifelse(x<0, -x, x)
```

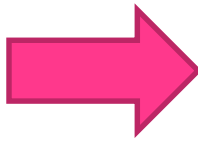
Defino la variable valorAbs como un dato de tipo función
que tiene un dato de entrada (que es x)
y que devuelve el resultado del return.

$$\sum_{i=1}^{x} i$$

Iteración

sum: función predefinida de R que retorna la suma de todos los elementos del objeto que recibe como dato de entrada.

```
sumatoria = function (x)
{
  suma = 0
  for (i in 1:x) {
     suma = suma + i
     }
  return(suma)
  }
```



```
sumatoria = function (x)
  {
  return(sum(1:x))
  }
```

IMPORTANTE: Las estructuras de control repetitivas deben usarse **lo menos posible** ya que **ralentizan** mucho los cálculos. En su lugar, cuando sea posible, usar: sum(), mean(), apply(), tapply(), sapply(), etc. (que vamos a ver más adelante).

Iteración

- **while** (cond) expr
- **repeat** expr

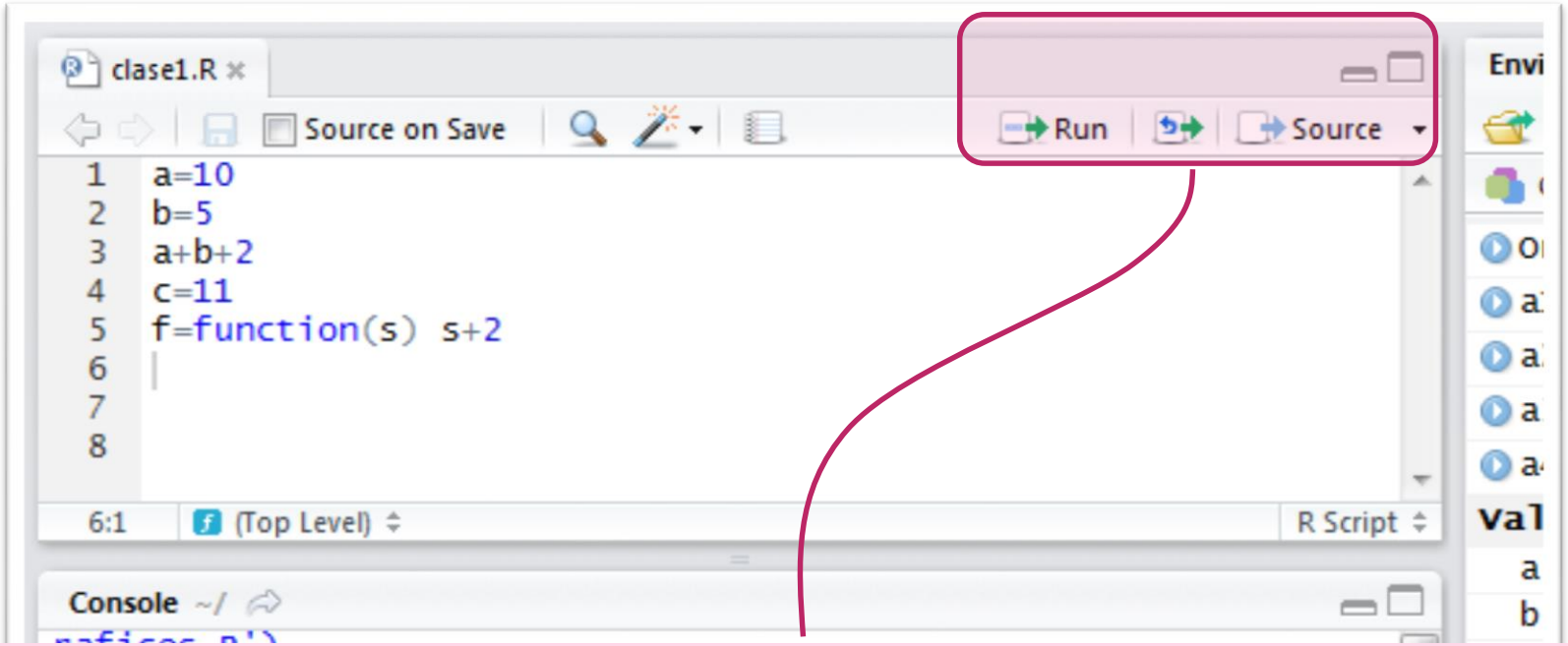
  Se sale de cualquier bucle con *break*.

  Con *next* se fuerza el comienzo de una nueva iteración.

IMPORTANTE: Las estructuras de control repetitivas deben usarse **lo menos posible** ya que ralentizan mucho los cálculos. En su lugar, cuando sea posible, usar: sum(), mean(), apply(), tapply(), sapply(), etc. (que vamos a ver más adelante).

Scripts

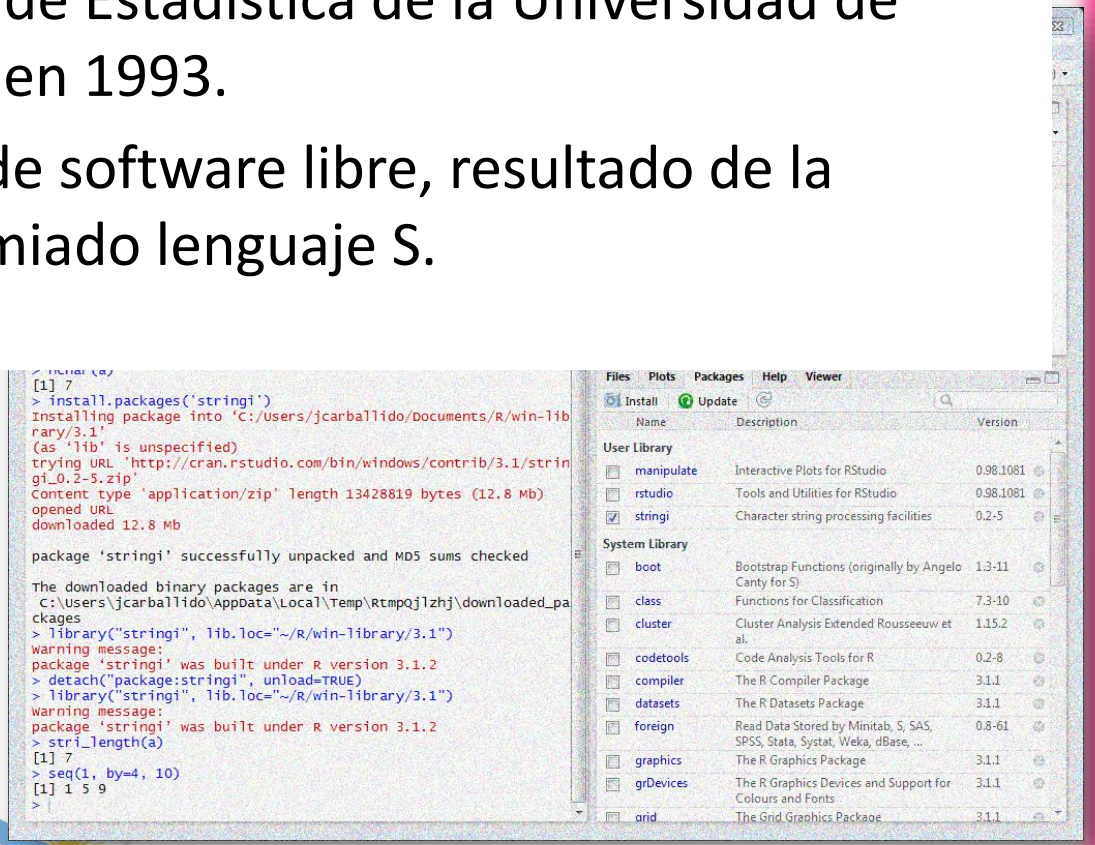
- Se pueden almacenar varias expresiones y definiciones de funciones.
- Se usan también para almacenar secuencias de comandos.



RUN: ejecuta la línea o selección actual
RE-RUN: ejecuta la última versión de la porción de código que se ejecutó la última vez (what???)
SOURCE: "compila" y ejecuta todo el documento

R

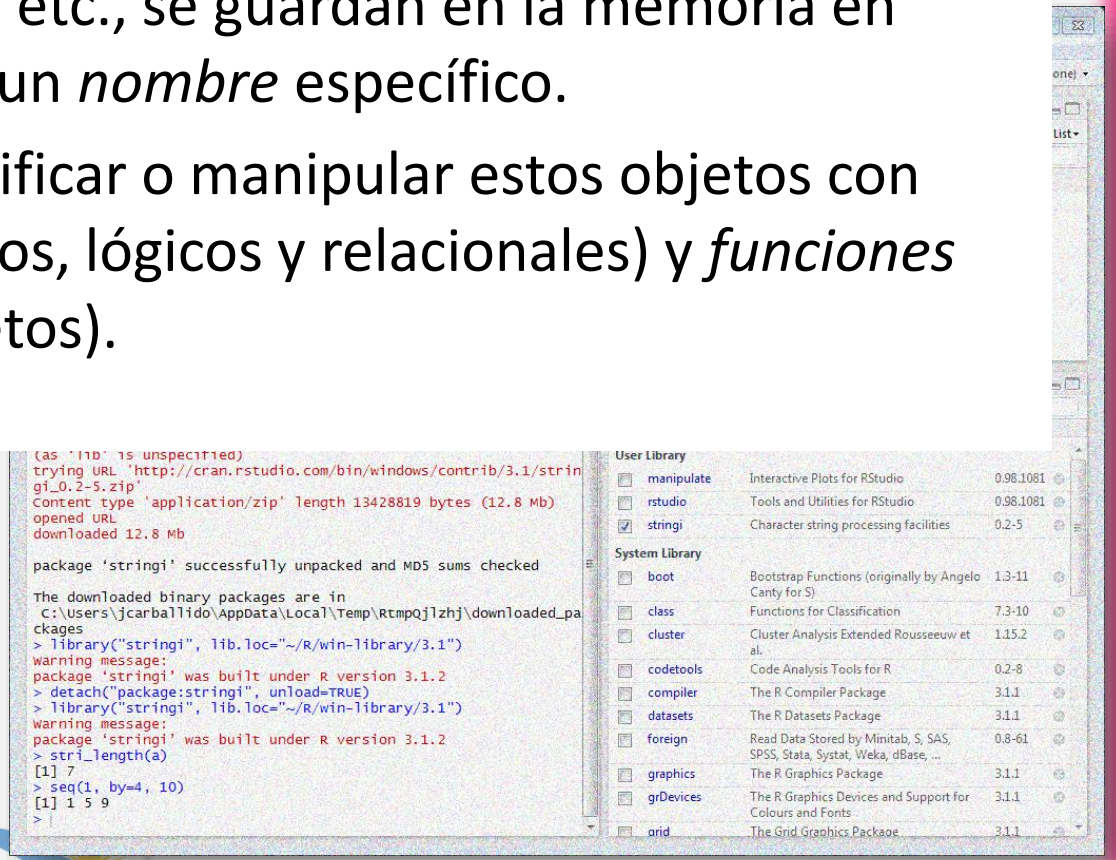
- **R** es un lenguaje y entorno de programación para análisis estadístico y gráfico.
- Fue desarrollado inicialmente por *Robert Gentleman* y *Ross Ihaka* del Departamento de Estadística de la Universidad de Auckland (New Zealand) en 1993.
- Se trata de un proyecto de software libre, resultado de la implementación del premiado lenguaje S.



Características

Lenguaje ORIENTADO A OBJETOS.

- *Orientado a Objetos* significa que las variables, datos, funciones, resultados, etc., se guardan en la memoria en forma de *objetos* con un *nombre* específico.
- El usuario puede modificar o manipular estos objetos con *operadores* (aritméticos, lógicos y relacionales) y *funciones* (que a su vez son objetos).



Características

Lenguaje SCRIPT

- Grupo de lenguajes de programación que son típicamente interpretados y pueden ser tipeados desde el teclado.
- Los scripts son un conjunto de instrucciones generalmente almacenadas en un archivo de texto que deben ser interpretados línea a línea en tiempo real para su ejecución.
- Los scripts pueden estar embebidos en otro lenguaje para aumentar las funcionalidades de este, como es el caso los scripts PHP o Javascript en código HTML.

Características

- Lenguaje interpretado, con un entorno para computación fuertemente orientado al cálculo científico y estadístico.
- Tipado dinámico.
- Se compone de un lenguaje básico y paquetes con funciones pre-computadas.
- Similar a Matlab, y su sintaxis se parece mucho a la del lenguaje C.

Algunas ventajas

- Gran facilidad para combinar paquetes existentes con código propio y con otros lenguajes de programación.
- Gran calidad de gráficos.
- **Fuerte poder expresivo** en el manejo de vectores, matrices y texto.
- Una comunidad de usuarios amplia, dinámica y con estadísticos de prestigio:
  - Grandes repositorios de paquetes (para casi cualquier cosa)
  - Mucha ayuda en la red!!!

PAGINA OFICIAL: http://www.r-project.org

Documentación

- Manuales básicos que acompañan a R:
  - Introduction to R.
  - R language definition.
- R site: http://cran.r-project.org/ (Manuals -> Contributed)
  - “R para Principiantes”, the Spanish version of “R for Beginners”, translated by Jorge A. Ahumada.
  - A Spanish translation of “An Introduction to R” by Andrés González and Silvia González.
  - “Gráficos Estadísticos con R” by Juan Carlos Correa and Nelfi González.
  - “Generacion automatica de reportes con R y LaTeX” by Mario Alfonso Morales Rivera.

Ayuda

```
> help(funcion)          # cada función tiene su página de ayuda.
> ?funcion               # ídem anterior
> example(funcion)       # los ejemplos son muy útiles
                         # para entender las funciones.
> apropos(texto)         # me da una lista con todas las funciones
                         # que contienen ese texto
```

- FAQS

  http://cran.r-prject.org/faqs.html
- Mailing lists de R
  - R-help@stat.math.ethz.ch
- En Google: R help *cuestión*


it's a
GOOD
DAY
to have
a
good
DAY