Lenguaje de programación R
Introducción

Lorena Zúñiga S.

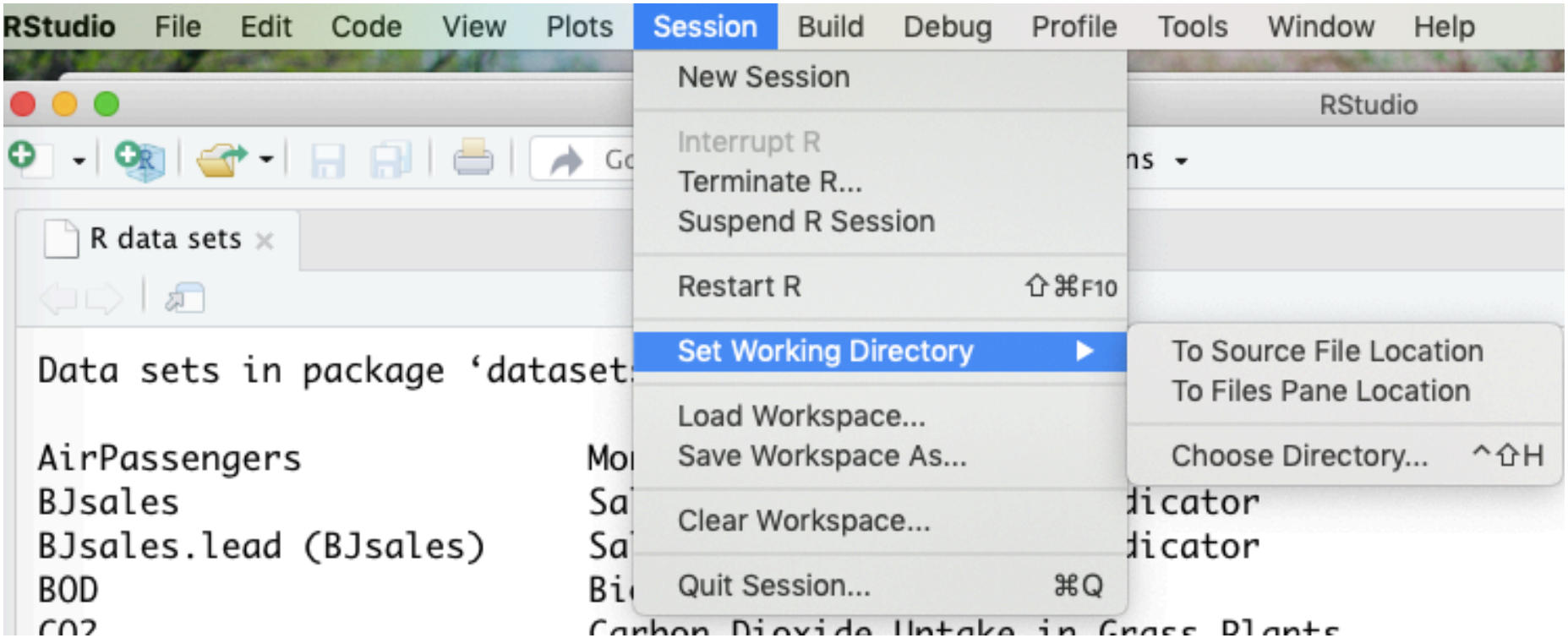
Febrero,2020

Introducción a R

- Es “case sensitive”
- Archivos con extensión .r
- Operadores de asignación: <- o bien =

El ambiente de R

- Establecer la ruta de la sesión de trabajo



El ambiente de R

- Seleccionar el directorio de trabajo: **setwd('ruta')**

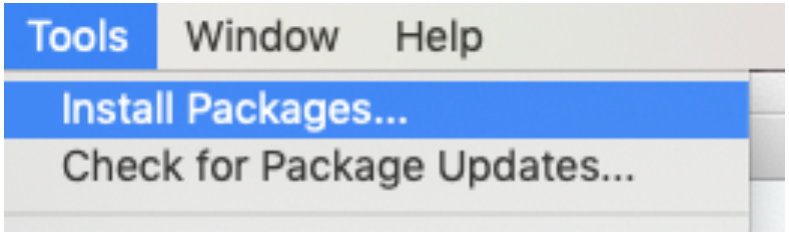
- Consultar cuál es mi directorio actual: **getwd()**

El ambiente de R

- **library()**
  - Listar los paquetes o bibliotecas que se tienen instalados
- **search()**
  - Listar paquetes que se tienen cargados
- **data()**
  - Listar datasets de ejemplo pre instalados con R
- **data(**nombreDelDataset**)**
  - Cargar en memoria datos preinstalados

El ambiente de R

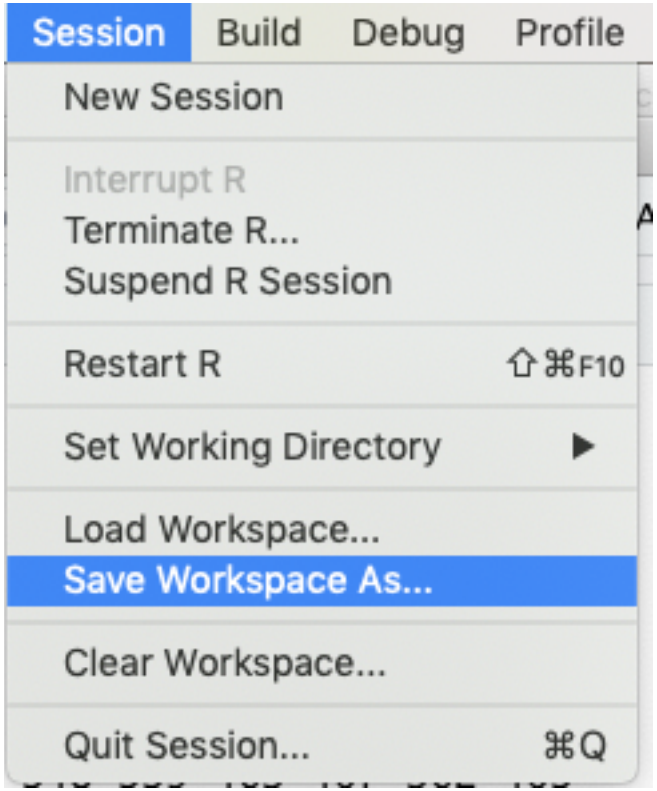
- **Instalar bibliotecas**
  - install.packages('nombreDelPaquete')



- **Cargar el paquete o biblioteca**
  - library(nombreDelPaquete)

El ambiente de R

- Guardar el espacio de trabajo
  - Archivo .Rdata
  - save(file='ruta')



El ambiente de R

- Cargar la sesión posteriormente: load(‘ruta’)
- Guardar o cargar todos los comandos/ instrucciones escritas
  - Archivo .Rhistory
  - savehistory(‘nombre.RHistory’)
  - loadHistory(‘nombre.RHistory’)

Estructuras de datos

- Escalares
- Vectores
- Cadenas de caracteres
- Matrices
- Listas
- Data Frames
- Clases

Vectores

- **Creación:**

```
> valores<- c(100,300,400)
> rango<- c(1:15)
> valores
[1] 100 300 400
> rango
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

Operaciones comunes

- Suma

```
>
> vec1= c(5,6,8,90,-4)
> vec2=c(22,5,-2,33,1)
> vec1 + vec2
[1]  27  11   6 123  -3
```

- Multiplicación

```
> vec1 * vec2
[1]  110   30  -16 2970   -4
```

- Lo mismo aplica para división, resta, residuo (%%)

Operaciones comunes

- Acceso a elementos:

```
> rango[3:7]
[1] 3 4 5 6 7
```

- Excluir excluir elementos en la selección:

```
> rango[-10]
 [1]  1  2  3  4  5  6  7  8  9 11 12 13 14 15
> rango[c(-6,-9,-11)]
 [1]  1  2  3  4  5  7  8 10 12 13 14 15
```

Funciones any() y all()

- Indican si uno o todos sus argumentos cumplen con una condición dada.

```
> rango
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
> any(rango > 10)
[1] TRUE
> any(rango < 0)
[1] FALSE
> all(rango >10)
[1] FALSE
> all(rango > 0)
[1] TRUE
```

Operadores Aritméticos

Operador	Descripción
+	Suma
-	Resta
*	Multiplicación
/	División
**	Exponente
X %% y	Módulo (x mod y, ej 10 %% 3 es 1)
X %/% y	División entera

Operadores Lógicos

Operador	Descripción
< , >	Menor que, Mayor que
<=, >=	Menor o igual, Mayor o igual
==	Igual
!=	Diferente
!x	Not X
X & Y	X and y
X | Y	X or Y

Filtrado

- Extraer elementos del vector que cumplen con alguna(s) condicione(s)

```
> rango
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
> rango[rango > 5]
 [1]  6  7  8  9 10 11 12 13 14 15
> rango[rango > 5 & rango <=10]
[1]  6  7  8  9 10
> rango[rango %% 2 == 0]
[1]  2  4  6  8 10 12 14
```

Filtrado

Función subset()

```
> rango
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
> subset(rango,rango > 8)
[1]  9 10 11 12 13 14 15
```

Filtrado

- Función **which()**
  - Devuelve las posiciones de los elementos que cumplen la(s) condición(es)

```
> rango2<-seq(1,30,2)
> rango2
 [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29
> which(rango2 > 5 & rango2< 20)
[1]  4  5  6  7  8  9 10
```

Data Frames

- Es una estructura de filas y columnas
- Creación: **data.frame()**
- **Ejemplo:**
  - Datos <- data.frame(columna1,columna2, …,columnaN)
  - Donde: columna1…columnaN son vectores de cualquier tipo

Entrada de datos

- En R los datos pueden provenir de diversas fuentes:
  - Paquetes estadísticos (SPSS, SAS, Stata)
  - Archivos de texto (ASCII, XML, Webscraping)
  - Sistemas Administradores de Bases de Datos (SQL, Oracle, MySQL, Access)
  - Otros: Excel
  - etc

Importar datos de un archivo de texto

- **read.csv**() o bien **read.table**

- misDatos <- read.table(archivo,header=valor, sep='delimitador', row.names='nombre')

Importar datos de un archivo csv

```
> datos<-read.csv('CoffeeChain.csv',header = TRUE,sep=';',stringsAsFactors = FALSE)
> str(datos)
'data.frame':   4248 obs. of  13 variables:
 $ Date          : chr  "12/1/13 00:00" "12/1/13 00:00" "11/1/13 00:00" "10/1/13 00:00"
 $ Market        : chr  "South" "South" "South" "South" ...
 $ Market.Size   : chr  "Small Market" "Small Market" "Small Market" "Small Market" ..
 $ Product       : chr  "Colombian" "Chamomile" "Chamomile" "Decaf Irish Cream" ...
 $ Product.Line  : chr  "Beans" "Leaves" "Leaves" "Beans" ...
 $ Product.Type  : chr  "Coffee" "Herbal Tea" "Herbal Tea" "Coffee" ...
 $ State         : chr  "Louisiana" "Louisiana" "Louisiana" "Louisiana" ...
 $ Type          : chr  "Regular" "Decaf" "Decaf" "Decaf" ...
 $ Inventory     : int  845 540 552 851 599 864 613 599 351 -113 ...
 $ Marketing     : int  13 31 33 13 25 17 28 15 14 58 ...
 $ Profit        : int  68 114 126 67 37 87 43 48 61 4 ...
 $ Sales         : int  128 228 246 126 160 153 167 110 129 153 ...
 $ Total.Expenses: int  25 43 45 25 58 26 58 26 35 81 ...
```

Algunas funciones útiles

Para explorar los datos:

- str()
- summary()
- attributes
- table()
- head()
- tail()

Data Frames - ejemplo

- Obtener información sobre las variables y tipos de datos en el data frame:

```
> str(datos)
'data.frame':   4248 obs. of  13 variables:
 $ Date          : chr  "12/1/13 00:00" "12/1/13 00:00" "11/1/13 00:00" "10/1/13 00:00
 $ Market        : chr  "South" "South" "South" "South" ...
 $ Market.Size   : chr  "Small Market" "Small Market" "Small Market" "Small Market" ..
 $ Product       : chr  "Colombian" "Chamomile" "Chamomile" "Decaf Irish Cream" ...
 $ Product.Line  : chr  "Beans" "Leaves" "Leaves" "Beans" ...
 $ Product.Type  : chr  "Coffee" "Herbal Tea" "Herbal Tea" "Coffee" ...
 $ State         : chr  "Louisiana" "Louisiana" "Louisiana" "Louisiana" ...
 $ Type          : chr  "Regular" "Decaf" "Decaf" "Decaf" ...
 $ Inventory     : int  845 540 552 851 599 864 613 599 351 -113 ...
 $ Marketing     : int  13 31 33 13 25 17 28 15 14 58 ...
 $ Profit        : int  68 114 126 67 37 87 43 48 61 4 ...
 $ Sales         : int  128 228 246 126 160 153 167 110 129 153 ...
 $ Total.Expenses: int  25 43 45 25 58 26 58 26 35 81 ...
```

Data Frames - ejemplo

- Obtener información estadística sobre las variables en el data frame:

```
> summary(datos)
     Date              Market           Market.Size          Product
 Length:4248        Length:4248        Length:4248        Length:4248
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character



 Product.Line       Product.Type          State              Type
 Length:4248        Length:4248        Length:4248        Length:4248
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character



   Inventory         Marketing          Profit            Sales      Total.Expenses
 Min.   :-3534.0   Min.   :  0.00   Min.   :-638.0   Min.   : 17   Min.   : 10.00
 1st Qu.:  432.0   1st Qu.: 13.00   1st Qu.:  17.0   1st Qu.:100   1st Qu.: 33.00
 Median :  619.0   Median : 22.00   Median :  40.0   Median :138   Median : 46.00
 Mean   :  749.4   Mean   : 31.19   Mean   :  61.1   Mean   :193   Mean   : 54.06
 3rd Qu.:  910.5   3rd Qu.: 39.00   3rd Qu.:  92.0   3rd Qu.:230   3rd Qu.: 65.00
 Max.   : 8252.0   Max.   :156.00   Max.   : 778.0   Max.   :912   Max.   :190.00
```

Componentes de un data frame

- attributes(nombreDataFrame)

```
> attributes(datos)
$names
 [1] "Date"           "Market"         "Market.Size"    "Product"        "Product.Line"
 [6] "Product.Type"   "State"          "Type"           "Inventory"      "Marketing"
[11] "Profit"         "Sales"          "Total.Expenses"

$class
[1] "data.frame"

$row.names
   [1]    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18
  [19]   19   20   21   22   23   24   25   26   27   28   29   30   31   32   33   34   35   36
  [37]   37   38   39   40   41   42   43   44   45   46   47   48   49   50   51   52   53   54
  [55]   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69   70   71   72
```

Data Frames - ejemplo

- Conteos por valores de una variable

```
> table(datos$Market)

Central     East    South     West
   1344      888      672     1344
> table(datos$Product.Line)

 Beans Leaves
  2232   2016
```

Data Frames - ejemplo

- Explorar las primeras o últimas filas

```
> head(datos)
           Date Market  Market.Size           Product Product.Line Product.Type     State
1 12/1/13 00:00  South Small Market         Colombian        Beans       Coffee Louisiana
2 12/1/13 00:00  South Small Market         Chamomile       Leaves   Herbal Tea Louisiana
3 11/1/13 00:00  South Small Market         Chamomile       Leaves   Herbal Tea Louisiana
4 10/1/13 00:00  South Small Market Decaf Irish Cream        Beans       Coffee Louisiana
5  9/1/13 00:00  South Small Market             Lemon       Leaves   Herbal Tea Louisiana
6  8/1/13 00:00  South Small Market Decaf Irish Cream        Beans       Coffee Louisiana
```

```
> tail(datos)
             Date Market  Market.Size        Product Product.Line Product.Type       State
4243 5/1/12 00:00   East Small Market Decaf Espresso        Beans     Espresso Connecticut
4244 4/1/12 00:00   East Small Market     Darjeeling       Leaves          Tea Connecticut
4245 4/1/12 00:00   East Small Market      Colombian        Beans       Coffee Connecticut
4246 3/1/12 00:00   East Small Market           Mint       Leaves   Herbal Tea Connecticut
4247 3/1/12 00:00   East Small Market          Lemon       Leaves   Herbal Tea Connecticut
4248 2/1/12 00:00   East Small Market Decaf Espresso        Beans     Espresso Connecticut
```

Acceso a elementos del dataframe

- Elegir columnas mediante subíndices

```
> datos[7:11]
           State    Type Inventory Marketing Profit
1      Louisiana Regular       845        13     68
2      Louisiana   Decaf       540        31    114
3      Louisiana   Decaf       552        33    126
4      Louisiana   Decaf       851        13     67
5      Louisiana   Decaf       599        25     37
6      Louisiana   Decaf       864        17     87
7      Louisiana   Decaf       613        28     43
8      Louisiana   Decaf       599        15     48
9      Louisiana Regular       351        14     61
10     Louisiana Regular      -113        58      4
11     Louisiana Regular        37        57      1
12     Louisiana   Decaf       767        13     70
13     Louisiana   Decaf       376        16     56
14     Louisiana   Decaf       484        32     62
15     Louisiana   Decaf       382        17     61
```

Acceso a elementos del dataframe

- Elegir columnas mediante subíndices

```
> datos[,c(2,5,8)]
      Market Product.Line    Type
1      South         Beans Regular
2      South        Leaves   Decaf
3      South        Leaves   Decaf
4      South         Beans   Decaf
5      South        Leaves   Decaf
6      South         Beans   Decaf
7      South        Leaves   Decaf
8      South        Leaves   Decaf
9      South         Beans Regular
10     South         Beans Regular
```

Data Frames – Acceso a sus elementos

- Usando el nombre de una columna

```
> datos$Product.Type
   [1] "Coffee"     "Herbal Tea" "Herbal Tea" "Coffee"     "Herbal Tea" "Coffee"     "Herbal Tea"
   [8] "Herbal Tea" "Espresso"   "Espresso"   "Espresso"   "Coffee"     "Espresso"   "Herbal Tea"
  [15] "Espresso"   "Herbal Tea" "Espresso"   "Espresso"   "Espresso"   "Espresso"   "Espresso"
  [22] "Herbal Tea" "Espresso"   "Espresso"   "Herbal Tea" "Espresso"   "Herbal Tea" "Espresso"
```

Data Frames – Acceso a sus elementos

- Seleccionar varias columna usando el nombre

```
> datos[c('Market','Product','Sales')]
     Market            Product Sales
1     South          Colombian   128
2     South          Chamomile   228
3     South          Chamomile   246
4     South Decaf Irish Cream   126
5     South              Lemon   160
6     South Decaf Irish Cream   153
7     South              Lemon   167
8     South          Chamomile   110
9     South        Caffe Mocha   129
10    South        Caffe Latte   153
11    South        Caffe Latte   160
12    South Decaf Irish Cream   129
13    South     Decaf Espresso   137
14    South              Lemon   206
15    South     Decaf Espresso   146
```

Data Frames – Acceso a sus elementos

- Seleccionar un rango de filas:

```
> datos[1:5,]
           Date Market  Market.Size           Product Product.Line
1 12/1/13 00:00  South Small Market         Colombian        Beans
2 12/1/13 00:00  South Small Market         Chamomile       Leaves
3 11/1/13 00:00  South Small Market         Chamomile       Leaves
4 10/1/13 00:00  South Small Market Decaf Irish Cream        Beans
5  9/1/13 00:00  South Small Market             Lemon       Leaves
```

Data Frames – Acceso a sus elementos

- Seleccionar un rango de filas aplicando condiciones

```
> datos[datos$Market=='South' & datos$Profit>=100,]
              Date Market  Market.Size           Product Product.Line Product.Type     State
2    12/1/13 00:00  South Small Market         Chamomile       Leaves   Herbal Tea Louisiana
3    11/1/13 00:00  South Small Market         Chamomile       Leaves   Herbal Tea Louisiana
31    7/1/13 00:00  South Major Market       Caffe Mocha        Beans     Espresso     Texas
35    8/1/12 00:00  South Major Market         Colombian        Beans       Coffee     Texas
53    7/1/13 00:00  South Major Market         Colombian        Beans       Coffee     Texas
54   12/1/12 00:00  South Major Market         Colombian        Beans       Coffee     Texas
55   12/1/12 00:00  South Major Market       Caffe Mocha        Beans     Espresso     Texas
57    2/1/12 00:00  South Major Market         Colombian        Beans       Coffee     Texas
267  11/1/12 00:00  South Major Market         Colombian        Beans       Coffee     Texas
```

Data Frames – Acceso a sus elementos

- Seleccionar un rango de filas aplicando condiciones con la función **subset**

```
> subset(datos,datos$Market=='South' & datos$Profit>=100)
                Date Market  Market.Size           Product Product.Line P
2    12/1/13 00:00  South Small Market          Chamomile       Leaves
3    11/1/13 00:00  South Small Market          Chamomile       Leaves
31    7/1/13 00:00  South Major Market       Caffe Mocha        Beans
35    8/1/12 00:00  South Major Market         Colombian        Beans
53    7/1/13 00:00  South Major Market         Colombian        Beans
54   12/1/12 00:00  South Major Market         Colombian        Beans
55   12/1/12 00:00  South Major Market       Caffe Mocha        Beans
57    2/1/12 00:00  South Major Market         Colombian        Beans
```

Agregar filas data frame – rbind()

rbind(dataFrame, data.frame(col1=valor1, col2=valor2,…colN=valorN))

- Se debe reasignar:
  - **dataFrame <- rbind(dataFrame, data.frame(col1=valor1, col2=valor2,… colN=valorN))**

Agregar datos al data frame

```
> p
  nombre edad
1   juan   10
2    ana   20
3   jose   30
> p<-rbind(p,data.frame(nombre='emilio',edad=11))
> p
  nombre edad
1   juan   10
2    ana   20
3   jose   30
4 emilio   11
```

Agregar una columna

- **cbind()**

```
> p<-cbind(p,Status=c('Activo','Activo','Inactivo','Activo','Inactivo')
+ )
> p
  nombre edad   Status
1   juan   10   Activo
2    ana   20   Activo
3   jose   30 Inactivo
4 emilio   11   Activo
5  laura   20 Inactivo
```

Agregar una columna

- **Otra forma de agregar una columna**

```
> provincia<-c('San Jose','Alajuela','Cartago','Heredia','Alajuela')
> p
  nombre edad   Status
1   juan   10   Activo
2    ana   20   Activo
3   jose   30 Inactivo
4 emilio   11   Activo
5  laura   20 Inactivo
> p$zona<- provincia
> p
  nombre edad   Status     zona
1   juan   10   Activo San Jose
2    ana   20   Activo Alajuela
3   jose   30 Inactivo  Cartago
4 emilio   11   Activo  Heredia
5  laura   20 Inactivo Alajuela
```

Eliminar datos del data frame

- Filas
  - nombreDataFrame[-númeroFila,]

```
> p<-p[-3,]
> p
  nombre edad   Status     zona
1   juan   10   Activo San Jose
2    ana   20   Activo Alajuela
4 emilio   11   Activo  Heredia
5  laura   20 Inactivo Alajuela
```

Eliminar datos del data frame

- Columnas:
  - nombreDataFrame[,-númeroColumna]

```
> p
  nombre edad   Status     zona
1   juan   10   Activo San Jose
2    ana   20   Activo Alajuela
4 emilio   11   Activo  Heredia
5  laura   20 Inactivo Alajuela
> p[,-2]
  nombre   Status     zona
1   juan   Activo San Jose
2    ana   Activo Alajuela
4 emilio   Activo  Heredia
5  laura Inactivo Alajuela
```

  - nombreDataFrame$col<-NULLL

Ordenar los datos

- **order**
- Ordenamiento ascendente por una columna

```
> datos[order(datos$Market),]
              Date  Market  Market.Size          Product Product.Line
145 12/1/13 00:00 Central Major Market             Mint       Leaves
146 12/1/13 00:00 Central Major Market        Earl Grey       Leaves
147 12/1/13 00:00 Central Major Market        Colombian        Beans
148 12/1/13 00:00 Central Major Market        Chamomile       Leaves
149 12/1/13 00:00 Central Major Market      Caffe Mocha        Beans
```

Ordenar los datos

- Ordenamiento descendente por una columna

```
> p
  nombre edad   Status     zona
1   juan   10   Activo San Jose
2    ana   20   Activo Alajuela
4 emilio   11   Activo  Heredia
5  laura   20 Inactivo Alajuela
> p[order(-p$edad),]
  nombre edad   Status     zona
2    ana   20   Activo Alajuela
5  laura   20 Inactivo Alajuela
4 emilio   11   Activo  Heredia
1   juan   10   Activo San Jose
```

Guardar el dataframe

- En un archivo csv
  - write.csv
- En un archivo de texto
  - write.table

```
> p
  nombre edad   Status     zona
1   juan   10   Activo San Jose
2    ana   20   Activo Alajuela
4 emilio   11   Activo  Heredia
5  laura   20 Inactivo Alajuela
> write.csv(p,'dfp.csv')
```

Manejo de datos faltantes

- NA
- **is.na**(valores)

```
> escolaridad<-c(3,15,8,NA)
> is.na(escolaridad)
[1] FALSE FALSE FALSE  TRUE
```

```
> mean(escolaridad)
[1] NA
> mean(escolaridad,na.rm = TRUE)
[1] 8.666667
```

Manejo de datos faltantes

- Asignar un valor por omisión si se detecta un NA

```
> escolaridad
[1]  3 15  8 NA
> promedio<-round(mean(escolaridad,na.rm = TRUE))
> promedio
[1] 9
> ifelse(is.na(escolaridad),promedio, escolaridad)
[1]  3 15  8  9
```

Manejo de datos faltantes

- Excluir valores NA del análisis

```
> p
  nombre edad   Status     zona escolaridad
1   juan   10   Activo San Jose           3
2    ana   20   Activo Alajuela          15
4 emilio   11   Activo  Heredia           8
5  laura   20 Inactivo Alajuela          NA
> na.omit(p)
  nombre edad Status     zona escolaridad
1   juan   10 Activo San Jose           3
2    ana   20 Activo Alajuela          15
4 emilio   11 Activo  Heredia           8
```

¿PREGUNTAS?