R Studio

Curso – Taller de uso de **R Studio** para Estadística y Probabilidad I

Instructor: Gamar Zaid Joseph García Castillo

Forma de trabajo

- El curso estará dividido en 4 sesiones de 5 horas cada una, con uno o dos recesos, a consideración de los participantes. El miércoles 12 de diciembre no hay labores en la UNAM.

- **Puntualidad** y asistencia 100% (asistir los cuatro días del curso para tener derecho a constancia).

- Ejercicios propuestos en cada sesión.

- Presentar una propuesta de estrategia didáctica (de considerarlo posible) el día viernes 14 de diciembre de forma individual o en equipos.

Objetivos del curso

- Relacionar los temas de estadística descriptiva con el uso del software R Studio, mediante comando básicos y graficación.

- Que refuercen sus conocimientos de Microsoft Excel aplicados a los temas del curso.

- Que los profesores que imparten la asignatura de Estadística y Probabilidad conozcan una herramienta de cómputo para el apoyo en diferentes temas de la asignatura.

- Que los profesores realicen la propuesta de una estrategia didáctica con el uso del software R Studio para sus estudiantes.

Temas generales a revisar

- Breve historia de R Project y herramientas existentes para el análisis de datos.
- Manejo básico de Microsoft Excel.
- Comandos para trabajar en R Studio.
- Graficación.
- Estadística descriptiva.
- Conceptos de datos bivariados.

Día 1

Breve historia de R Project

- R se desprende del lenguaje de programación S.

- En los laboratorios Bell se tenía pensado diseñar un ambiente de trabajo enfocado en el análisis estadístico.

- En los años 80's Chambers y Hastie implementan la funcionalidad del análisis estadístico en S y ya se tenían una gran variedad de técnicas estadísticas implementadas.

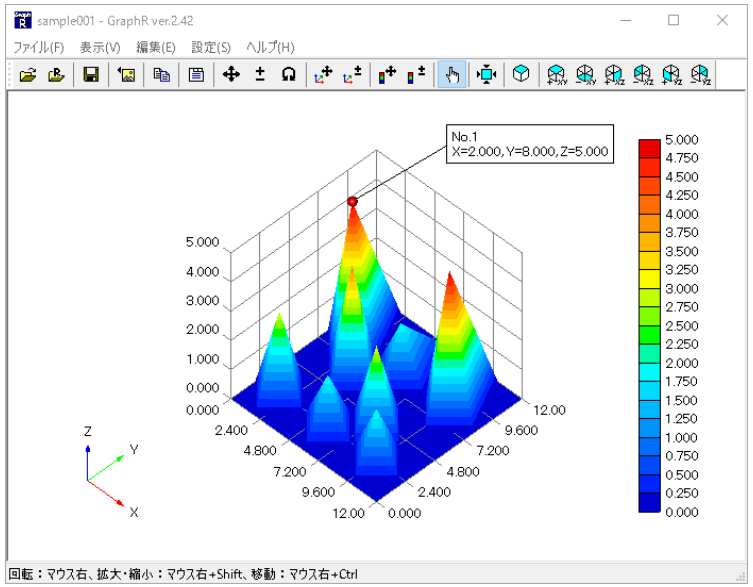
Breve historia de R Project

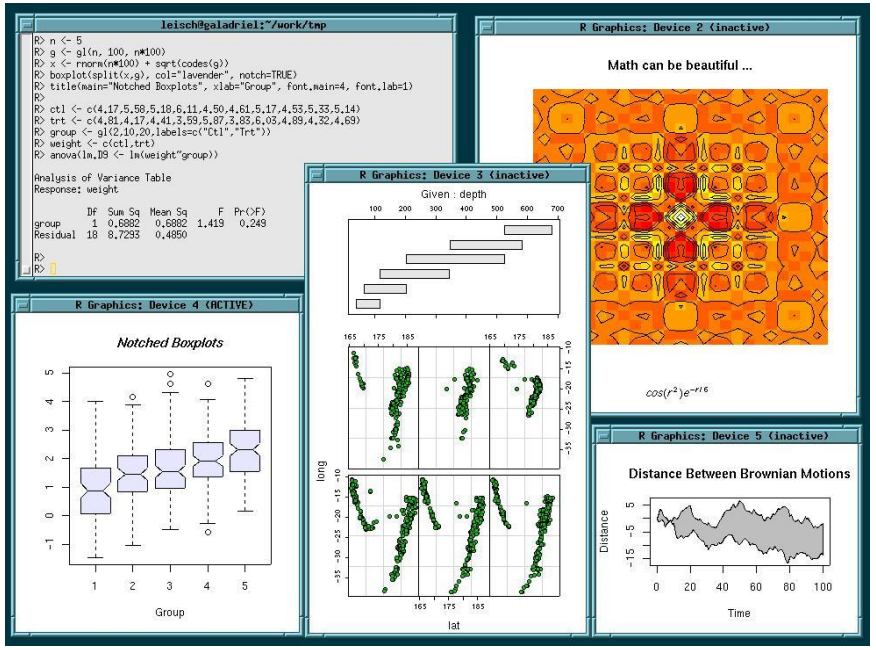
- En 1991, Ross Ihaka y Robert Gentleman deciden implementar su propia versión de S, ellos la bautizaron como R. Tan bien lograron hacer esto que para 1993 ponen R al público general. Esto tuvo tanto éxito, que para 1995 liberaron el código bajo una licencia GNU.

- Para el año 2015 las versiones de R son totalmente estables y corren prácticamente en cualquier plataforma, Windows, Mac, Linux y tiene un cambio de versiones muy rápido.

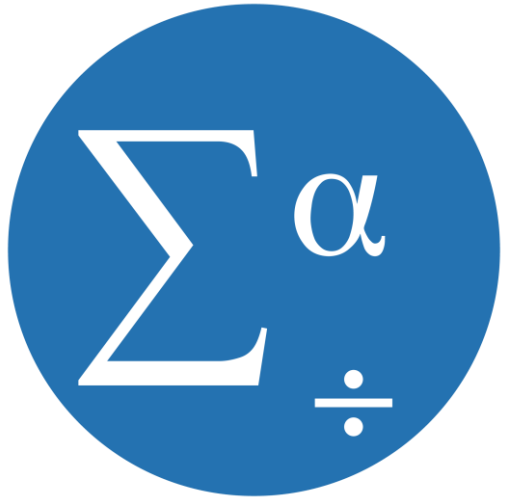
Breve historia de R Project

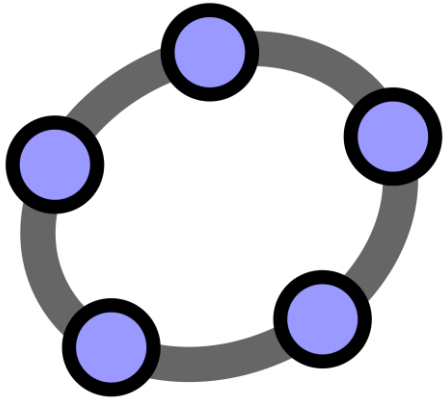
- El lenguaje R en realidad es muy pequeño y todas las funcionalidades de él se van agregando con **paquetes**. La capacidad de graficación además es muy madura y está considerado uno de los mejores sistemas para graficar.





Herramientas para el manejo de datos





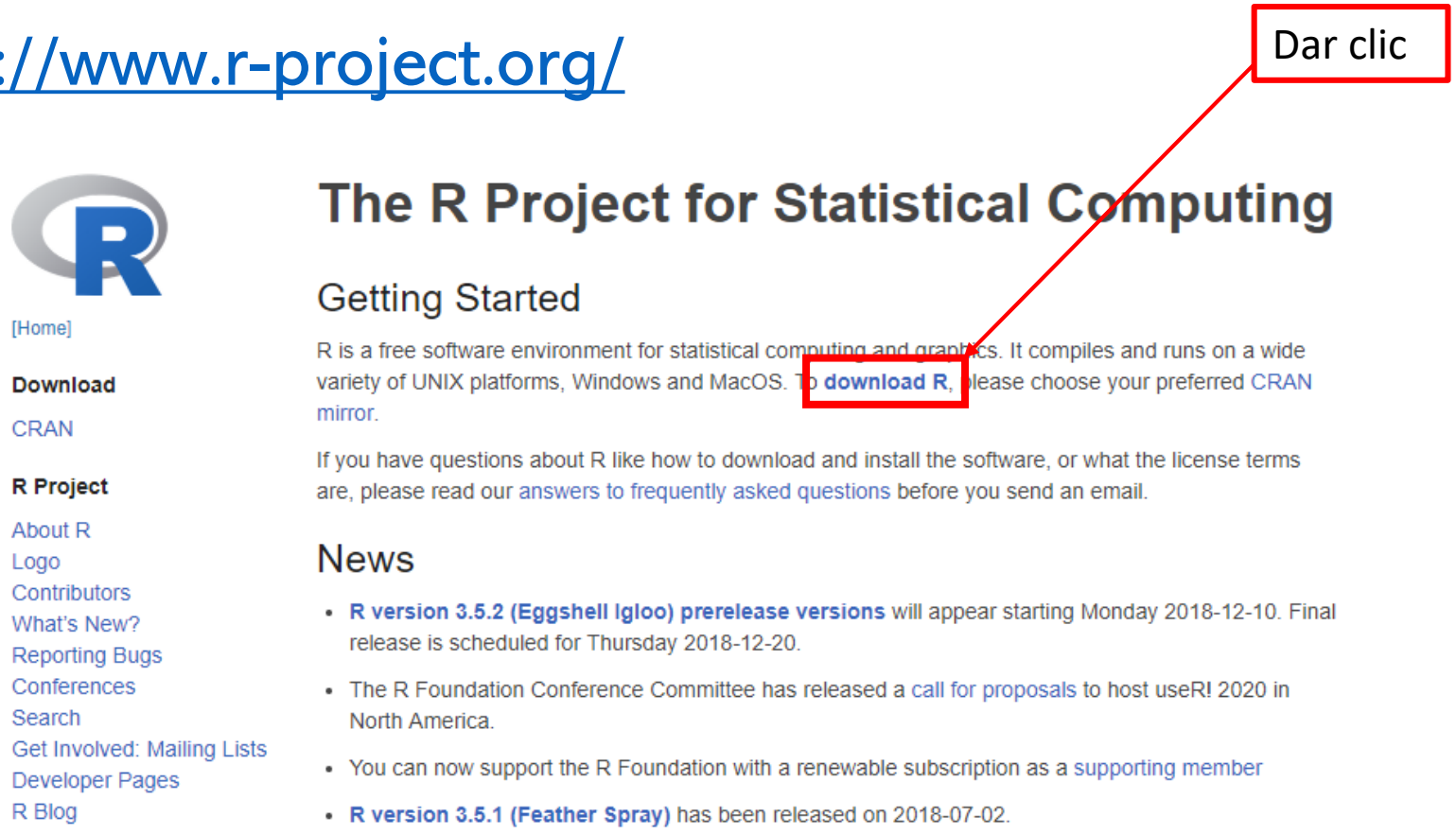
TinkerPlots

Fathom®

Instalación de R Project

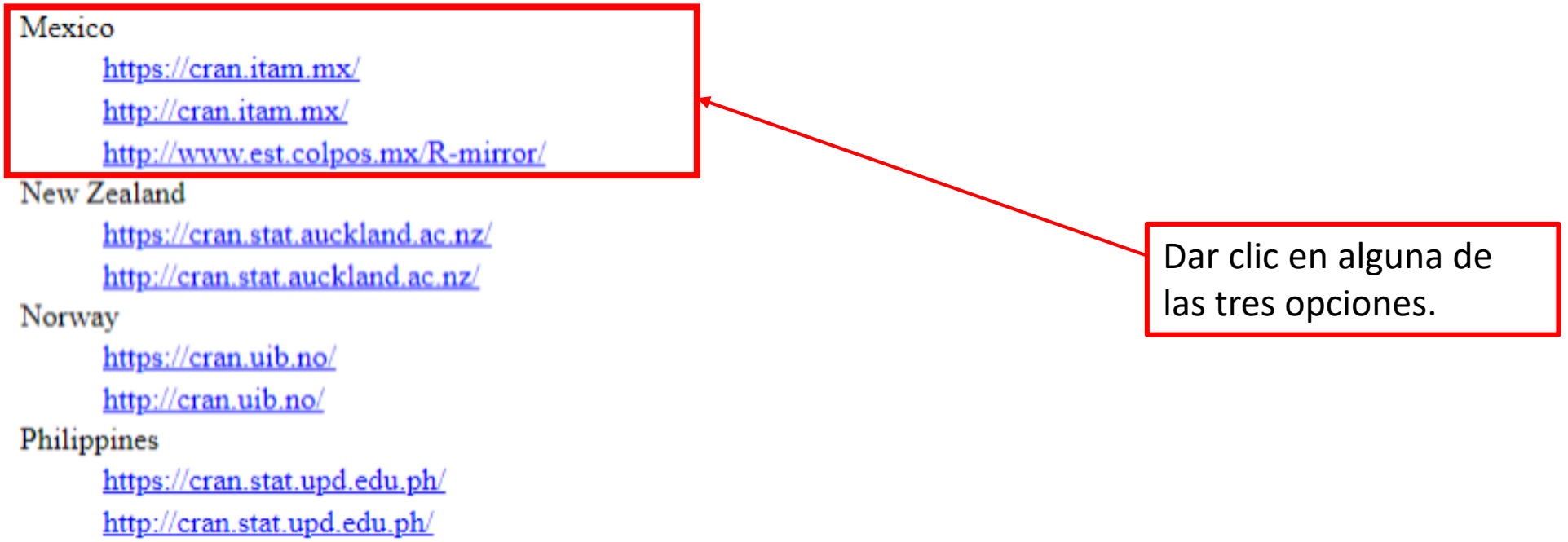
- Ingresar al siguiente vínculo:

  https://www.r-project.org/



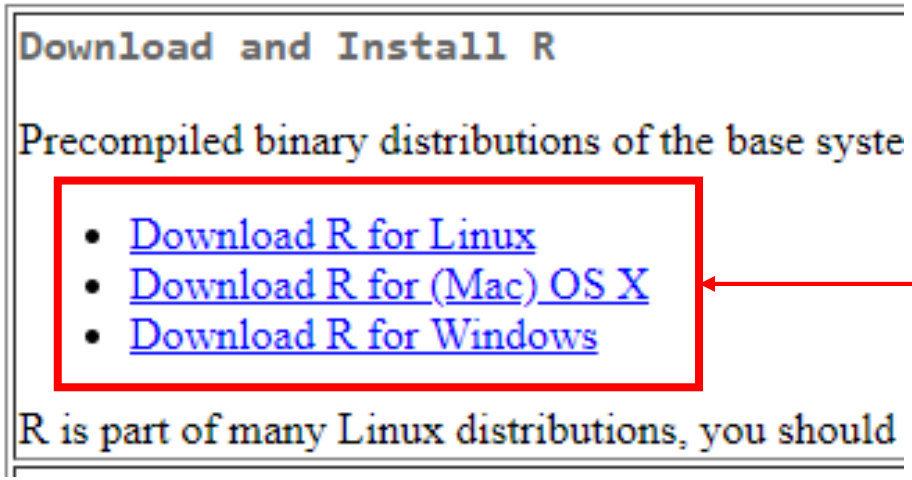
Instalación de R Project

- Buscar la opción de descarga en México:



Instalación de R Project

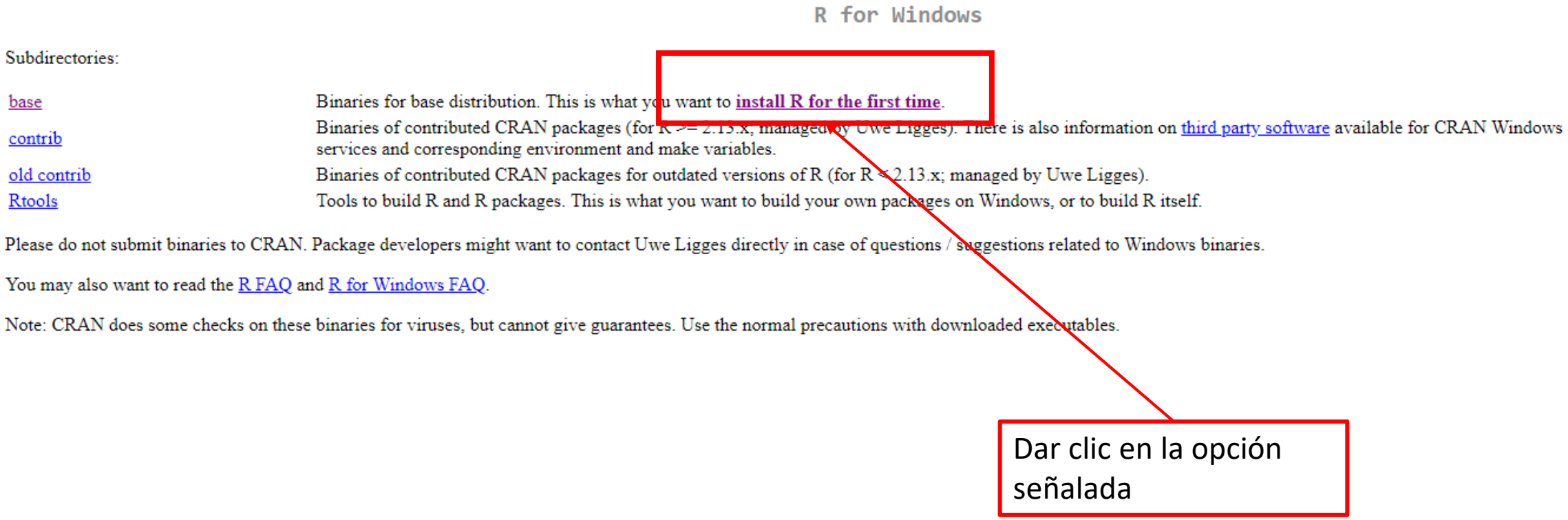
- Elegir el sistema operativo de su preferencia:



Dar clic en alguna de las tres opciones.

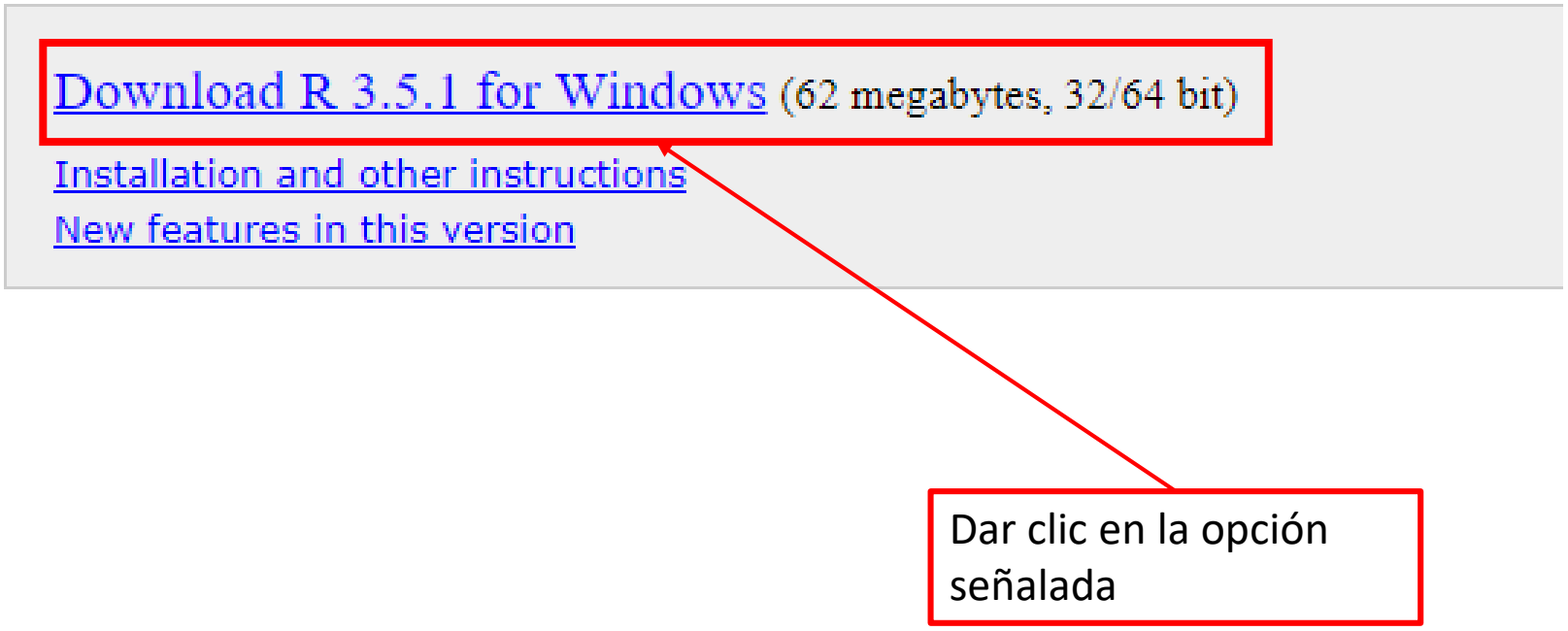
Instalación de R Project

- Si es sobre Windows, dar clic sobre el siguiente vínculo:



Instalación de R Project

- Finalmente elegimos la opción de descarga y seguimos las instrucciones de instalación:



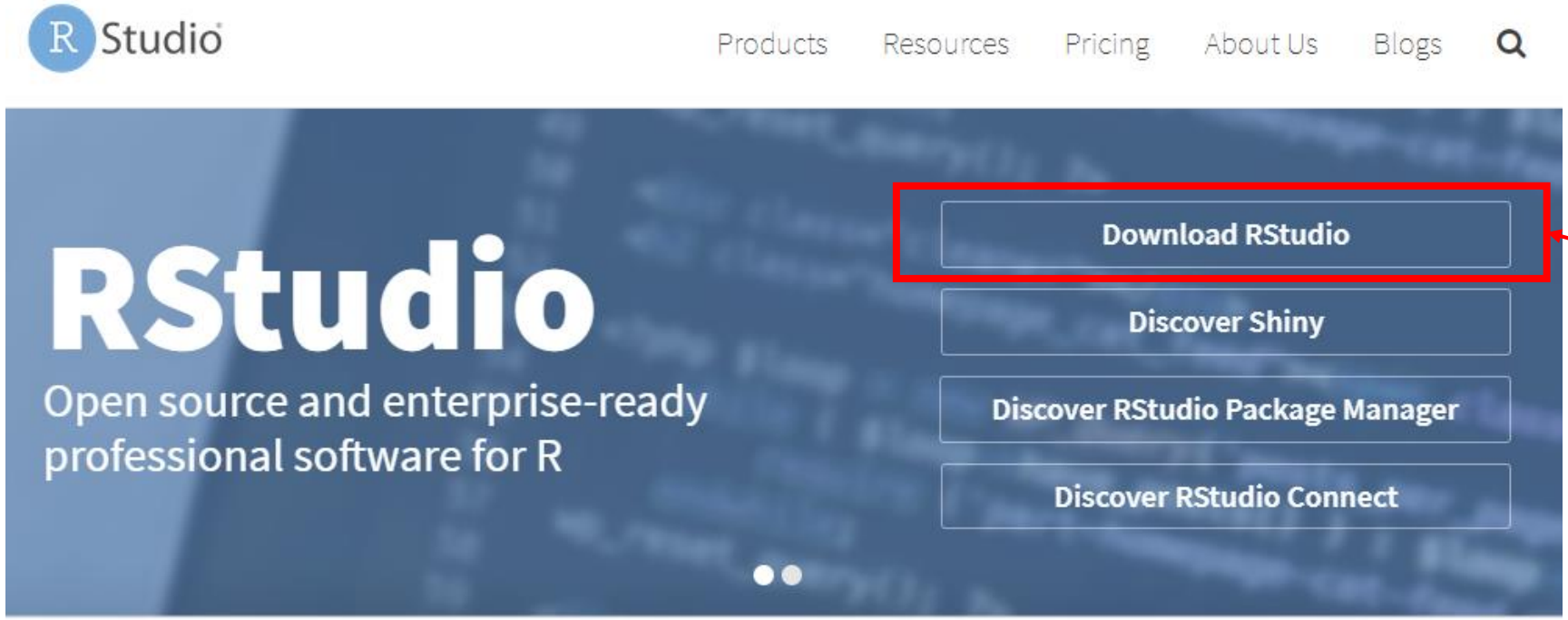
R Project

- R Project es el sistema de archivos **base** que contiene una gran cantidad de paquetes útiles para la manipulación de datos y realizar cálculos matemáticos y sobre todo estadísticos, incluyendo un buen sistema de graficación.

Instalación de R Studio

- Ingresar al siguiente vínculo:

  https://www.rstudio.com/

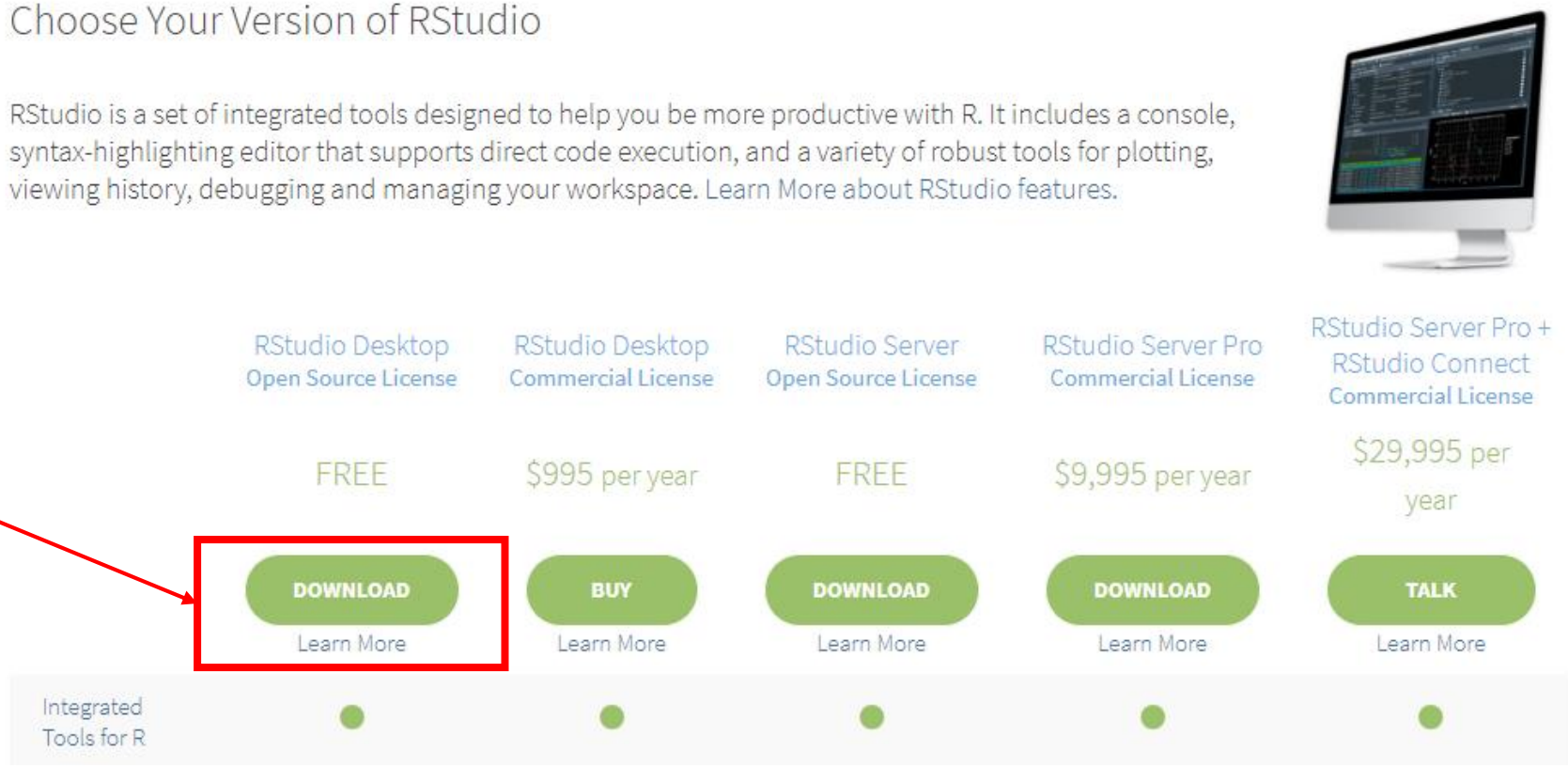


Dar clic en la opción descargar RStudio

Instalación de R Studio

- Seleccionar la opción libre:

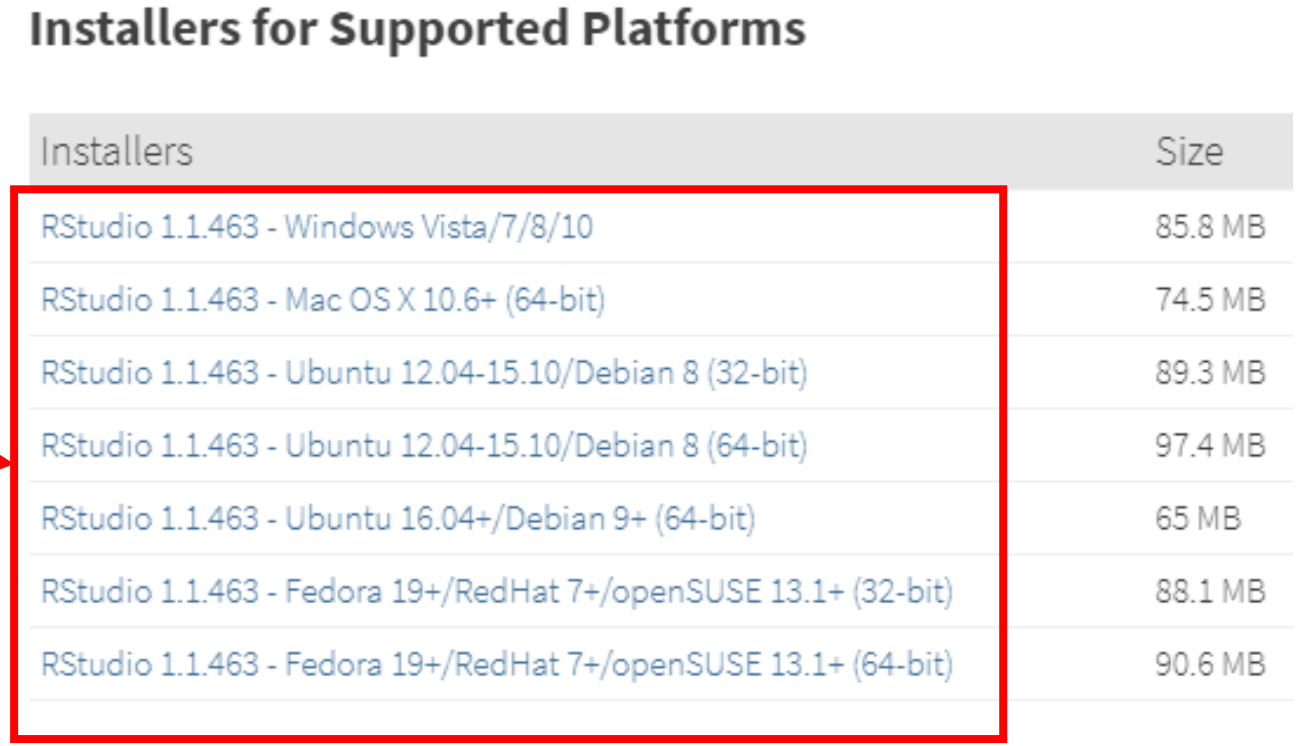
Dar clic en la opción descargar



Instalación de R Studio

- Se elige el sistema operativo de su preferencia, se descargará un archivo y hay que seguir las instrucciones de instalación.

Dar clic en la opción adecuada



Installers for Supported Platforms

Installers	Size
RStudio 1.1.463 - Windows Vista/7/8/10	85.8 MB
RStudio 1.1.463 - Mac OS X 10.6+ (64-bit)	74.5 MB
RStudio 1.1.463 - Ubuntu 12.04-15.10/Debian 8 (32-bit)	89.3 MB
RStudio 1.1.463 - Ubuntu 12.04-15.10/Debian 8 (64-bit)	97.4 MB
RStudio 1.1.463 - Ubuntu 16.04+/Debian 9+ (64-bit)	65 MB
RStudio 1.1.463 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (32-bit)	88.1 MB
RStudio 1.1.463 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (64-bit)	90.6 MB

¡¡¡Mucha atención!!!

Es recomendable hacer primero la instalación de R Project y después la de R Studio, ya que cuando se instala R Studio sin una instalación previa de R Project, al momento de querer abrirlo manda un mensaje de alerta que indica que no encuentra ninguna versión previa de R instalada, esto es porque R Project contiene el sistema base de archivos y paquetes del software y R Studio es prácticamente un ambiente visual de trabajo.

Primeros pasos con R

Creación de un nuevo proyecto

- Abrimos R Studio y observamos sus elementos.
- Seleccionamos al opción File -> New Project…
- Después damos clic en New Directory…
- Elegimos New Project…
- Y colocamos el nombre de nuestro proyecto y asignamos la carpeta donde se guardará dentro de nuestra computadora.
- Finalmente damos clic en el botón Create Project.

Operaciones aritméticas

- R puede parecer una calculadora, ya que se puede realizar cualquier operación básica que se desee.

- El objeto básico y de mayor uso en R se conoce como **vector**, que es un listado de elementos del mismo tipo.

- Hay tres forma de ingresar datos a un vector de forma manual:
  1) datos <- c(1,2,3,4,5,6,7,8,9,10)
  2) datos = c(1,2,3,4,5,6,7,8,9,10)
  3) datos <- scan()

Creación de variables

- Para crear una variable que contenga datos, es decir, un vector, se utiliza generalmente el operador asignación:

<-

- Y la letra c para agregar contenido al vector:

```
x <- c(1,3,5)
```

Tipos de datos

```
> x <- c(0.3, 0.9)       #numérico
> x <- c(FALSE, TRUE)    #lógico
> x <- c(T, F)           #lógico
> x <- c("a", "c", "d") #caracteres
> x <- 19:49             #enteros
> x <- c(1+0i, 2+4i)     #complejos
```

- Con el símbolo # se pueden agregar comentarios en R.

Cargar un listado de elementos desde archivos

- Se puede cargar una serie de datos y asignarlos a una variable. En R generalmente se utilizan archivos de texto(bloc de notas) y archivos csv (del inglés comma separated values) o archivos separados por comas.

- La sintaxis para un archivo csv es la siguiente:

```
datos <- read.csv(file="Datos.csv", head=TRUE, sep=",")
```

file	El nombre y ruta del archivo
head	Si tiene encabezados o no
sep	Tipo de separador

Cargar un listado de elementos desde archivos

El comando **read.table** sirve para poder leer datos desde un archivo externo, dentro del paréntesis va la ruta del archivo que contiene los datos y si la primera línea del archivo contiene el nombre de la variables se utiliza header=T.

- La sintaxis para un archivo de texto es la siguiente:

```
datos <- read.table("C:/Ejemplo/datos.txt", header=T)
```

Ejercicio

- Construye una tabla de frecuencias para datos no agrupados y datos agrupados utilizando Excel para el siguiente conjunto de datos:

32	31	28	29	33	32	31	30
31	31	27	28	29	30	32	31
31	30	30	29	29	30	30	31
30	31	34	33	33	29	29	

Día 2

Coerción

- La coerción se utiliza para cambiar de un tipo de dato a otro en R. Si queremos cambiar el tipo de dato de forma explícita en R utilizamos la función **as.***

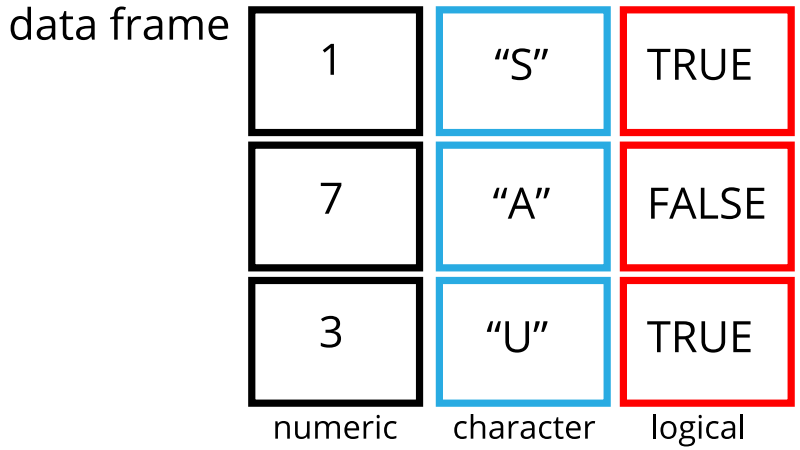
- as.numeric( )
- as.logical( )
- as.character( )
- as.integer( )

Ejemplo de coerción

```
> x <- 0:10
> class(x)
[1] "integer"
> a <- as.numeric(x)
> b <- as.logical(x)
> c <- as.character(x)
> d <- as.integer(x)
```

Data.frame en R

Cuando se realiza un estudio estadístico sobre los sujetos u objetos de una muestra, la información se organiza tabla, es decir, una hoja de datos, en los que cada fila corresponde a un elemento y cada columna a una variable. La estructura de un data.frame es muy similar a la de una matriz. La diferencia es que una matriz sólo admite valores numéricos, mientras que en un data.frame podemos incluir también datos de diferetes tipos.



Data.frame en R

- Ingresa las siguientes líneas de código en R y observa el resultado:

```
edad <- c(22, 34, 29, 25, 30, 33, 31, 27, 25, 25)
tiempo <- c(14.21, 10.36, 11.89, 13.81, 12.03, 10.99, 12.48, 13.37, 12.29, 11.92)
sexo <- c("M","H","H","M","M","H","M","M","H","H")
misDatos <- data.frame(edad,tiempo,sexo)
```

Resolviste el ejercicio y quedó así!!!!

- Construye una tabla de frecuencias para datos no agrupados y datos agrupados utilizando Excel para el siguiente conjunto de datos:

32	31	28	29	33	32	31	30
31	31	27	28	29	30	32	31
31	30	30	29	29	30	30	31
30	31	34	33	33	29	29	

Tabla de datos NO agrupados

Temperaturas	F absoluta	F Acumulada	F Relativa	F Relativa Acumulada
27	1	1	0.032258065	0.032258065
28	2	3	0.064516129	0.096774194
29	6	9	0.193548387	0.290322581
30	7	16	0.225806452	0.516129032
31	8	24	0.258064516	0.774193548
32	3	27	0.096774194	0.870967742
33	3	30	0.096774194	0.967741935
34	1	31	0.032258065	1
	31		**1**	

Tabla de datos agrupados

Clases	Límites		Fronteras		Marcas	F absoluta	F Acumulada	F Relativa	F Relativa Acumulada
1	27	28.1	26.95	28.15	27.55	3	3	0.096774194	0.096774194
2	28.2	29.3	28.15	29.35	28.75	6	9	0.193548387	0.290322581
3	29.4	30.5	29.35	30.55	29.95	7	16	0.225806452	0.516129032
4	30.6	31.7	30.55	31.75	31.15	8	24	0.258064516	0.774193548
5	31.8	32.9	31.75	32.95	32.35	3	27	0.096774194	0.870967742
6	33	34.1	32.95	34.15	33.55	4	31	0.129032258	1
						31		**1**	

Ahora sigue los siguientes pasos para hacerlo en R

1. Carga los datos de las 31 temperaturas mediante un archivo externo csv con la siguiente instrucción:

```
datos <- read.csv(file="Datos.csv", head=TRUE, sep=",")
```

2. Escribe el comando attach() para ingresar a las variables que contiene el archivo de la siguiente forma:

```
attach(datos)
```

3. Ahora escribe **temp** en la línea de comando y observa lo que pasa:

```
temp
```

4. Ejecuta el siguiente comando y observa:

```
table(temp)
```

5. Escribe la siguiente línea y observa:

```
tabla<-as.data.frame(table(datos$temp))
tabla
```

6. Para cambiar el nombre de las variables del encabezado de un data.frame realiza lo siguiente:

```
names(tabla)[1] = ‘temperaturas’
names(tabla)[2] = 'fabsoluta’
tabla
```

7. Ya tienes las temperaturas y las frecuencias absolutas, ahora calcula las frecuencias relativas de la siguiente forma:

```
frelativa <- prop.table(tabla$fabsoluta)
frelativa
```

8. Para calcular la frecuencia acumulada utiliza la función **cumsum()**:

```
facumulada <- cumsum(tabla$fabsoluta)
facumulada
```

9. Falta obtener la frecuencia relativa acumulada, hazlo de la siguiente forma:

```
facumrel <- cumsum(prop.table(frelativa))
facumrel
```

10. Para juntar todas las columnas y pegarlas en una tabla escribe las siguientes líneas:

```
tablafinal <- cbind(tabla, facumulada, frelativa, facumrel)
tablafinal
```

El resultado es el siguiente:

	Temperturas	fabsoluta	facumulada	frelativa	facumrel
1	27	1	1	0.03225806	0.03225806
2	28	2	3	0.06451613	0.09677419
3	29	6	9	0.19354839	0.29032258
4	30	7	16	0.22580645	0.51612903
5	31	8	24	0.25806452	0.77419355
6	32	3	27	0.09677419	0.87096774
7	33	3	30	0.09677419	0.96774194
8	34	1	31	0.03225806	1.00000000

¿Y para datos agrupados?

Tabla de datos agrupados en R

- Primero obtenemos el Rango:

```
range(temp)
```

- Obtenemos el número de clases:

```
nclass.Sturges(temp)
```

- Se determinan los límites de cada intervalo:

```
seq(27,34,length=nclass.Sturges(temp))
```

- Se construyen los intervalos mediante la función **cut()**:

```
intervalos=cut(temp,breaks=seq(27,34,length=nclass.Sturges(temp)),include.lowest=TRUE)
intervalos
```

- Finalmente aplica el siguiente comando y observa el resultado:

```
table(intervalos)
```

¡¡¡Tu trabajo es completar la tabla de frecuencias!!! 

Día 3

Manejo de gráficos en R

Histogramas

Un histograma es una representación gráfica de una variable en forma de barras, en donde la superficie de cada barra es proporcional a la frecuencia de los valores representados. Sirven para obtener una "primera vista" general, o panorama, de la distribución de la población, o de la muestra, respecto a una característica.

En R se utiliza el siguiente comando:

hist()

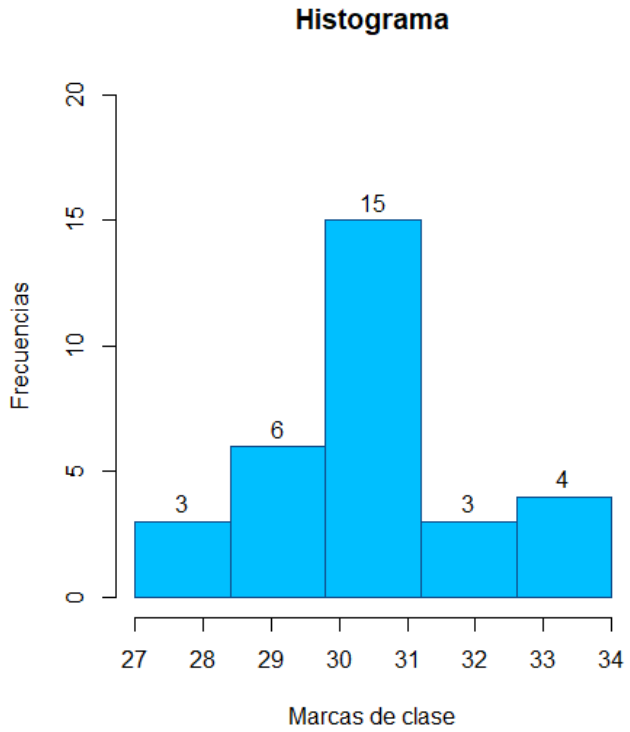
Comando hist()

El comando hist() sirve para realizar la gráfica de un histograma, este comando tiene muchas opciones para editar el gráfico, algunas son las siguientes:

- **main**: Nombre del gráfico.
- **xlab**: Etiqueta del eje x.
- **ylab**: Etiqueta del eje y.
- **labels = TRUE**: muestra los valores de frecuencias sobre la barra.
- **xlim**: Rango de valores que abarcará el eje x.
- **ylim**: Rango de valores que abarcará el eje y.
- **col**: Color de la barras.
- **border**: Color del borde de las barras.

Ejemplo del comando hist()

hist(datos, main = "Histograma", xlab = "Marcas de clase", ylab = "Frecuencias", labels = TRUE, ylim = c(0,20), col = "deepskyblue1", border = "dodgerblue4")



Polígono de frecuencias

- El polígono de frecuencias es una línea que une los puntos medios de la cima de cada barra del histograma. No existe un comando para graficar un polígono de frecuencias en R, pero se puede dibujar una línea sobre el histograma de la siguiente forma:

```
lines(c(min(histo$breaks), histo$mids, max(histo$breaks)), c(0,histo$counts,0), type="l",col="red")
```

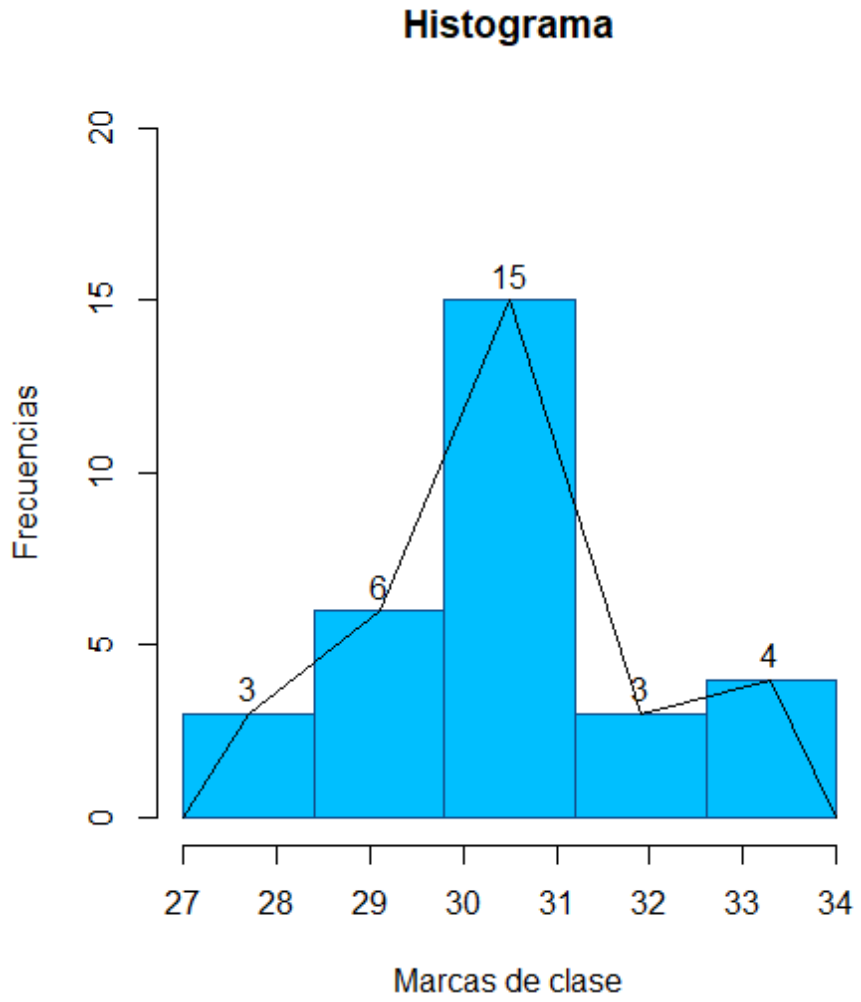
En donde **histo** es un vector que contiene un histograma.

Ejemplo de un polígono de frecuencias

- Primero se debe dibujar el histograma e inmediatamente después la línea sobre las barras, de la siguiente forma:

```
>hist(datos, main = "Histograma", xlab = "Marcas de clase", ylab = "Frecuencias", labels = TRUE, ylim = c(0,20), col = "deepskyblue1", border = "dodgerblue4")

#Asignamos el histograma a una variable

>histo <- hist(datos, main = "Histograma", xlab = "Marcas de clase", ylab = "Frecuencias", labels = TRUE, ylim = c(0,20), col = "deepskyblue1", border = "dodgerblue4")

#Finalmente dibujamos la línea sobre el histograma

>lines(c(min(histo$breaks),histo$mids,max(histo$breaks)),c(0,histo$counts,0),type="l", col="black")
```

El resultado es el siguiente:



Ojiva

- Una ojiva es la distribución de frecuencias y se grafica con la primer frontera inferior y todas las fronteras superiores en el eje x, y para el eje y se colocan las frecuencias acumuladas. **Siempre arranca desde el origen del sistema de referencia.**

- En R no existe un comando para graficar directamente una ojiva, así es que se usa el comando **plot()** combinado con el comando **lines()**

Dibujando la Ojiva

- Primero definimos un vector que contenga las fronteras:

```
fronteras <- c(27, 28.2, 29.3, 30.5, 31.7, 32.8, 34)
```

- Después un vector que contenga las frecuencias acumuladas incluyendo el 0:

```
acumulado <- c(0,3,9,16,24,27,31)
```

- Aplicamos el comando **plot()**:

```
plot(fronteras, acumulado)
```

- Inmediatamente después aplicamos el comando lines():

```
lines(fronteras, acumulado)
```

El gráfico de la ojiva queda así:

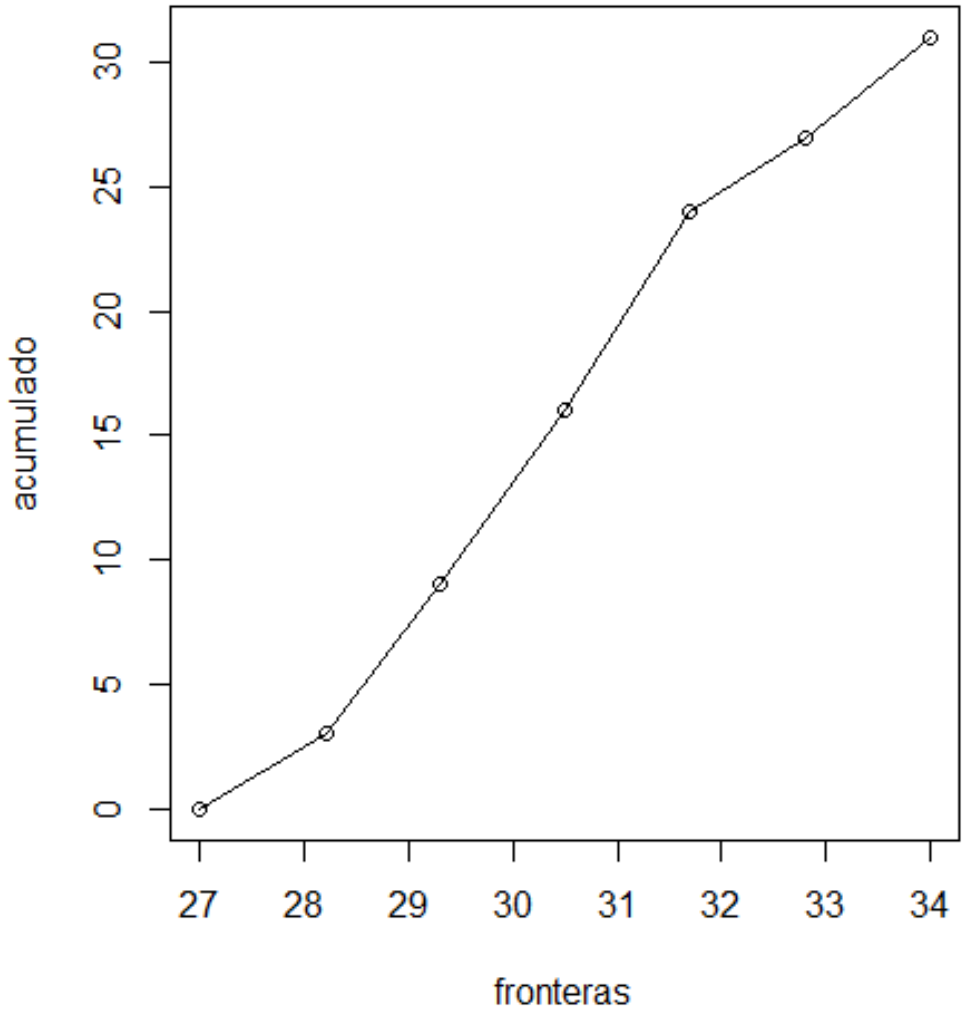


Gráfico de pastel

- Es gráfico de pastel es difícil de manejar en R ya que se tienen que relacionar datos numéricos con datos de tipo categórico. El comando utilizado es **pie()**. Observa las siguientes líneas:

```
valores <- c(10, 12, 4, 16, 8)
paises <- c("Francia", "Alemania", "Cuba", "Mexico", "USA")
pie(valores, labels=paises, col=rainbow(length(valores)), main="Índice de corrupción")
```

Para graficar un pastel en 3D

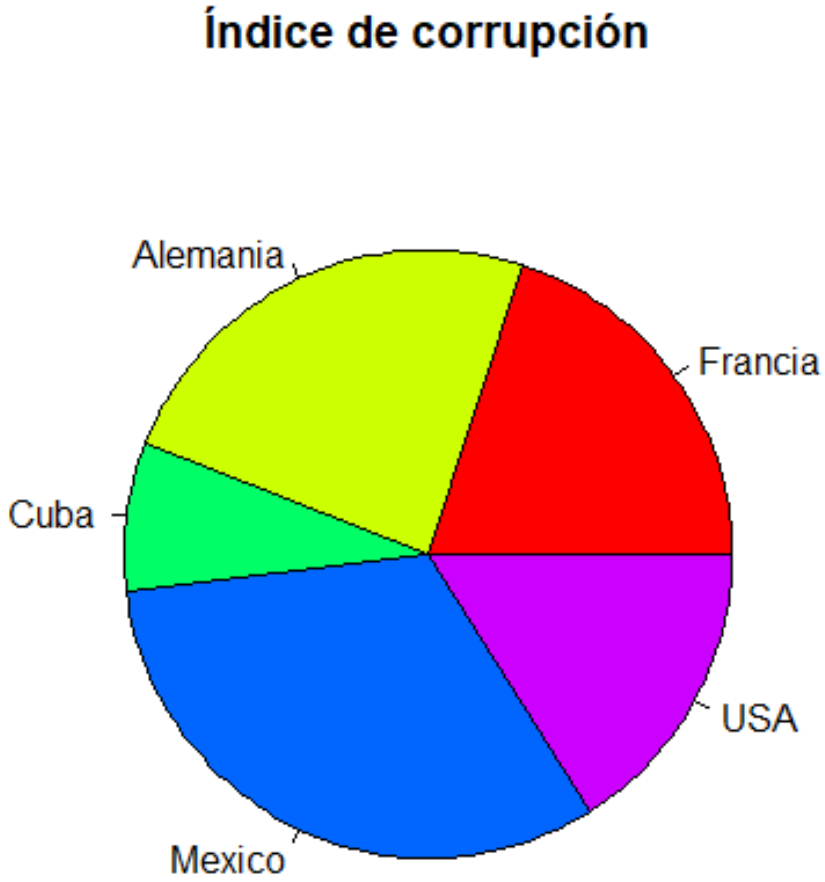
- Escribe el siguiente comando:

```
install.packages("plotrix")
```

- Ahora ingresa de nuevo la siguiente línea de código:

```
pie3D(valores, labels=paises, col=rainbow(length(valores)), main="Índice de corrupción")
```

Las gráficas se muestran así:



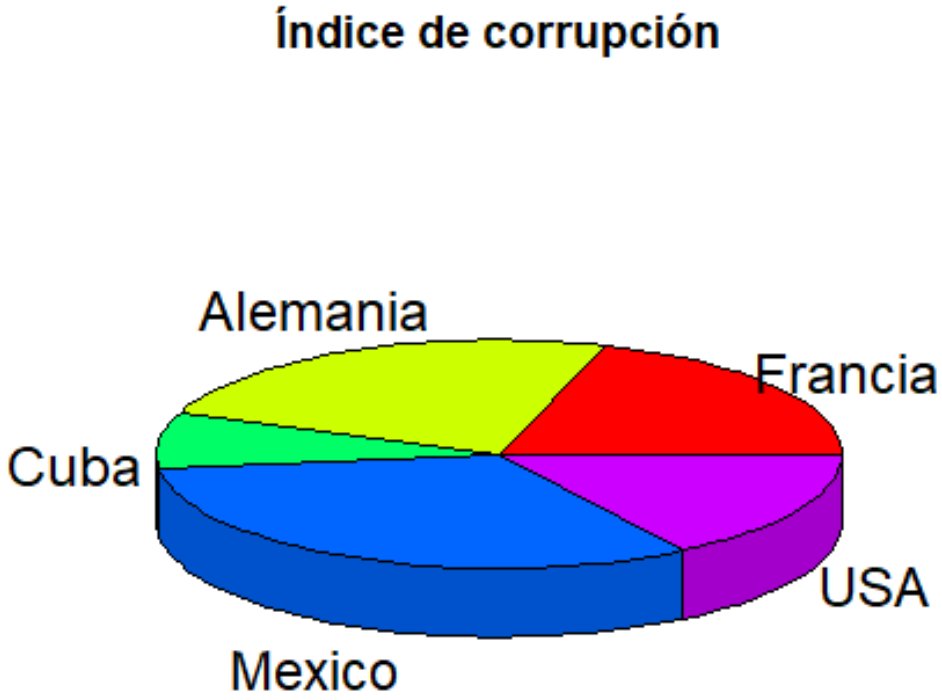


Gráfico de barras

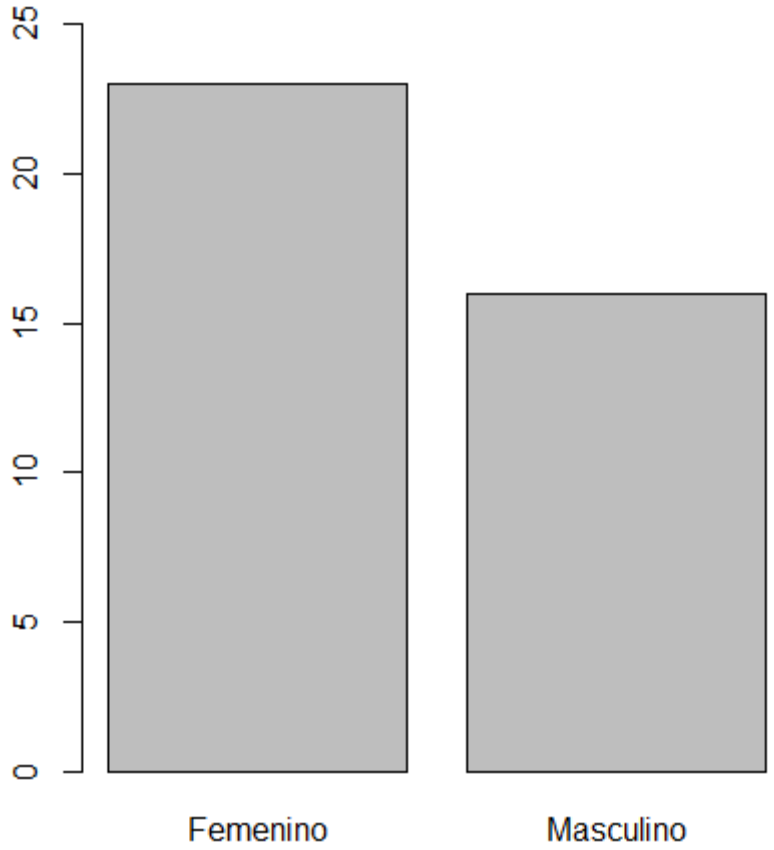
Tal vez es el gráfico que mejor se puede manipular en R, se utiliza el comando **barplot()** y al igual que los otros gráficos cuenta con un gran número de opciones, algunas de ellas se muestran en el siguiente ejemplo:

Carga el archivo encuesta40.cvs de forma gráfica y aplica los siguientes comandos:

```
completa <- data.frame(encuesta40)
contar <- table(completa$genero)
```

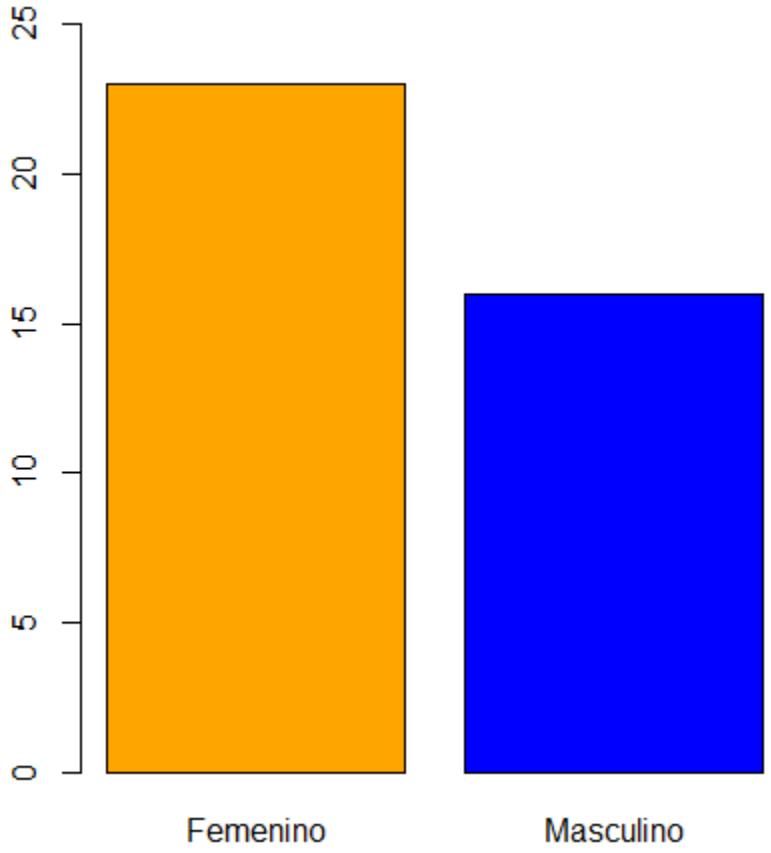
Comencemos a graficar:

barplot(contar, ylim = c(0,25))



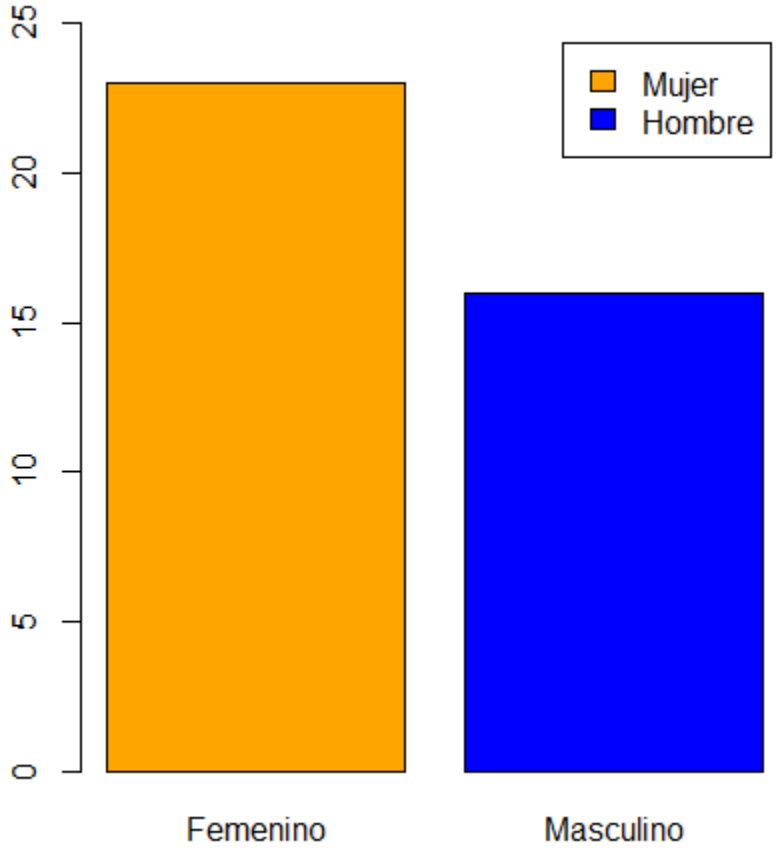
Cambio de color

barplot(contar, ylim = c(0,25), col=c("orange","blue"))



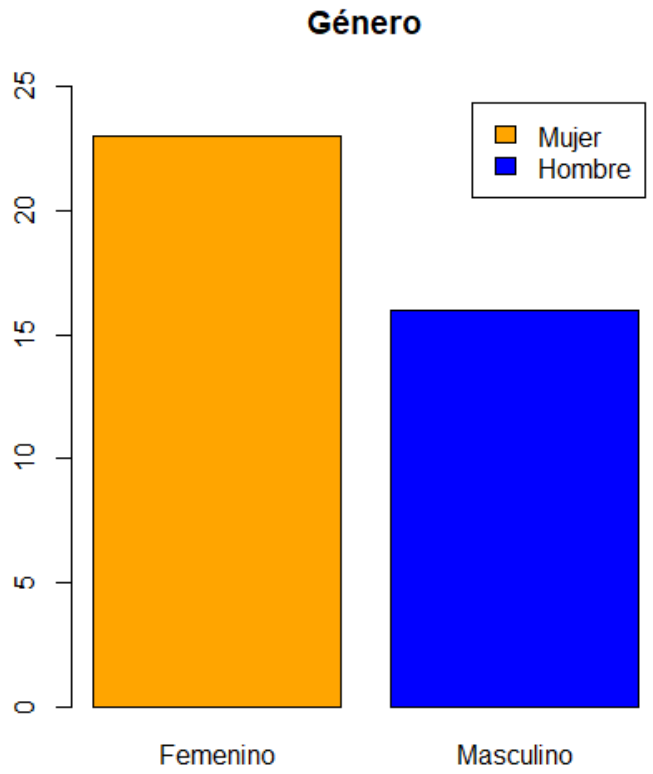
Agregando una leyenda

barplot(contar, ylim = c(0,25), col=c("orange","blue"), legend.text=c("Mujer","Hombre"))



Agregando un título

barplot(contar, ylim = c(0,25), col=c("orange","blue"), legend.text=c("Mujer","Hombre"), main = "Género")



Graficando dos variables

```
contar <- table(completa$genero, completa$area)

barplot(contar, ylim = c(0,25), col=c("orange","blue"),
legend.text=c("Mujer","Hombre"), main = "Género")
```

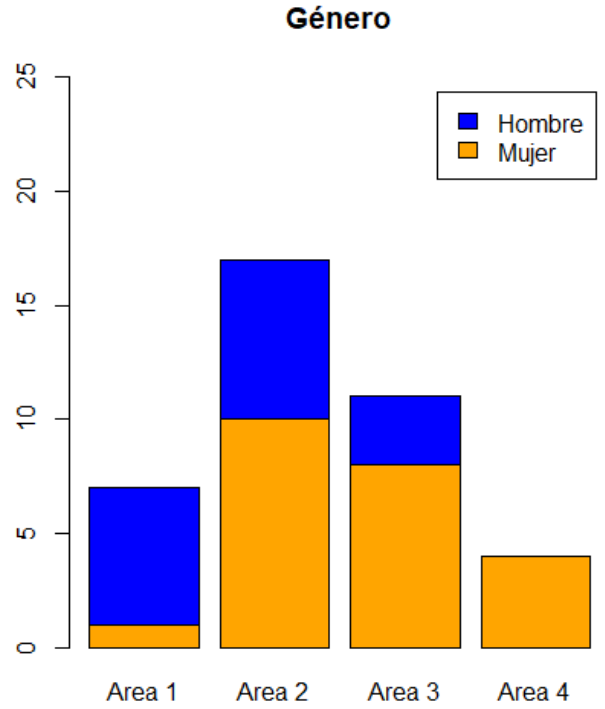


Diagrama de tallo y hojas

- Para obtener una representación de tallo y hojas se utiliza el comando stem() en R.

>stem(aciertos_ingreso)

```
  92 | 000000000000
  94 | 00000
  96 | 0000000000
  98 | 0
 100 | 000
 102 | 00
 104 | 0
 106 | 00
 108 | 0
 110 | 0
 112 | 0
```

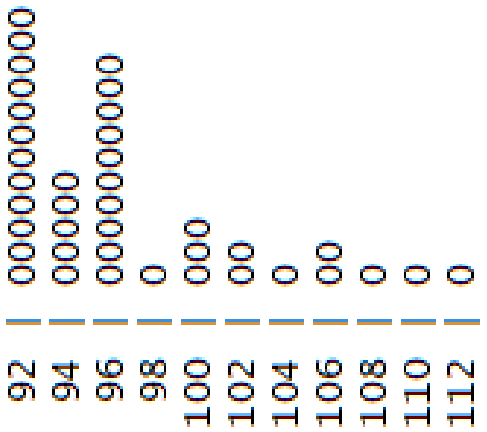


Diagrama de caja y bigotes

- El diagrama de caja es útil para conocer la distribución de los datos, en R se utiliza el comando **boxplot()**.