Clase 1

Basicos

```
TRUE == TRUE
```

```
## [1] TRUE
```

Vectores

¿Qúe es un vector?

La estructura más simple de datos en R.

`x <- c("a", "b", "c")`

Se puede construir un vector de tipo numérico, lógico o carácter. La letra **c** significa "concatenar", y de hecho es un acrónimo para dicha palabra.

Existen cuatro tipos comunes: *logical*, *integer*, *double*, *character*

Propiedades de un vector:

1. Tipo

```
x <- 1:5
typeof(x)
```

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

2. Longitud

```
x <- 1:5
length(x)
```

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

3. Atributos

```
x <- 1:5
class(x)
```

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

Trabajar con vectores

En R se indexa la información desde 1, para acceder a la información de una posición especifica de un vector se usan los [ ].

```
x <- c("Montserrat", "Nidia", "Armando", "Oscar", "Diego")
x[3]
```

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

```
x[c(1,2,3)]
```

```
## [1] "Montserrat" "Nidia"      "Armando"
```

```
x[1:3]
```

```
## [1] "Montserrat" "Nidia"      "Armando"
```

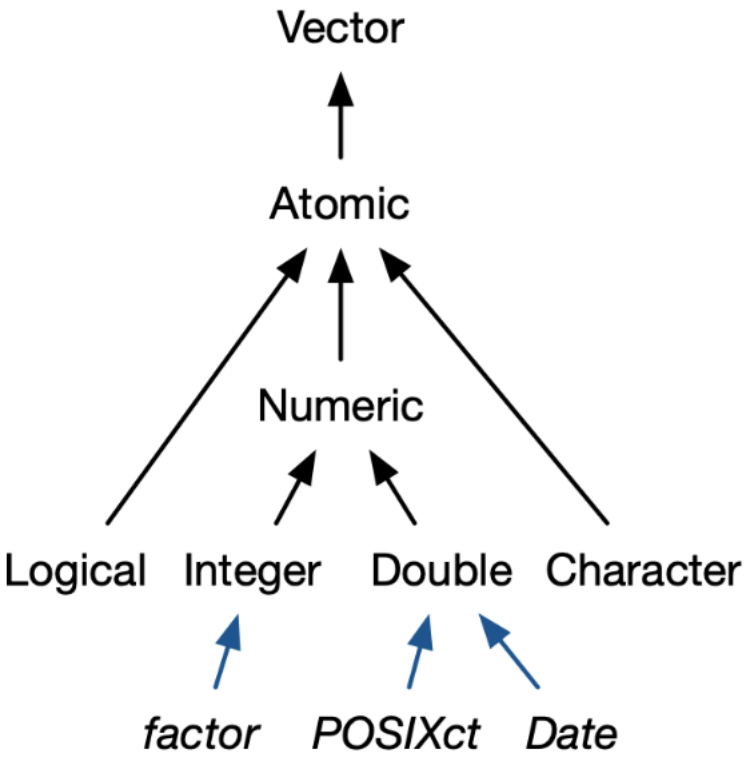

Figure 1: Vector S3

```
x[-3]
```

```
## [1] "Montserrat" "Nidia"      "Oscar"      "Diego"
```

Para agregar un valor al vector se debe tomar en cuenta el tipo de vector. *Nota: si indexo con signo negativo, se elimina esa posición*

```
x[6] <- "Fernanda"
x
```

```
## [1] "Montserrat" "Nidia"      "Armando"    "Oscar"      "Diego"
## [6] "Fernanda"
```

```
x <- x[-6]
x
```

```
## [1] "Montserrat" "Nidia"      "Armando"    "Oscar"      "Diego"
```

Ejercicio

1. ¿Qué pasa si agrego un número a un vector de caracteres?
2. Crea un vector con el nombre *mi_primer_vector* con 10 palabras.

Listas

Pueden tener mas de un objeto de diferente clase, es decir, puedo almanecer dentro de la misma lista objetos de clase caracter, factor y numericos, inclusive pueden añardiser data.frame!!!

```
my_list <- list(c("a", 1) , c("hola", "como", "estas"),
             data.frame(Nombres = c("Nidia", "Mont"), Sexo= c("F", "M")))
my_list
```

```
## [[1]]
## [1] "a" "1"
##
## [[2]]
## [1] "hola"  "como"  "estas"
##
## [[3]]
##   Nombres Sexo
## 1   Nidia    F
## 2    Mont    M
```

Para ver la estructura de cualquir tipo de dato se ocupa el comando **str()**

```
str(my_list)
```

```
## List of 3
##  $ : chr [1:2] "a" "1"
##  $ : chr [1:3] "hola" "como" "estas"
##  $ :'data.frame':    2 obs. of  2 variables:
##   ..$ Nombres: Factor w/ 2 levels "Mont","Nidia": 2 1
##   ..$ Sexo   : Factor w/ 2 levels "F","M": 1 2
```

¿Cómo se accede a los datos de una lista?

¿Cuál es la diferencia entre los siguientes dos codigos?

```
my_list[1]
```

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

```
my_list[[1]]
```

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

Con [[]] accedemos al valor (genera un objeto de menor peso de memoria), y con [] accedemos a la posición de la lista. Para obtener el numero 1 de la posicion 1 de mi lista se hace lo siguiente:

```
my_list[[1]][2]
```

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

Para acceder a la palabra "como" de nuestra lista se hace lo siguiente:

```
my_list[[2]][2]
```

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

Tambien se puede ocupar el simbolo **$** para acceder a los datos de una lista cuando cada valor tiene un nombre

```
my_list_2 <- list(Datos_sin_sentido = c("a", "b", 4, 8), Palabras = c("hola", "como", "estas"),
            Un_data_frame = data.frame(Nombres = c("Nidia", "Mont"), Sexo= c("F", "M")))

my_list_2
```

```
## $Datos_sin_sentido
## [1] "a" "b" "4" "8"
##
## $Palabras
## [1] "hola"  "como"  "estas"
##
## $Un_data_frame
##   Nombres Sexo
## 1   Nidia    F
## 2    Mont    M
```

```
my_list_2$Un_data_frame
```

```
##   Nombres Sexo
## 1   Nidia    F
## 2    Mont    M
```

Ejercicio

1. Buscar como agrego y quito valores de una lista
2. Crea una lista que contenga en la posición 1 un vector con los nombres de los integrantes de la clase; en la posición 2 un vector con las edades de los integrantes de la clase; en la posición 3 un vector con el sexo de los integrantes de la clase. Poner titulo a cada posición.

Data.frame

Un data.frame es una lista con el mismo numero de filas o columnas. Se puede considerar a cada columna como un vector (ya que tienen el mismo tipo de datos) o a cada fila una lista (ya que contiene diferente tipo de datos).

Ejemplo:

```r
my_df <- data.frame(
  Integrantes = c("Nidia", "Monse", "Mont", "Damian"),
  Sexo = c("Femenino", "Femenino", "Femenino", "Masculino"),
  Correo = c("nidibelh@gmail.com",
             "monserrat.urbina.santana@gmail.com ",
             "loredoguillen@gmail.com",
             "damianae.54@gmail.com"),
  Velocidad = c(8, 10, 7, 9)
)

my_df
```

```
##   Integrantes      Sexo                              Correo Velocidad
## 1       Nidia  Femenino                  nidibelh@gmail.com         8
## 2       Monse  Femenino monserrat.urbina.santana@gmail.com         10
## 3        Mont  Femenino             loredoguillen@gmail.com         7
## 4      Damian Masculino               damianae.54@gmail.com         9
```

Para acceder a los datos de un data frame se utiliza [,], en la primera posición (antes de la coma) se especifica el numero o numero de filas, nombre o vector de logicos; en la segunda posición se especifica que columnas se quieren; un espacio en blanco significa que tome todos los valores.

```r
my_df[2,]
```

```
##   Integrantes     Sexo                              Correo Velocidad
## 2       Monse Femenino monserrat.urbina.santana@gmail.com         10
```

```r
my_df[,3]
```

```
## [1] nidibelh@gmail.com                 monserrat.urbina.santana@gmail.com
## [3] loredoguillen@gmail.com            damianae.54@gmail.com
## 4 Levels: damianae.54@gmail.com ... nidibelh@gmail.com
```

¿Por qué sucede esto?

Otra forma de acceder a los datos de un data.frame es con su nombre y el simbolo **$**

```r
my_df$Integrantes
```

```
## [1] Nidia  Monse  Mont   Damian
## Levels: Damian Monse Mont Nidia
```

```r
my_df$Integrantes[3]
```

```
## [1] Mont
## Levels: Damian Monse Mont Nidia
```

Otros caminos

```r
df <- data.frame(x = 1:3, y = 3:1, z = letters[1:3])

df[df$x == 2, ]
```

```
##   x y z
## 2 2 2 b
```

```r
df[c(1, 3), ]
```

```
##   x y z
## 1 1 3 a
```

```
## 3 3 1 c
```

```
df[c("x", "z")]
```

```
##   x z
## 1 1 a
## 2 2 b
## 3 3 c
```

```
df[, c("x", "z")]
```

```
##   x z
## 1 1 a
## 2 2 b
## 3 3 c
```

¿Qué hay de esto?

```
str(df["x"])
```

```
## 'data.frame':    3 obs. of  1 variable:
##  $ x: int  1 2 3
```

```
str(df[, "x"])
```

```
##  int [1:3] 1 2 3
```

Ejercicios

- Buscar como añadir una columna al data.frame
- Como poner nombre a las columnas y a las filas
- Genera un data.frame con los datos de arriba y añadé dos columnas, una con edad y otra con peso.

EXTRA

- ¿Qué es un tibble?
- ¿Qué estructura de datos consume menos memoria al hacerle cambios?