Clase vectores

Práctica 3

Vectores

Los vectores en R son objetos de una dimensión (longitud del vector) que puede contener datos numéricos, cadena de caracteres o datos lógicos, entre otros. Pueden contener elementos de un solo tipo, aunque su tamaño podría ser ilimitado.

Ejemplos:

```
vector_numerico <- c(1, 3, 5, 7)       # vector numérico
vector_texto <-c("arbol", "casa", "pez") # vector de caracteres
vector_logico <- c(TRUE, FALSE, TRUE) # vector lógico
```

¿Podriamos unir/concatener estos vectores?

Vectores

```
vector_numerico <- c(1, 3, 5, 7)       # vector numérico
vector_texto <-c("arbol", "casa", "pez") # vector de caracteres
vector_logico <- c(TRUE, FALSE, TRUE) # vector lógico
```

```
nuevo_vector_1 <- c(vector_texto, vector_numerico)
nuevo_vector_1
```

```
## [1] "arbol" "casa"  "pez"   "1"     "3"     "5"     "7"
```

```
nuevo_vector_2<- c(vector_logico, vector_numerico)
nuevo_vector_2
```

```
## [1] 1 0 1 1 3 5 7
```

Podemos concatenar vectores que sean de distintas clases, pero R va a convertilos a una unica clase cuando los una.

Vectores - Creación

Podemos crear vectores de distintas formas, entre ellas se encuentran las funciones:

- c()
- assign
- vector
- rep
- seq

En general, es conveniente asignar espacio en memoria para vectores antes de empezar a operar. Esto quiere decir, es conveniente definir el vector y la longitud que tendra antes de hacer algun cálculo.

Vectores - Funciones útiles

- sum(): suma todos los elementos del vector. Ejemplo:

```
vector_numerico <- c(1, 3, 5, 7)
sum(vector_numerico)
```

```
## [1] 16
```

- mean(): calcula el valor promedio de los elementos de un vector. Ejemplo:

```
mean(vector_numerico)
```

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

- prod(): producto de todos los elementos del vector. Ejemplo:

```
prod(vector_numerico)
```

```
## [1] 105
```

Vectores - Funciones útiles

- length(): longitud del vector. Ejemplo:

```
vector_numerico <- c(1, 3, 5, 7)
length(vector_numerico)
```

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

- head(): proporciona los primeros n elementos, por default n=6. Ejemplo:

```
vector <- c(5, 8, 1, 10, 7, 4, 1, 20, 25, 2)
head(vector, n=3)
```

```
## [1] 5 8 1
```

- tail(): proporciona los últimos n elementos, por default n=6. Ejemplo:

```
tail(vector)
```

```
## [1]  7  4  1 20 25  2
```

Vectores - Funciones útiles

- sort(): ordena los elementos de un vector de forma ascendente o decreciente. Ejemplo:

```
vector <- c(5, 8, 1, 10, 7)
sort(vector)
```

```
## [1]  1  5  7  8 10
```

```
sort(vector, decreasing = T)
```

```
## [1] 10  8  7  5  1
```

Vectores - Indexación

Se llama indexación a la selección de un elemento de un objeto a partir de sus índices/posiciones.

Ejemplo:

```
a<- seq(0,10, by=2)
a
```

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

```
## Calculo el largo de a
length(a)
```

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

```
## Quiero el cuarto elemento de a
cuarto<- a[4]
cuarto
```

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

```
## Puedo pedir mas de un elemento, por ejemplo los primeros 3 elementos
tres_primeros<- a[1:3]
tres_primeros
```

```
## [1] 0 2 4
```

Vectores - Indexación lógica

La indexación se puede hacer también usando variables lógicas. Hay que tener en cuenta que se van a elegir los elementos asociados a la opción TRUE

Ejemplo:

```
b<- c( 25, 8, 6, 50 , 10, 0)

b[b< 15]
```

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

Vectores - Indexación

la función which permite encontrar las posiciones de un vector que cumplen cierta condición. Ejemplo:

```
a<- seq(0,10, by=2)
which(a>5)
```

```
## [1] 4 5 6
```

Quiere decir que las posiciones 4,5 y 6 de a son mayores a 5, pero ¿qué valores tiene a en esas posiciones?

```
mayores_5<- a[which(a>5)]
mayores_5
```

```
## [1]  6  8 10
```

Vectores - Indexación

Otra forma de obtener los valores de a que sean mayor a 5 seria a partir de la indexación lógica

```
a<- seq(0,10, by=2)



mayores_5<- a[a>5]
mayores_5
```

```
## [1]  6  8 10
```