Estructuras de datos R

Tipos de datos

Un vector es una secuencia ordenada de datos. R dispone de los siguientes tipos de datos:

- **logical**: (True or False)
- **integer**: $\mathbb{Z}$
- **numeric**: $\mathbb{R}$
- **complex**: $\mathbb{C}$

En los vectores de `R`, todos sus objetos han de ser del mismo tipo: todos números, todos palabras, etc. Cuando queramos usar vectores formados por objetos de diferentes tipos, tendremos que usar **listas generalizadas**, `lists` que veremos al final del tema.

Creacions básicas

- **c()**: crear un vector
- **scan(0)**: definir un vector
- **fix(x)**: modificar visualmente un vector `x`
- **rep(a,n)**: crear un vector que contiene el dato a repetido n veces

Ejemplo

```
c(1,2,3)
```

```
[1] 1 2 3
```

```
rep("Mates",7)
```

```
[1] "Mates" "Mates" "Mates" "Mates" "Mates" "Mates" "Mates"
```

Pogresiones y secuencias

Una progresión aritmética es una sucesión de números tales que la **diferencia**, d, de cualquier par de términos sucesivos de la secuencia es constante.

$$a_n = a_1 + (n-1) \cdot d$$

- **seq(a,b,by=d)**: para generear una progresión aritmética de diferencia d que empiezaen a y termina en b

```
seq(5,60,by=5)
```

```
 [1]  5 10 15 20 25 30 35 40 45 50 55 60
```

- **seq(a,b,length.out=n)**: define una progresión aritmética de longitud n que va de a a b con difernecia $d = \frac{b-a}{n-1}$. (n es la cantidad de elementos del array)

```
seq(4, 35, length.out=7)
```

```
[1]  4.000000  9.166667 14.333333 19.500000 24.666667 29.833333 35.000000
```

- **seq(a,by=d, length.out=n)**: define la progresión aritmética de longitud n y diferencia d que empieza en a

```
seq(4, by=25, length.out=3)
```

```
[1]  4 29 54
```

- **a:b**: define la secuencia de números **enteros** ($\mathbb{Z}$) consecutivos entre dos números a y b

```
1:9
```

```
[1] 1 2 3 4 5 6 7 8 9
```

Funciones

Cuando queremos aplicar una función a cada uno de los elementos de un vector de datos, la función `sapply` nos ahorra tener que programar con bucles en `R`:

- **sapply(nombre_del_vector, FUN=nombre_de_la_función)**: para aplicar dicha función a dodos los elementos del vector

```
x = 1:10
sapply(x, FUN=function(x){sqrt(x)})
```

```
 [1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751 2.828427
 [9] 3.000000 3.162278
```

- **sqrt(x)**‘: calcula un nuevo vector con las raíces cuadradas de cada uno de los elementos del vector x

```
x = 1:10
x
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
x + pi
```

```
 [1]  4.141593  5.141593  6.141593  7.141593  8.141593  9.141593 10.141593
 [8] 11.141593 12.141593 13.141593
```

```
x * pi
```

```
 [1]  3.141593  6.283185  9.424778 12.566371 15.707963 18.849556 21.991149
 [8] 25.132741 28.274334 31.415927
```

```
sqrt(x)
```

```
 [1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751 2.828427
 [9] 3.000000 3.162278
```

```
2^x
```

```
 [1]    2    4    8   16   32   64  128  256  512 1024
```

```
x^2
```

```
 [1]   1   4   9  16  25  36  49  64  81 100
```

- **mean(x)**: calcula la media arimtética de las entradas del vector x

```
mean(1:10)
```

```
[1] 5.5
```

- **diff(x)**: cañcula el vector formado por las diferencias sucesivas entre entradas del vector original

```
diff(1:10)
```

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

- **cumsum(x)**: calcula el vector formado por las sumas acumuladas de las entradas del vector original x
  - Permite definir sucesiones descritas mediante sumatorios
  - Cada entrada de *cumsum(x)* es la suma de las entradas de x hasta su posición

```
cumsum(1:10)
```

```
 [1]  1  3  6 10 15 21 28 36 45 55
```

Orden

- **sort(x)**: ordena el vector en orden natural de los objetos que lo forman: el orden numérico creciente, orden alfabético...
- **rev(x)**: invierte el orden de los elementos del vector x

```r
v = c(1,7,5,2,4,6,3)
sort(v)
```

```
[1] 1 2 3 4 5 6 7
```

```r
rev(v)
```

```
[1] 3 6 4 2 5 7 1
```

Otros

- **length(x)** : longitud del vector
- **max(x)** : elemento máximo del vector
- **min(x)** : mínimo del vector
- **sum(x)** : suma de todos los elementos del vector
- **prod(x)** : producto de todos los elementos del vector

```r
x = 1:10
length(x)
```

```
[1] 10
```

```r
max(x)
```

```
[1] 10
```

```r
min(x)
```

```
[1] 1
```

```r
sum(x)
```

```
[1] 55
```

```r
prod(x)
```

```
[1] 3628800
```