VECTORES Y MATRICES

Vector:

Un vector es una lista ordenada de valores, estos valores pueden ser números, palabras o cualquier otra combinación alfanumérica. Los vectores son muy útiles en informática ya que nos permiten guardar una gran cantidad de datos en una sola variable y hacer cálculos de toda índole con ellos como por ejemplo calcular la media de todos los valores (si todos son numéricos) o encontrar valores repetidos. Los vectores se suelen construir hilando valores como puede ser introduciendo uno a uno cada valor o como resultado de un proceso recursivo.

Vector genérico:

V1	V2	V3	V4	V5

V= 1, manzana, 345, 2y36x77, pablo

Matrices:

Una matriz es la versión en dos dimensiones de un vector. En vez de asignar a cada valor sólo una posición en una línea se asignan posiciones en columnas y filas (dos dimensiones), cada columna y fila constituye un vector. Las matrices también presentan mucha utilidad por ejemplo para representaciones gráficas, almacenar datos y realizar cálculos complejos. En informática se pueden construir matrices de muchas formas, se pueden combinar vectores para constituir las columnas o filas de la matriz o se pueden insertar los valores directamente en una matriz nula previamente creada.

Matriz genérica:

	c1	c2	c3	c4	c5
f1	123,	4,	pera,	67xe3,	54
f2	Gff3,	44,	perro,	azul,	65

f3 43, efrt2, 34, unidad, 45678

¿CÓMO EXPRESAR VECTORES?

Hay varias formas de expresar un vector, dependiendo de si previamente conocemos los valores que almacena o no. Dichos valores no tienen por qué ser numéricos, también pueden ser palabras, letras, etc.

1. Si conocemos los valores:
   - Elegimos un nombre para el vector, en este caso w. Este vector w contiene una serie de valores que podemos expresar escribiendo el nombre del vector, w, igualado a c(). Dentro de los paréntesis y separados por comas se escriben los elementos.
   - Ejemplo: w=c("bus",3,4, "coche")
     Si queremos que nos devuelva el vector simplemente hay que escribir el nombre en la consola y devolverá: bus 3 4 coche.
   - Si queremos que nos devuelva solamente un valor del vector debemos introducir w[x], siendo x la posición del elemento que queremos que devuelva. Por ejemplo: w[2] nos devuelve 3.

2. Si no conocemos los valores:
   - Podemos preguntar por los valores que va a contener el vector mediante:

     n = as.numeric(readline(prompt = "Introduce el valor de n: "))

     n va a ser el número de elementos que contiene el vector v y se preguntará al iniciar el programa. Se utiliza as.numeric para que n quede limitado a número, aunque habría otras muchas posibilidades como poner as.integer que solo acepta números enteros.

     v = numeric(n)

     Se declara el vector v.

     for (i in 1:n) {

     v[i] = as.numeric(readline(prompt = "Introduce los valores del vetor: "))
     }

Utilizamos un bucle for para que al preguntar los valores que va a contener v, los vaya almacenando en el vector. Por ello, preguntará un número de valores igual a n.

print (v)

- Ejemplo:

```
1
2  n = as.numeric(readline(prompt = "Introduce el valor de n: "))
3
4  v = numeric(n)
5
6  for (i in 1:n) {
7
8    v[i] = as.numeric(readline(prompt = "Introduce los valores del vetor: "))
9  }
10
11 print (v)
12
```

12:1 (Top Level)

Console | Terminal | Background Jobs

R 4.4.1 · ~/

```
> source("~/Programación R/for_2.R")
Introduce el valor de n: 3
Introduce los valores del vetor: 4
Introduce los valores del vetor: 1
Introduce los valores del vetor: 2
[1] 4 1 2
```

OPERACIONES CON VECTORES

Siendo los vectores: v=c(8,3,4,5) y w=c(1,2,3,4)

1. **Suma y resta:**
   Se suman las componentes: v+w=(8+1, 3+2, 4+3, 5+4) = (9, 5, 7, 9).

2. **Multiplicación por componentes:**
   Se multiplica componente a componente: v*w=(8*1, 3*2, 4*3, 5*4) = (8, 6, 12, 20).

3. **Producto escalar:**
   v%*%w=(8*1 + 3*2 + 4*3 + 5*4) = 46

EJEMPLOS DE USO DE MATRICES

1.

```
# Se pide el valor  de n, el cual está obligado a ser un número entero (por eso el as.integer)

n <- as.integer(readline(prompt = "Introduce el valor de n: "))


# Se declaran los vectores v y w

v <- numeric(n)

w <- numeric(n)


# Se calculan los vectores por medio de bucles for

for (i in 1:n) {

 v[i] <- i^i

 w[i] <- sqrt(i)

}
```

Se establece la matriz M, cuya función (matrix) necesita de 3 variables. El 0 se usa para declarar la ausencia de valores iniciales en filas/columnas. En este caso, la matriz está obligada a ser cuadrada ya que tanto filas como columnas son el valor de n. Así, si queremos una matriz cuadrada 2x2, al inicio del programa estableceremos el valor de n en 2. En otros casos, nos podemos encontrar con X = (0,n,m), donde n hace referencia al número de filas y m al de columnas.

```
M <- matrix(0, n, n)


# Se rellena M con los vectores.
```

Se debe de introducir en 2 bucles for ya que i es el nº de vueltas a realizar para rellenar las filas y j el nº de vueltas a realizar para rellenar las columnas.

```
for (i in 1:n) {

 for (j in 1:n) {

  M[i, j] <- v[i] + w[j]
```

```
 }
}
print(M)
```

Si queremos invertir una matriz:

```
a= solve(M)
```

La función “solve” realiza la inversa de la matriz.

Si, por ejemplo estamos realizando un código para el polinomio interpolador, y queremos calcular los coeficientes:

```
a = solve(M)%*%b
```

%*% se usa para multiplicar matricialmente la inversa de M y b

b es un vector o matriz de términos independientas