



# Computación I

## Curso 2023

Facultad de Ingeniería  
Universidad de la República

# Ejercicio 1

Buscar la  $n$ -ésima ocurrencia de un elemento en un vector y devolver su posición. Si el elemento no pertenece al vector o está menos de  $n$  veces, devolver  $-1$ .

# Ejercicio 1

## Solución

```
function res = n_ocurr(n,elem,v)
largo = length(v)
if largo == 0
    y = -1;
elseif n == 1 & v(1) == elem
    y = 1;
else
    if v(1) == elem
        res = n_ocurr(n-1, elem,v(2:largo));
    else res = n_ocurr(n, elem,v(2:largo));
    end
    if res ~= -1
        res = res + 1;
    end
end
end
```

# Ejercicio 1

## Solución “optimizada”

```
function res = n_ocurr(n,elem,v)
largo = length(v)
if largo < n || largo == 0
    y = -1;
elseif n == 1 & v(1) == elem
    y = 1;
else
    if v(1) == elem
        res = n_ocurr(n-1, elem,v(2:largo));
    else res = n_ocurr(n, elem,v(2:largo));
    end
    if res ~= -1
        res = res + 1;
    end
end
end
```

# Ejercicio 1

## Solución “optimizada” con largo 0 como PB

```
function res = n_ocurr(n,elem,v)
largo = length(v)
if n == 0
    y = 0;
elseif largo < n || largo == 0
    y = -1;
else
    if v(1) == elem
        res = n_ocurr(n-1, elem,v(2:largo));
    else res = n_ocurr(n, elem,v(2:largo));
    end
    if res ~= -1
        res = res + 1;
    end
end
end
```



# Ejercicio 2

Intercalar ordenadamente los elementos de dos vectores ordenados.

# Ejercicio 2

```
function v = intercalar(v1,v2)
if length(v1)==0
    v = v2;
elseif length(v2)==0
    v = v1;
elseif v1(1) < v2(1)
    v = [v1(1), intercalar(v1(2:length(v1)),v2)] ;
elseif v2(1) < v1(1)
    v = [v2(1), intercalar(v1,v2(2:length(v2)))] ;
else v = [v1(1), intercalar(v1(2:length(v1)),
                            v2(2:length(v2)))]
end
```

# Ejercicio 3

Implementar en Octave la suma de los elementos de una matriz en forma recursiva.

Se puede procesar de a una fila o de a una columna. En este caso lo resolveremos procesando de a una fila, usando una función auxiliar también recursiva.



# Ejercicio 3

## Solución

```
function y=sumaMatriz(M)
    [m,n]=size(M);
    if m==0
        y=0;
    else
        y=sumaVector(M(1,1:n)) +
            sumaMatriz(M(2:m,1:n));
    end
```

# Ejercicio 3

## Solución

```
function y=sumaVector(v)
    n=length(v);
    if n==0
        y=0;
    else
        y=v(1) + sumaVector(v(2:n));
    end
```

# Ejercicio 4

Implementar en Octave la suma de los elementos de una matriz en forma recursiva.

Si la matriz es cuadrada y su lado es potencia de 2, se puede partir la matriz en cuatro cuadrados en cada paso.

# Ejercicio 4

## Solución

```
function y=sumaMatriz (M)
    [m,n]=size (M) ;
    if m==0
        y=0;
    elseif m==1
        y=M(1,1) ;
    else
        y=sumaMatriz (M(1:m/2,1:n/2)) +
            sumaMatriz (M(1:m/2,n/2+1:n)) +
            sumaMatriz (M(m/2+1:m,1:n/2)) +
            sumaMatriz (M(m/2+1:m,n/2+1:n)) ;
    end
```