6. Variables lógicas e instrucciones condicionales.

- Variables lógicas
- Operadores lógicos
  - all
  - any
  - and
  - or
  - find
- Sentencias condicionales: `if … elseif`
- Sentencias condicionales: `switch`
- Ejercicios

Variables lógicas

En matlab las variables lógicas son aquellas que toman dos valores: true (que se codifica como 1) y false (que se codifica como 0).

Ejemplo: el siguiente vector toma valores lógicos:

```
>> A=[true true true]
```

```
A =


  1  1  1
```

En la práctica los valores lógicos se producen al hacer comparaciones. Por ejemplo, si definimos las variables:

```
>> a=7;
 b=5;
 c=7;
```

podemos compararlas entre sí, y la respuesta será un valor lógico. El operador *"¿es igual que?"* es el doble signo igual "=="

```
>> a==b
```

```
ans = 0
```

```
>> a==c
```

```
ans = 1
```

También podemos compararlas para ver si son distintas. El operador *"¿es distinta de"* es el símbolo igual precedido de una tilde: "~="

```
>> a~=b
```

```
ans = 1
```

NOTA: La tilde (~) en matlab significa negación. En la mayoría de los teclados (en español) se obtiene este símbolo pulsando las teclas `Alt Gr-4`. A veces se puede generar este símbolo pulsando la tecla `Alt` y (sin soltarla) tecleando los números 126 en el teclado numérico de la derecha.

Las comparaciones pueden hacerse también sobre vectores o matrices; en este caso la comparación se hace término a término y matlab devuelve un vector (o matriz) de unos y ceros. Por ejemplo:

```
>> u=[1 2 3 2];
 v=[2 2 2 2];
 u==v
```

```
ans =

  0  1  0  1
```

Operadores lógicos

Matlab cuenta con varios operadores que actúan sobre variables lógicas:

all

`all`: comprueba si **todos** los valores de una operación lógica son verdad.

```
>> all(u==1)
```

```
ans = 0
```

```
>> all(v==2)
```

```
ans = 1
```

```
>> all(u==v)
```

```
ans = 0
```

any

any: comprueba si **alguno o algunos** de los valores de una operación lógica son verdad.

```
>> any(u==1)
```

```
ans = 1
```

```
>> any(u==5)
```

```
ans = 0
```

```
>> any(u==v)
```

```
ans = 1
```

and

and: comprueba si dos valores lógicos son ambos verdaderos; se puede para ello usar el operador &:

```
>> a=7; b=5; c=7;
 a==c & b~=3
```

```
ans = 1
```

```
>> a==c & b~=5
```

```
ans = 0
```

o de modo equivalente se puede usar la función and:

```
>> and(a==c, b~=3)
```

```
ans = 1
```

```
>> and(a==c, b~=5)
```

```
ans = 0
```

or

`or`: comprueba si dos valores lógicos son ambos verdaderos; se puede usar para ello el operador `|`:

```
>> a==c | b~=3
```

```
ans = 1
```

```
>> a==c | b~=5
```

```
ans = 1
```

De modo equivalente se puede usar la función `or`:

```
>> or(a==c, b~=3)
```

```
ans = 1
```

```
>> or(a==c, b~=5)
```

```
ans = 1
```

find

`find` encuentra índices y valores de elementos verdaderos; por ejemplo, en el siguiente vector podemos encontrar qué posiciones ocupan los valores mayores que 5:

```
>> A = [2 8 5 9 12 3 2];
 find(A>5)
```

```
ans =

   2   4   5
```

Si aplicamos `find` a una matriz, usaremos la siguiente notación para que nos devuelva las filas y columnas donde se producen valores verdaderos:

```
>> A=[1 2 3; 4 5 6]
 B=[2 7 3; 8 5 9]
 [fila,columna]=find(A==B);
 [fila,columna]
```

```
A =

   1   2   3
   4   5   6

B =

   2   7   3
```

```
   8   5   9

ans =

   2   2
   1   3
```

Vemos que las matrices A y B coinciden en las posiciones (2,2) y (1,3)

Ver más información sobre variables y operaciones lógicas en el sitio web de Mathworks

Sentencias condicionales: `if` … `elseif`

Las sentencias condicionales en Matlab permiten hacer cosas (cálculos, gráficas, salidas por pantalla, …) sólo si se cumple determinada condición. En general son de la forma:

```
if (condición)
  < hacer cosas >
end
```

Ejemplo:

La función `rem(a,b)` calcula el resto de dividir `a` por `b`. La siguiente sintaxis emplea esta función y utiliza el condicional `if` para comprobar si un número es par:

```
a=8

if (rem(a,2)==0)
  disp('a es par')
end
```

El resultado es:

```
a =  8
a es par
```

La sentencia `if` puede acompañarse de `elseif` y de `else`:

```
a=9

if (rem(a,2)==0)
  disp('a es par')
elseif (rem(a,3)==0)
  disp('a es impar y divisible por 3')
else
  disp('a es impar no divisible por 3')
end
```

```
a =  9
a es impar y divisible por 3
```

Sentencias condicionales: `switch`

Cuando hay que comprobar muchas condiciones, en lugar de `if` a veces resulta más cómodo utilizar `switch`, cuya sintaxis es de la forma:

```
switch (X)
  case 1
    Hacer algo;
  case 2
    Hacer algo distinto;
  otherwise
    Hacer otra cosa;
end
```

En cada linea etiquetada como `case` se compara X con el valor que se especifica. La sintaxis de switch es equivalente a una colección de `if` - `elseif` encadenados.

Por ejemplo, si ejecutamos el código:

```
X=2

switch (X)
  case 1
    disp("X vale 1");
  case 2
    disp("X vale 2");
  otherwise
    disp("X no es ni 1 ni 2");
end
```

obtenemos como resultado

```
X =  2
X vale 2
```

Los valores especificados en `case` no tienen que ser necesariamente numéricos, también podrían ser caracteres:

```
X='c'

switch (X)
  case 'a'
    disp("X es la letra a");
  case 'b'
    disp("X es la letra b");
  otherwise
    disp("X no es ni a ni b");
end
```

```
X = c
X no es ni a ni b
```

Ejercicios