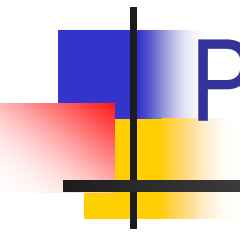
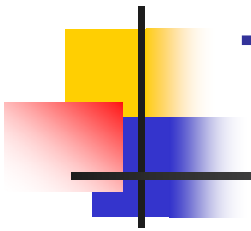


5 Control de Flujo de Programas

Prof. Javier Cañas



Temario

- Introducción
- Decisiones
- Operadores lógicos y relacionales
- Sentencias if y switch
- Sentencias for y while

Introducción

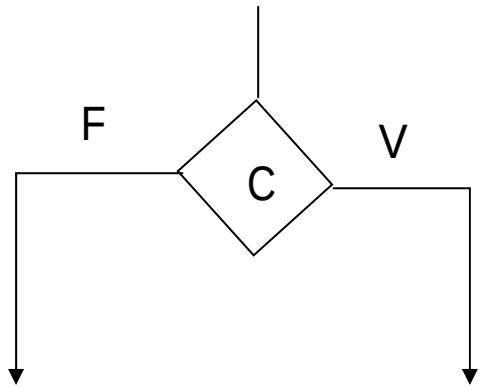
- Este capítulo introduce conceptos de programación general, es decir, aspectos que son comunes a todos los lenguajes de programación.
- Veremos sentencias que permiten tomar decisiones y sentencias que permiten realizar iteraciones.

...Introducción

- Al final de este capítulo se aprenderá a utilizar Matlab como cualquier lenguaje de programación

Decisiones

- Las decisiones permiten realizar distintas acciones dependiendo de una condición lógica.



Condiciones

- Las condiciones lógicas son expresiones que se forman utilizando operadores relacionales y conectivos lógicos.

Operadores relacionales

- El lenguaje de programación de MATLAB dispone de los siguientes operadores relacionales:

  < menor que

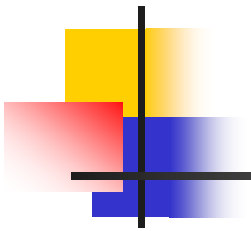
  > mayor que

  <= menor o igual que

  >= mayor o igual que

  == igual que

  ~= distinto que



...Operadores relacionales

- Se pueden utilizar operadores relacionales para comparar escalares, dos arreglos del mismo tamaño o un arreglo con un escalar.

Ejemplo

```
>> A=[-6 4 1 ; 19 -2 80; -2 -3 6]

A =

   -6    4    1
   19   -2   80
   -2   -3    6

>> A>1

ans =

    0    1    0
    1    0    1
    0    0    1
```

Operadores lógicos

- Se considera como verdadero cualquier valor distinto de cero.
- Valor falso es el cero

...Operadores lógicos

Operador	***Descripción***
&	and
|	or
~	not

Ejemplos: Tablas de verdad

A	B	~A	A|B	A&B	xor(A,B)
0	0	1	0	0	0
0	1	1	1	0	1
1	0	0	1	0	1
1	1	0	1	1	0

Control de Flujo

```
if expresiones lógicas
      Sentencias
end
```

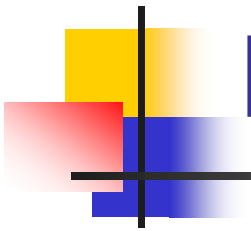
Ejemplo:

```
if d < 40
  cuenta=cuenta+1
   disp(d)
end
```

Ejemplos

- Por ejemplo si el intervalo de una variable es menor que uno fijar el valor de xinc a intervalo/10; sino, fijar el valor de xinc a 0.1.

```
if interval < 1
    xinc = interval/10;
else
    xinc = 0.1;
end
```



Ejemplos

- Cuando muchos niveles de if-else están anidados, resulta difícil determinar cual expresión lógica debe ser verdadera (o falsa). En este caso la construcción elseif ayuda a clarificarla lógica del programa.
- La siguiente diapositiva muestra un ejemplo.

Ejemplo: elseif

```
if temperature > 100
disp(‘Muy alta temperatura – mal funcionamiento.’)
elseif temperature > 90
disp(’Rango normal.’)
elseif temperature > 50
disp(‘bajo el rango normal.’)
else
disp(’muy frío – apagar equipo.’)
end
```

Sentencia IF: Condiciones

- Una observación muy importante: la condición del ***if*** puede ser una *condición matricial*, del tipo **A**==**B**, donde **A** y **B** son matrices del mismo tamaño.
- Para que se considere que la *condición* se cumple, es necesario que sean *iguales de dos a dos* ***todos*** *los elementos* de las matrices **A** y **B**.
- Una condición en la forma A~=B exige que todos los elementos sean diferentes dos a dos. Bastaría que hubiera dos elementos iguales para que la condición no se cumpliese

La sentencia switch

La sentencia ***switch*** realiza una función análoga a un conjunto de ***if...elseif*** concatenados. Su forma general es la siguiente:

```
switch expresión
case case_expr1,
bloque1
case {case_expr2, case_expr3, case_expr4,...}
bloque2
...
otherwise, % opción por defecto
bloque3
end
```

La sentencia switch: ejemplo

```
c=input('Ingrese opcion: ');
switch c,
    case 1,
        disp('valor uno')
    case {2, 5, 13}
        disp('valor 2, 5 o 13')
    case 4,
        disp('valor 4')
    otherwise,
        disp('ninguno')
end
```

La sentencia for

- El for permite ejecutar un grupo de sentencias un número determinado de veces.
- Su estructura es:

```
for i=1:n
sentencias
end
```

...La sentencia for

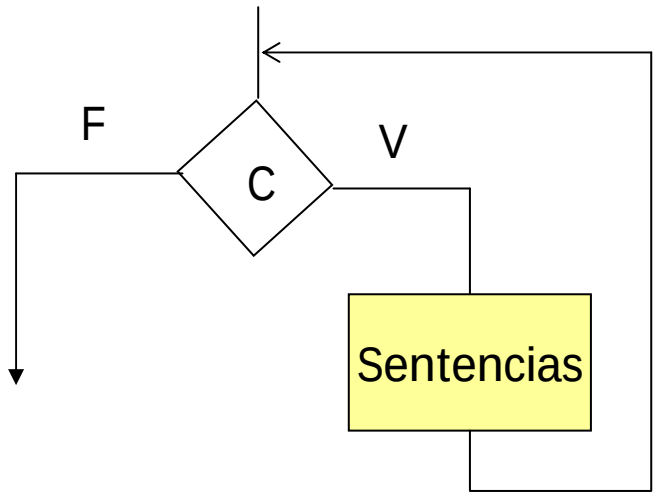
- La variable i también puede ser un vector. Consideremos el siguiente ejemplo:

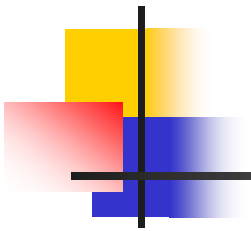
```
A =
     1     2     3
     4     5     6

>> for i=A
disp(i)
end
      1
      4

      2
      5

      3
      6
```

En cada iteración toma una columna de A

La sentencia while

- La sentencia while tiene el siguiente diagrama de flujo:





... La sentencia while

- La estructura es:

```
while Condicion
      sentencias
end
```

Ejemplo

- Construir una función para calcular:

$$sencuad(n) = \sum_{i=1}^{n} \sin^2(i) = \sin^2(1) + \sin^2(2) + ...\sin^2(50)$$

Ejemplo...

```
function f=sencuad(n)
s=0
for i=1:n
    s=s+sin(i)^2
end
f=s
```

Subfunciones

- Las ***sub-funciones***, son funciones adicionales definidas en un mismo archivo ****.m***, con nombres diferentes del nombre del archivo (y del nombre de la función principal).
- Las sub-funciones sólo pueden ser llamadas por las funciones contenidas en ese archivo, resultando “invisibles” para otras funciones externas.

....Subfunciones: ejemplo

- un archivo llamado ***mi_fun.m***:

```
function y=mi_fun(a,b)
y=subfun1(a,b);
function x=subfun1(y,z)
x=subfun2(y,z);
function x=subfun2(y,z)
x=y+z+2;
```

Ejercicio 1

- Construir la función:
  - es_primo(n): esta función devuelve el valor 1 si n es un número primo y 0 en caso contrario

Solución 1

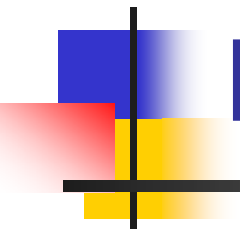
```
function x=es_primo(n)
for i=2:n/2
    if rem(n,i) == 0  % resto
        x=0;
        return
    end
end
x=1;
```

Ejemplo 2

- Utilizando la función anterior construir una función que determine si todos los elementos de un arreglo son números primos.

Solución 2

```
function x=matriz_prima(A)
[m n]=size(A);
for i=1:m
    for j=1:n
        if ~es_primo(A(i,j))
            x=0;
            return
        end
    end
end
x=1;
```



Fin