Fundamentos de Computación

Bloque II.

Primeros Scripts en Octave. Sentencias de Control: Condicionales / Ciclos

Tipos de datos. Cálculo matricial con Octave. Operando con strings

Sixto Herrera García

Departamento de Matemática Aplicada y Ciencias de la Computación

Fundamentos de Computación

Primeros Scripts en Octave
Sentencias de Control: Condicionales

Operadores Básicos: Aritméticos

Operación	Símbolo	Ejemplo
Suma	+	5 + 3
Resta	-	5 - 3
Multiplicación	*	5 * 3
División derecha	/	5 / 3
División izquierda	\	5 \ 3 = 3 / 5
Exponenciación	^	5^ 3 (5^3=125)

Además de las operaciones aritméticas básicas, Octave/Matlab tiene implementadas un gran número de operaciones (trigonométricas, logarítmicas, etc.). A modo de ejemplo, escribir en la terminal de Octave los siguientes comandos:

help log

help sin

Ejemplos: Operadores Aritméticos

Dar el resultado de las siguientes operaciones:

1) 3 / 2 * 4
2) 4 + 3 / 2
3) (4 + 3) / 2
4) 17 / 5
5) floor(17 / 5)
6) mod(17, 5)
7) cos(pi)
8) cos(180)
9) sin(pi / 2)
10) log(exp(1) ^ 5)
11) 1 + 2 * 3
12) (1 + 2) * 3
13) 1 / 2 + 3
14) 1 / (2 + 3)

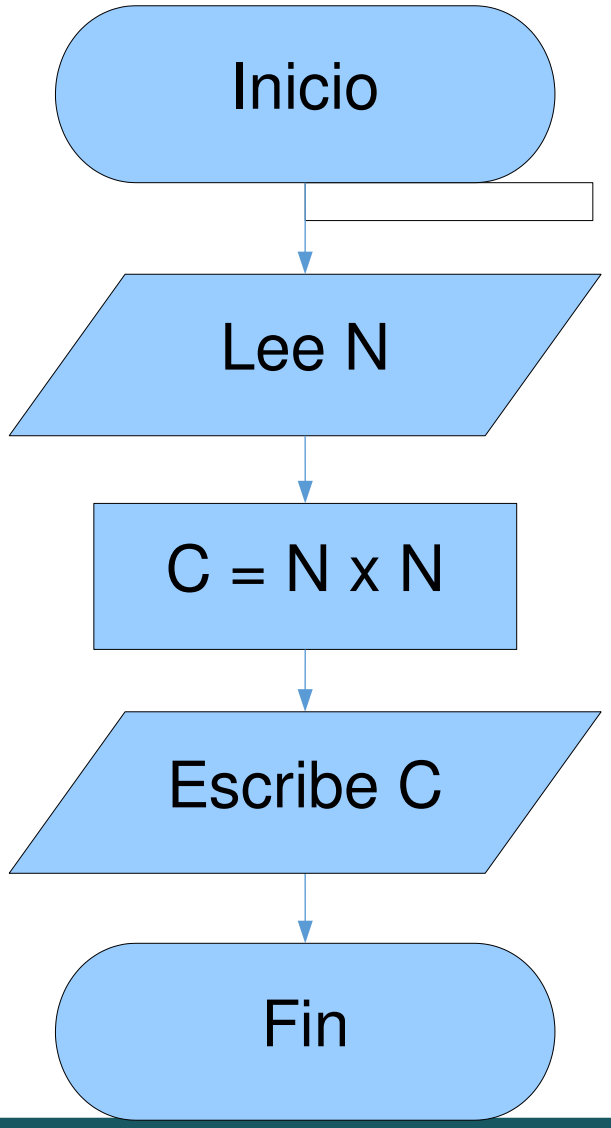
Ejemplos: Operadores Aritméticos

Dar el resultado de las siguientes operaciones:

1) 3 / 2 * 4 **= 6**
2) 4 + 3 / 2 **= 5.5000**
3) (4 + 3) / 2 **= 3.5000**
4) 17 / 5 **= 3.4000**
5) floor(17 / 5) **= 3**
6) mod(17, 5) **= 2**
7) cos(pi) **= -1**
8) cos(180) **= -0.59846**
9) sin(pi / 2) **= 1**
10) log(exp(1) ^ 5) **= 5**
11) 1 + 2 * 3 **= 7**
12) (1 + 2) * 3 **= 9**
13) 1 / 2 + 3 **= 3.5000**
14) 1 / (2 + 3) **= 0.2000**

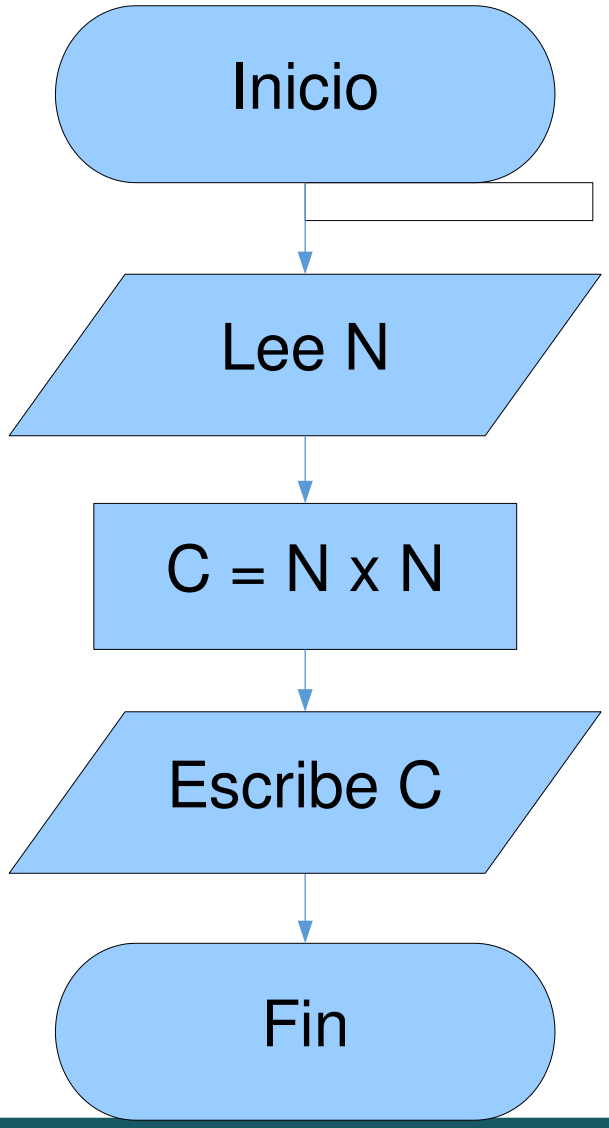
Calcula el Cuadrado

Hacer un programa que solicite un número y calcule su cuadrado:



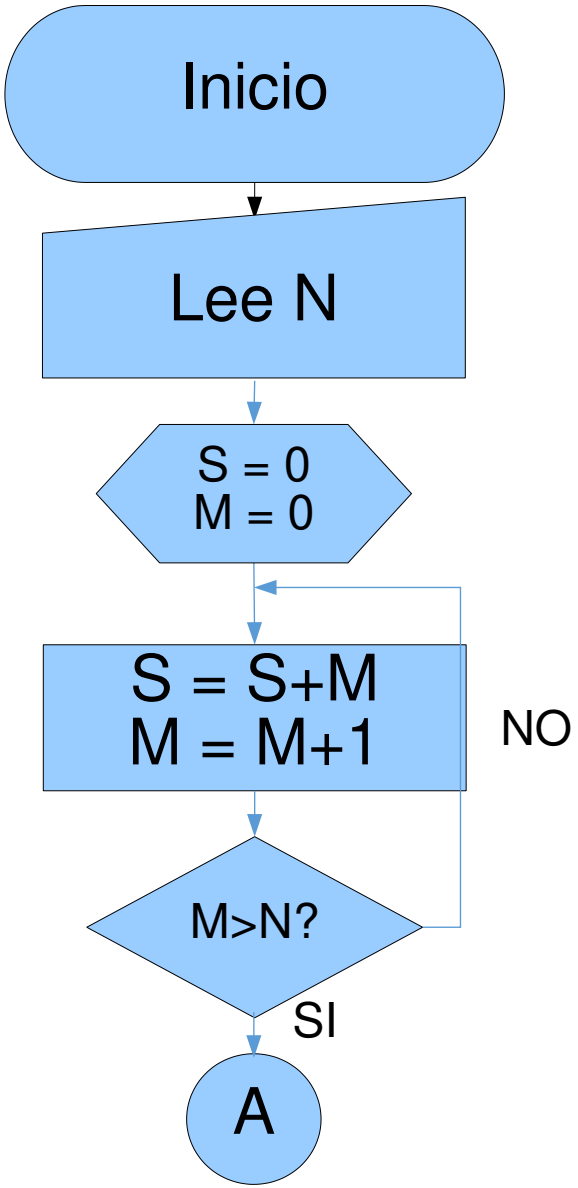
Calcula el Cuadrado

Hacer un programa que solicite un número y calcule su cuadrado:



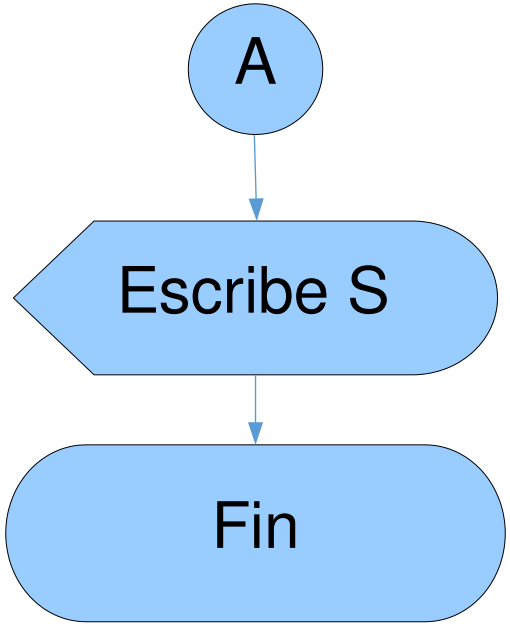

```
% Solicitamos el numero:
N = input('Dame un numero: ');
% Calculamos el cuadrado:
C = N*N;
% Mostramos el resultado por pantalla:
disp('El cuadrado del numero es: ');
disp(C);
```

Sentencias de Control



El conjunto de operaciones nos permitiría únicamente operar de forma secuencial, es decir, **todas las sentencias o instrucciones que introducimos en el código se ejecutan una por una y de arriba abajo**.

Sin embargo, hemos visto que hay otro tipo de estructuras, como los ciclos y las bifurcaciones.



Operadores Básicos: Relacionales

Operación	Símbolo	Ejemplo	Resultado
Menor que	<	5 < 3	
Mayor que	>	5 > 3	
Menor o igual que	<=	5 <= 3	
Mayor o igual que	>=	5 >= 3	
Distinto que	~=	5 ~= 3	
Igual que	==	5 == 3	

Nota: Estos operadores se aplican para la comparación de números, no de cadenas de caracteres o vectores, las cuales veremos más adelante ya que involucran funciones específicas (p.e. strcmp(), strmatch(), etc.).

Operadores Básicos: Relacionales

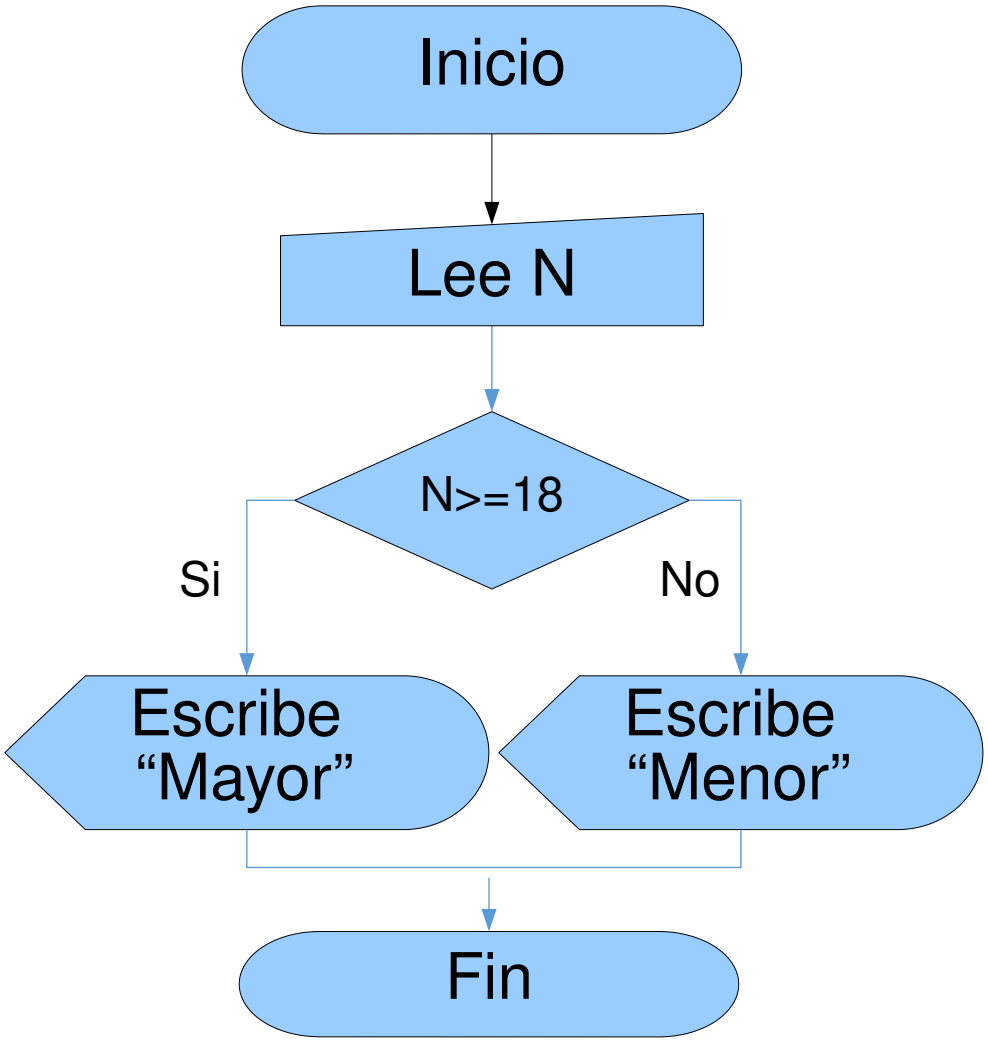
Operación	Símbolo	Ejemplo	Resultado
Menor que	<	5 < 3	*false*
Mayor que	>	5 > 3	*true*
Menor o igual que	<=	5 <= 3	*false*
Mayor o igual que	>=	5 >= 3	*true*
Distinto que	~=	5 ~= 3	*true*
Igual que	==	5 == 3	*false*

Nota: Estos operadores se aplican para la comparación de números, no de cadenas de caracteres o vectores, las cuales veremos más adelante ya que involucran funciones específicas (p.e. strcmp(), strmatch(), etc.).

Escribir en la terminal de Octave cualquiera de los ejemplos anteriores, **¿cómo representa Octave/Matlab el resultado false/true?**

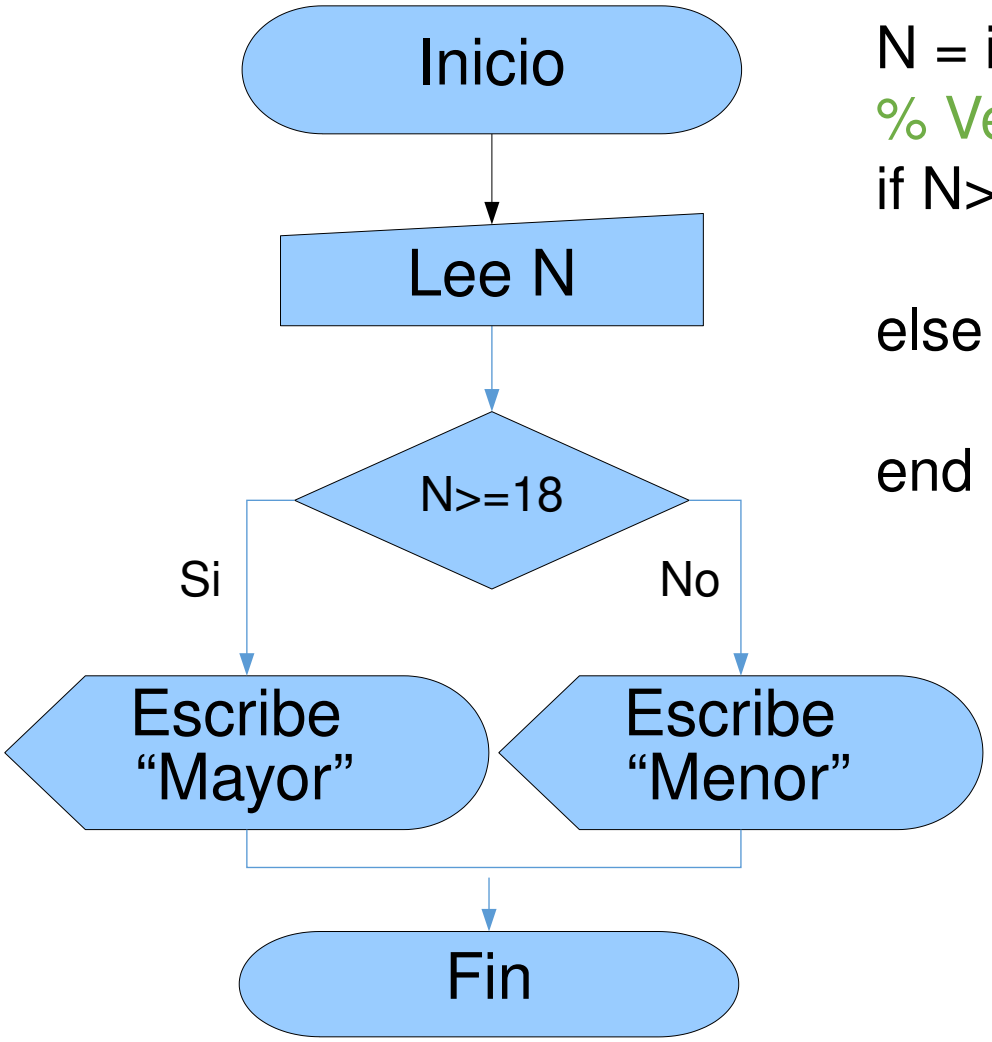
Mayor de Edad

Hacer un programa que solicite la edad y diga si es mayor o menor de edad:



Mayor de Edad

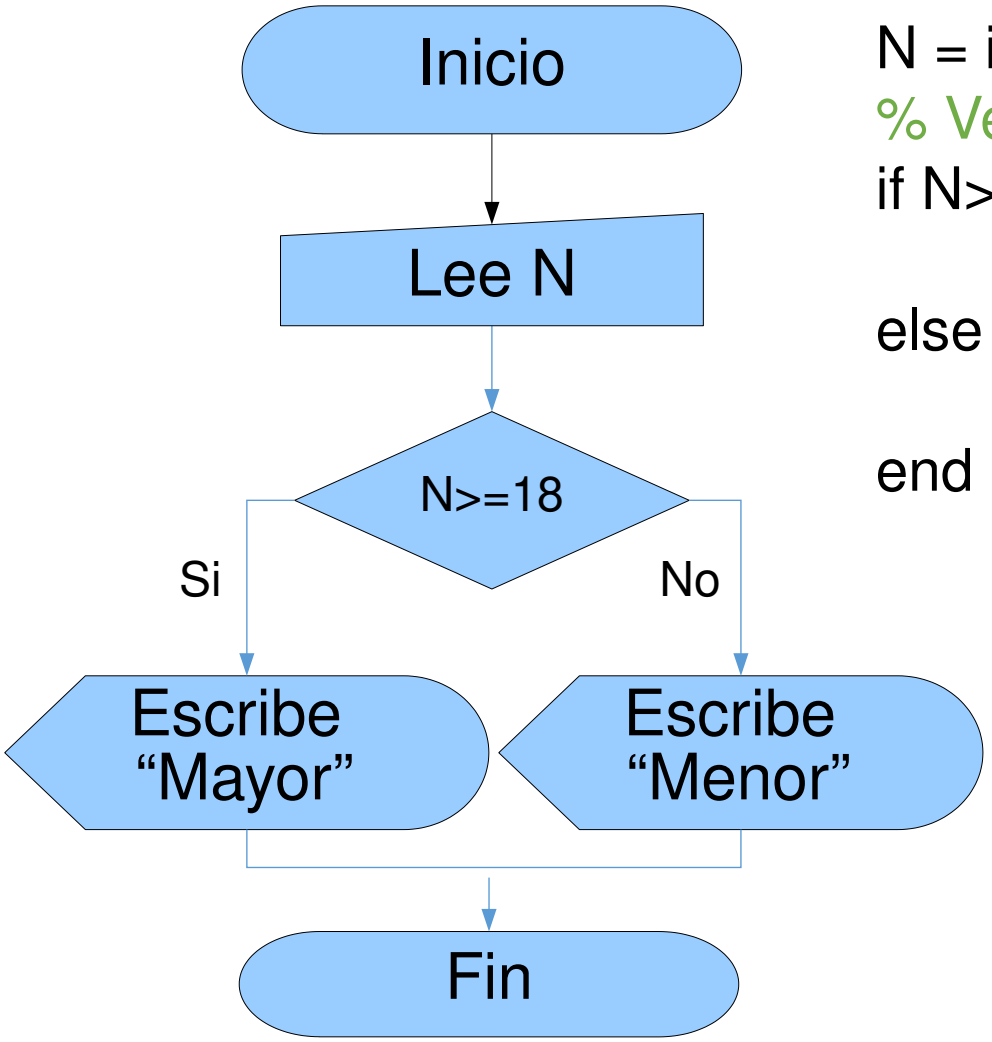
Hacer un programa que solicite la edad y diga si es mayor o menor de edad:




```
% Solicitamos el numero:
N = input('Dame la edad: ');
% Vemos si es mayor o menor:
if N>=18
        disp('Es mayor de edad');
else
        disp('Es menor de edad');
end
```

Sentencias de Control

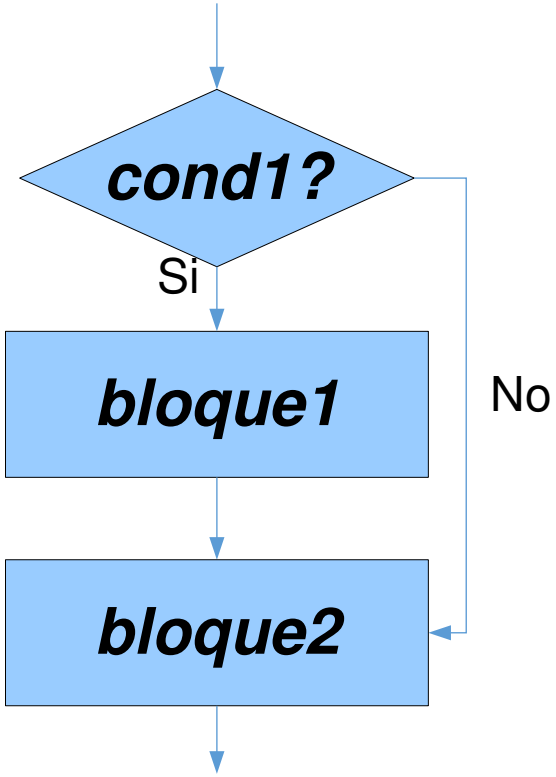
Hacer un programa que solicite la edad y diga si es mayor o menor de edad:




```
% Solicitamos el numero:
N = input('Dame la edad: ');
% Vemos si es mayor o menor:
if N>=18
        disp('Es mayor de edad');
else
        disp('Es menor de edad');
end
```

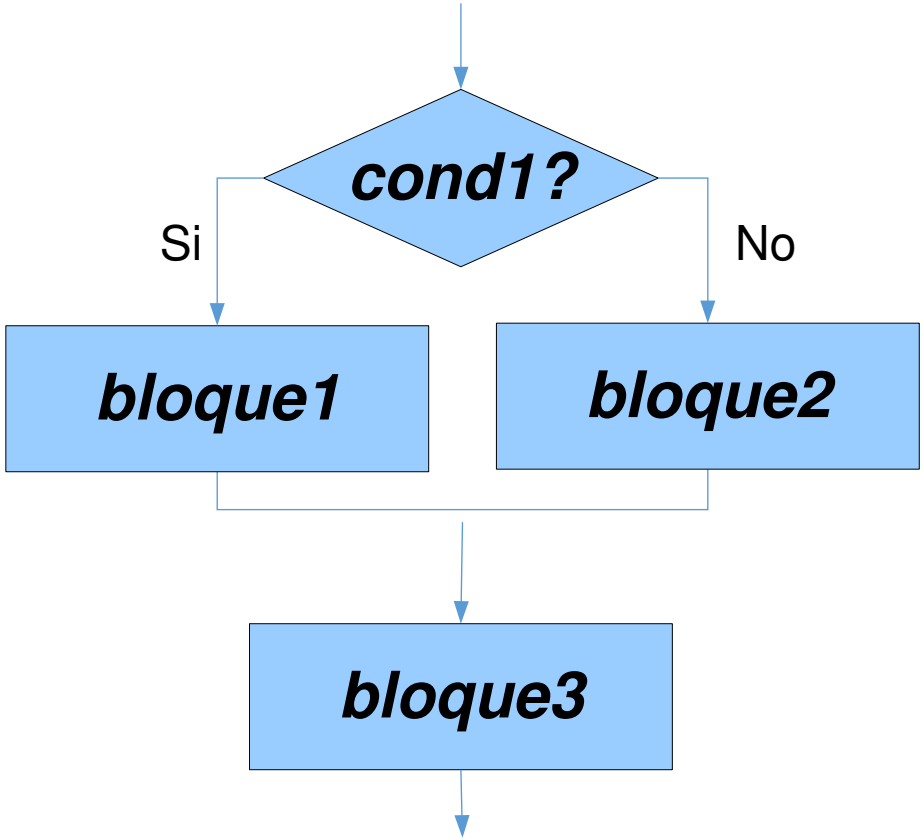
bifurcación/decisión/condición (no todas las sentencias se ejecutan dependiendo de condiciones que impongamos)

Sentencias de Control: Condicionales



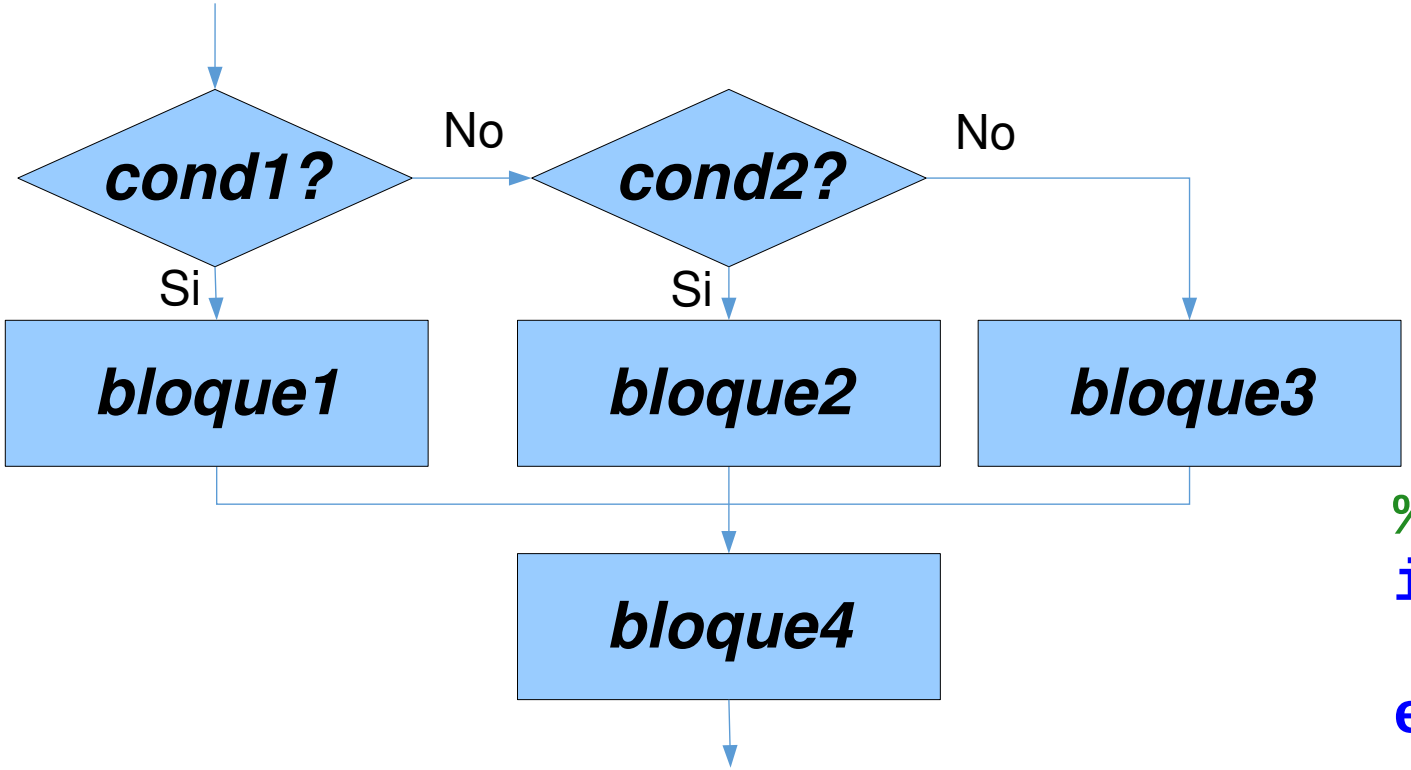

```
%OCTAVE/MATLAB
if cond1
   bloque1
end
bloque2
```

Sentencias de Control: Condicionales




```
%OCTAVE/MATLAB
if cond1
  bloque1
else
  bloque2
end
bloque3
```

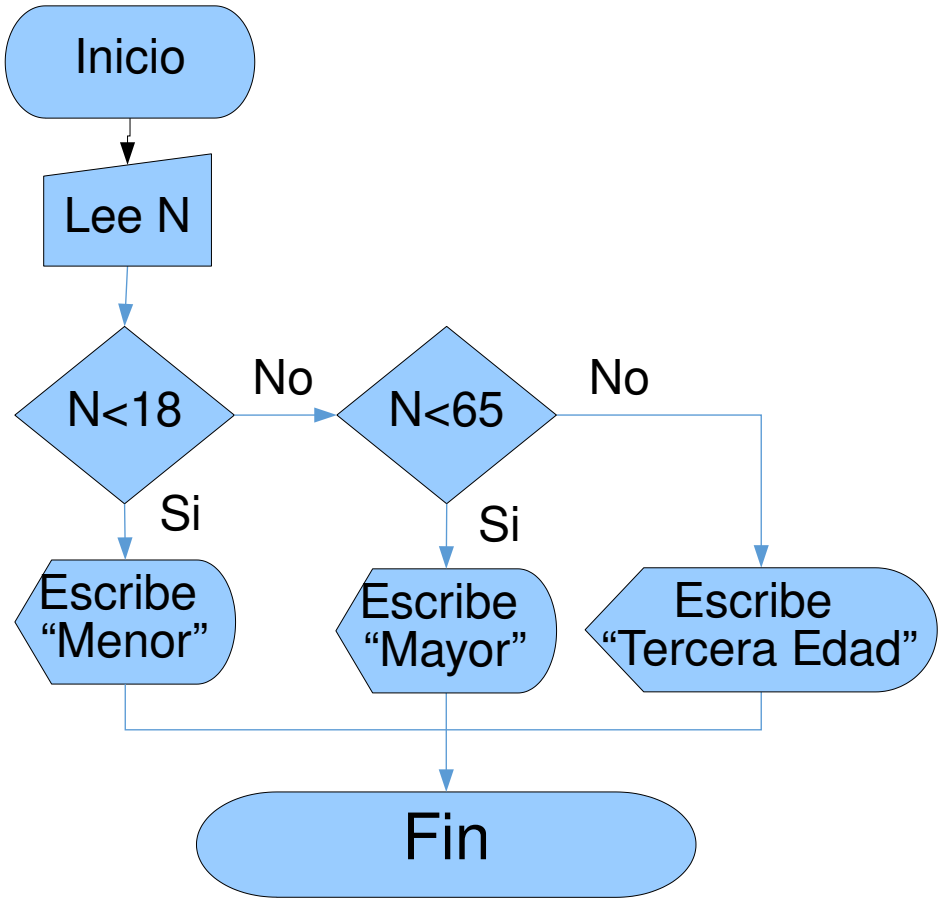
Sentencias de Control: Condicionales




```
%OCTAVE/MATLAB
if cond1
  bloque1
elseif cond2
  bloque2
else
  bloque3
end
bloque4
```

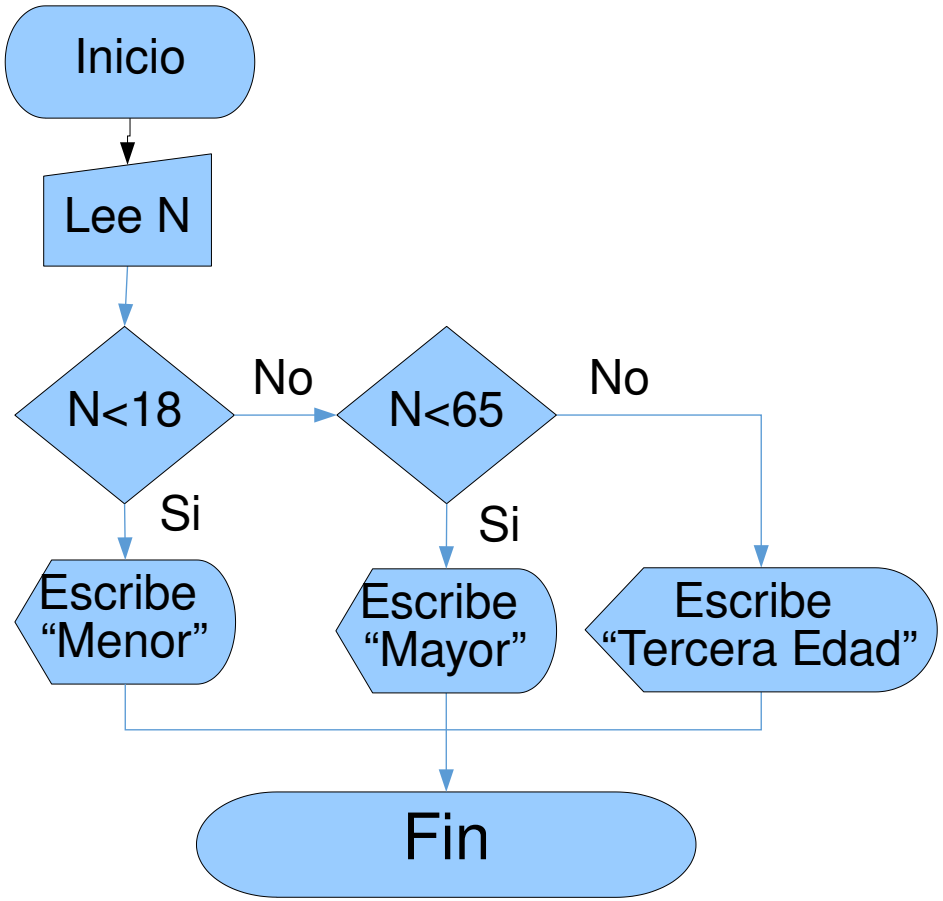
Menor, Mayor o Tercera Edad

Hacer un programa que solicite la edad y diga si es menor, mayor o de la tercera edad:



Menor, Mayor o Tercera Edad

Hacer un programa que solicite la edad y diga si es menor, mayor o de la tercera edad:




```
% Solicitamos el numero:
N = input('Dame la edad: ');
% Vemos si es mayor o menor:
if N<18
        disp('Es menor de edad');
elseif N<65
        disp('Es mayor de edad');
else
        disp('Es de la tercera edad');
end
```

Operadores Básicos: Lógicos

Operación	Símbolo	Ejemplo
Y lógico (AND)	&	a & b
O Lógico (OR)	|	a | b
NO lógico (NOT)	~	~a
XOR lógico (XOR)	xor()	xor(a,b)

Los operadores lógicos nos permiten concatenar condiciones. Por ejemplo, la definición de un intervalo sería 0<X<18, que es la conjunción de dos condiciones, 0<X y X<18, por lo que es necesario un operador lógico que permita realizar esta concatenación: (0<X) & (X<18).

Operadores Básicos: Lógicos

X	Y	Resultado (X & Y)
true (1)	true (1)	true (1)
true (1)	false (0)	false (0)
false (0)	true (1)	false (0)
false (0)	false (0)	false (0)

X	Y	Resultado (X | Y)
true (1)	true (1)	true (1)
true (1)	false (0)	true (1)
false (0)	true (1)	true (1)
false (0)	false (0)	false (0)

Operadores Básicos: Lógicos

X	Y	Resultado (xor(X,Y))
true (1)	true (1)	false (0)
true (1)	false (0)	true (1)
false (0)	true (1)	true (1)
false (0)	false (0)	false (0)

X	Resultado (~X)
true (1)	false (0)
false (0)	true (1)

Operadores Básicos: Lógicos

X	Y	~(X & Y) = (~X | ~Y)
true (1)	true (1)	false (0)
true (1)	false (0)	true (1)
false (0)	true (1)	true (1)
false (0)	false (0)	true (1)

X	Y	~(X | Y) = (~X & ~Y)
true (1)	true (1)	false (0)
true (1)	false (0)	false (0)
false (0)	true (1)	false (0)
false (0)	false (0)	true (1)

Precedencia de operadores

() paréntesis el más interno es el que primero se ejecuta.

^ exponenciación.

* / \ multiplicación y división (igual precedencia).

+ - Suma y resta (igual precedencia).

< <= > >= == ~= relacionales

& AND lógico

| OR lógico

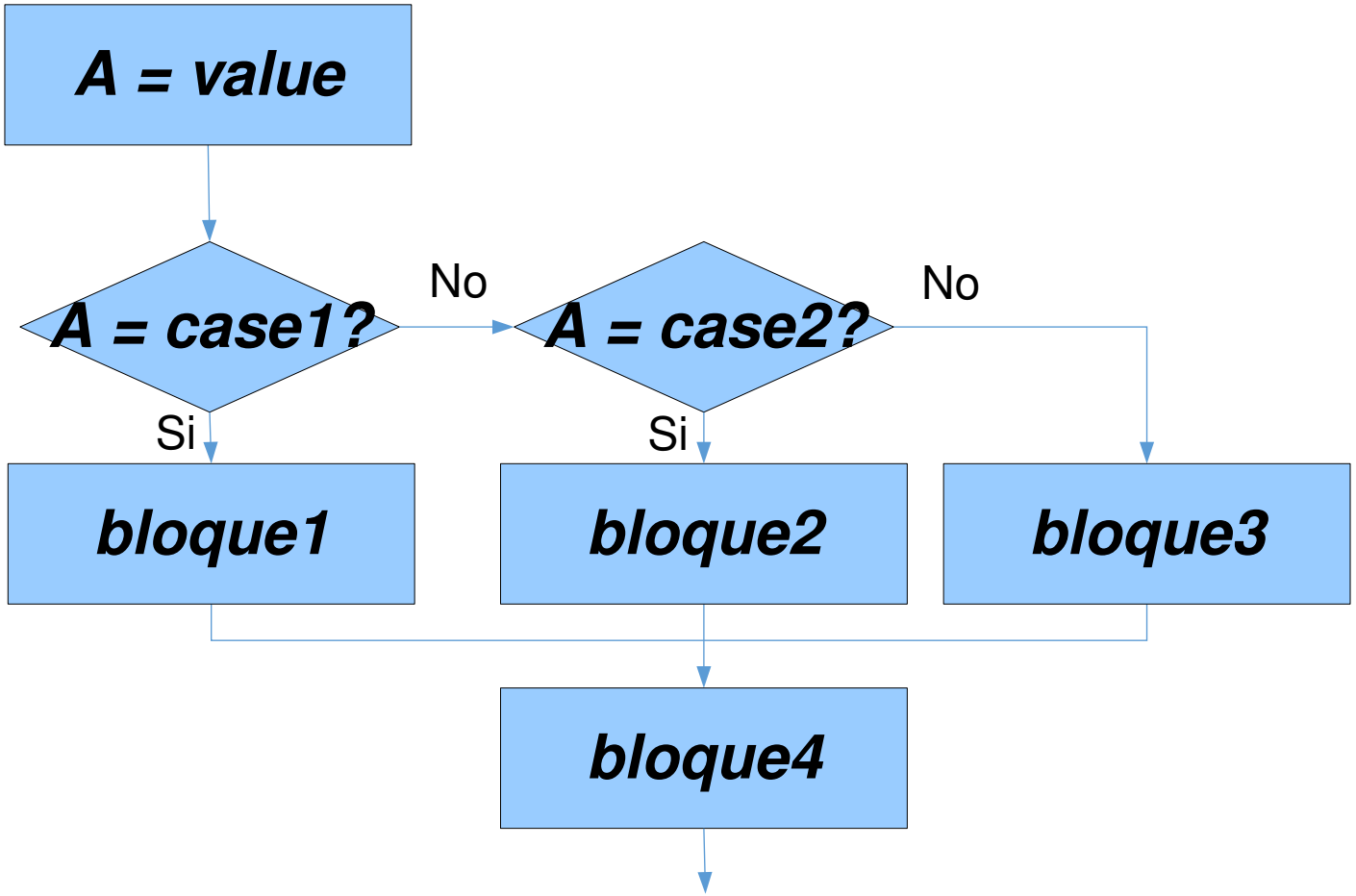
Si dos o más operaciones tiene la misma precedencia, la expresión entonces será ejecutada de izquierda a derecha.

Ejemplos: Operadores Aritméticos

Dar el resultado de las siguientes operaciones:

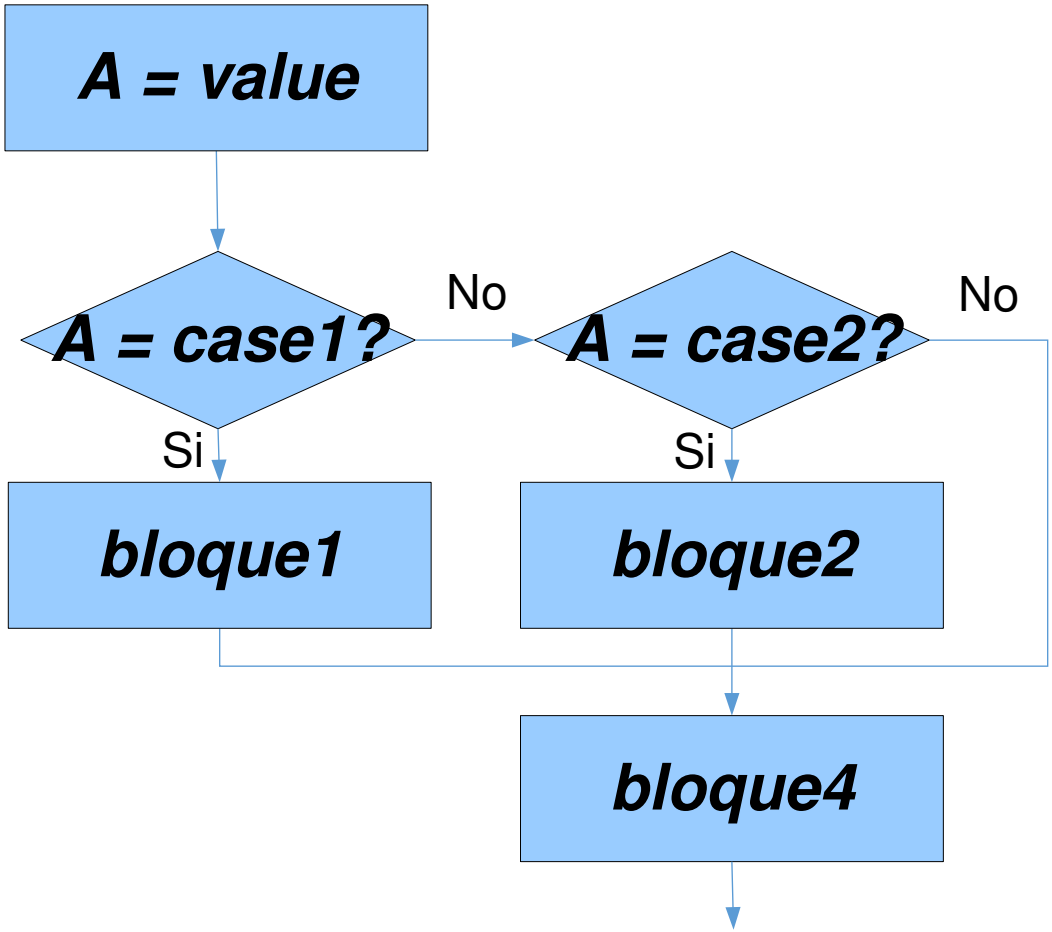
1) 3 / 2 * 4 **= 6**
2) 4 + 3 / 2 **= 5.5000**
3) (4 + 3) / 2 **= 3.5000**
4) 17 / 5 **= 3.4000**
5) floor(17 / 5) **= 3**
6) mod(17, 5) **= 2**
7) cos(pi) **= -1**
8) cos(180) **= -0.59846**
9) sin(pi / 2) **= 1**
10) log(exp(1) ^ 5) **= 5**
11) 1 + 2 * 3 **= 7**
12) (1 + 2) * 3 **= 9**
13) 1 / 2 + 3 **= 3.5000**
14) 1 / (2 + 3) **= 0.2000**

Sentencias de Control: Condicionales




```
%OCTAVE/MATLAB
switch A
   case case1
     bloque1
   case case2
     bloque2
   otherwise
     bloque3
end
bloque4
```

Sentencias de Control: Condicionales




```
%OCTAVE/MATLAB
switch A
   case case1
     bloque1
   case case2
     bloque2
end
bloque4
```

Ejercicios

Hacer un programa que solicite el radio de una circunferencia y muestre por pantalla el diámetro, el perímetro y el área.

Hacer un programa para convertir una velocidad en m/s a km/h.

Hacer un programa que muestre por pantalla la temperatura en grados Kelvin y en grados Fahrenheit a partir de la temperatura en grados Celsius introducida manualmente por el usuario.

Hacer un programa que te pida un número y te diga si es par o es non

Hacer un programa que lea 2 números y escriba cual es el menor.

Hacer un programa que pida 3 números y diga cual es el mayor.

Hacer un programa que pida la hora y muestre si es por la mañana (<12), por la tarde (<20) o de noche(>=20).

Hacer un programa que solicite 4 calificaciones y diga si está aprobado o no.

Ejercicios

Hacer un programa que recoja dos valores (i y j). Que escriba “POSITIVOS” si los dos son positivos, “NEGATIVOS” si los dos son negativos o “MEZCLADOS” si uno es positivo y el otro no.

Hacer un programa que solicite los coeficientes de una ecuación cuadrática, la clasifique en función del discriminante y devuelva las soluciones. Las soluciones complejas se darán en función de su parte real e imaginaria.

Determine si un año es bisiesto. Un año es bisiesto si es múltiplo de 4 pero no de 100 a excepción de los múltiplos de 400 que siempre son bisiestos.

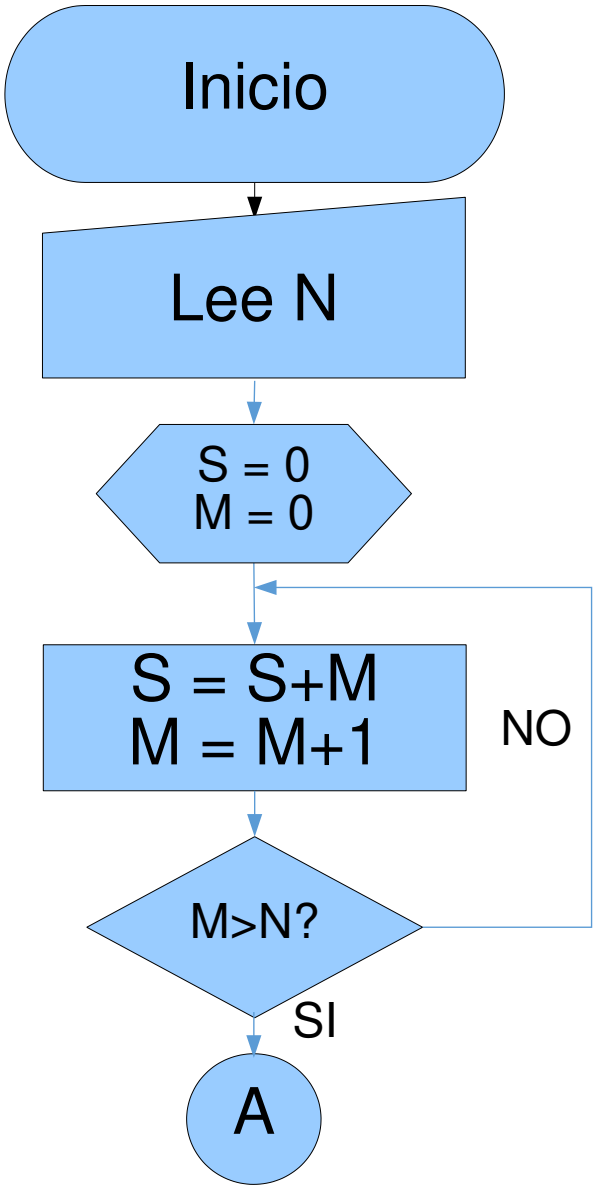
Lea un valor x y devuelva f(x):

$$f(x)=\begin{cases} \dfrac{1}{x-1} & si \quad x<1 \\ 0 & si \quad x=1 \\ \log(x-1) & si \quad x>1 \end{cases}$$

Fundamentos de Computación

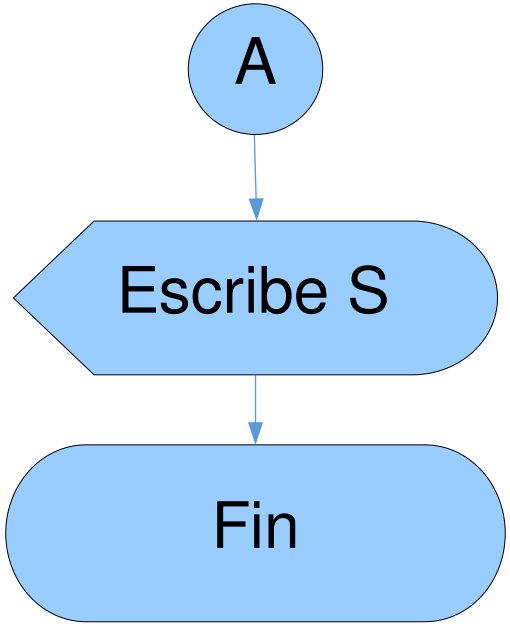
Primeros Scripts en Octave
Sentencias de Control: Ciclos

Sentencias de Control

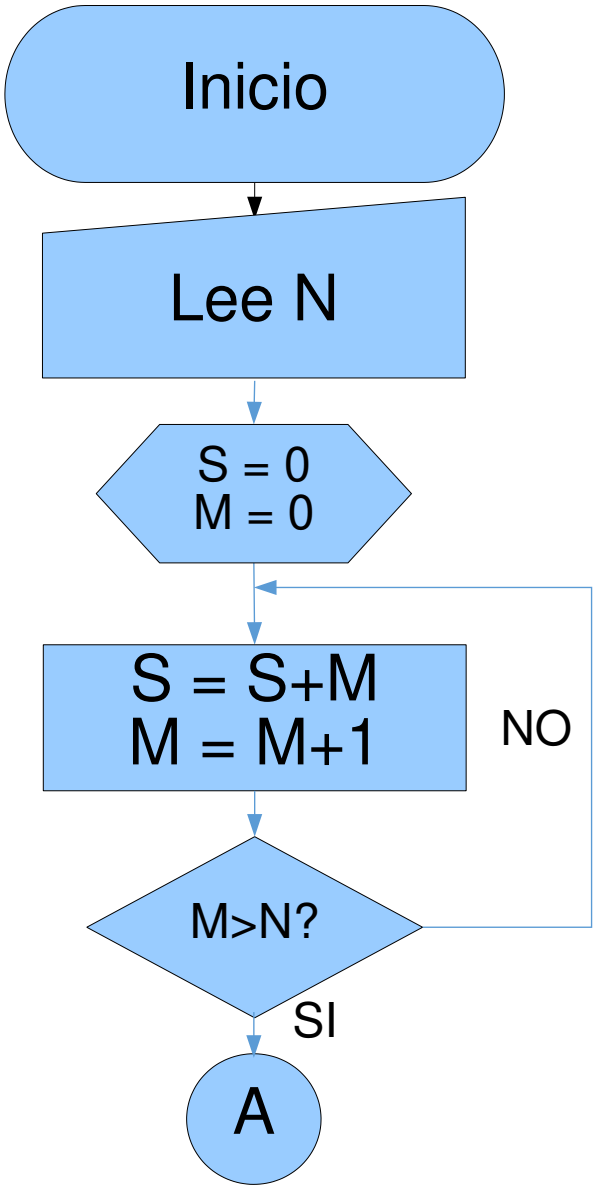


El conjunto de operaciones nos permitiría únicamente operar de forma secuencial, es decir, **todas las sentencias o instrucciones que introducimos en el código se ejecutan una por una y de arriba abajo**.

Sin embargo, hemos visto que hay otro tipo de estructuras, como los ciclos y las bifurcaciones.



Sentencias de Control

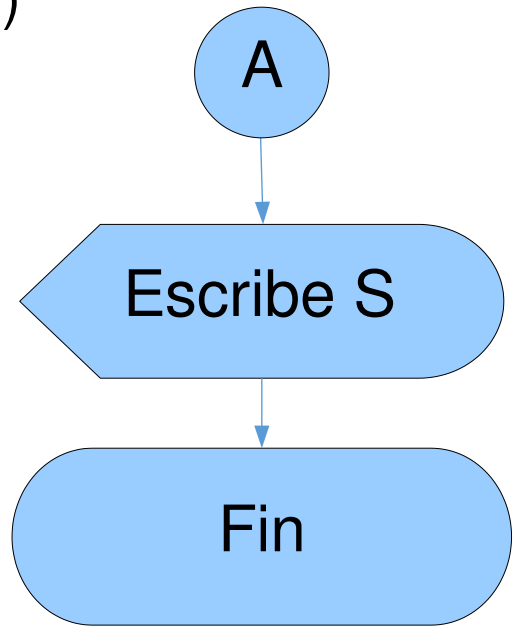


Iteración/Repetición/Ciclos:

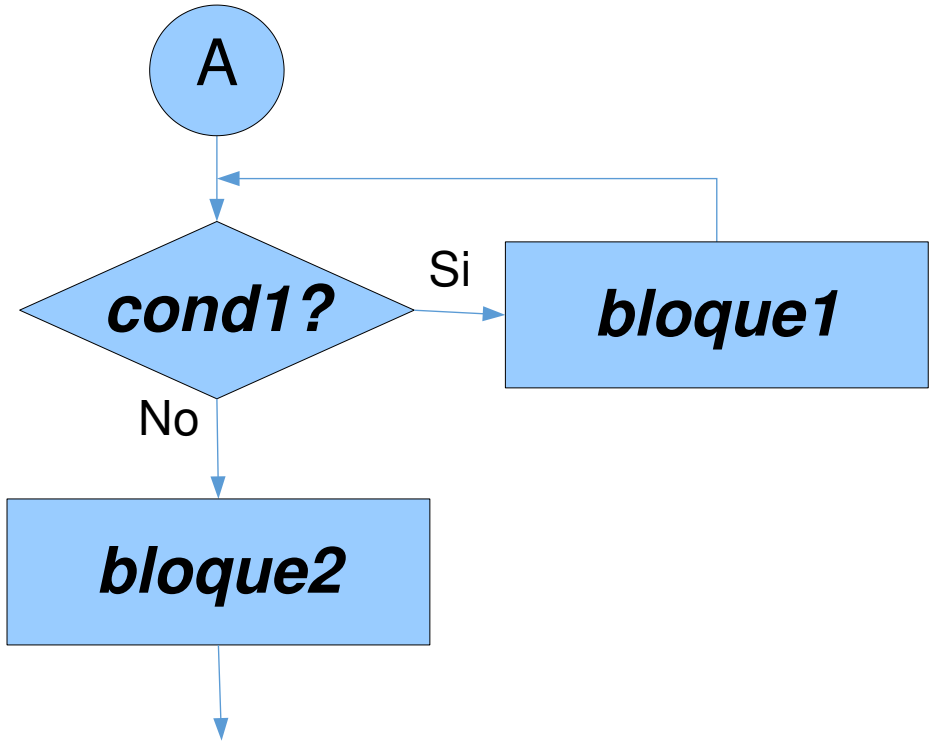
Los ciclos (bucles) permiten volver hacia atrás en el flujo de trabajo para repetir la ejecución de ciertas sentencias. Las órdenes básicas son:

while … **end** (repite ciertas órdenes **mientras** se cumpla una condición)

for … **end** (repite ciertas órdenes **para** unos valores dados)

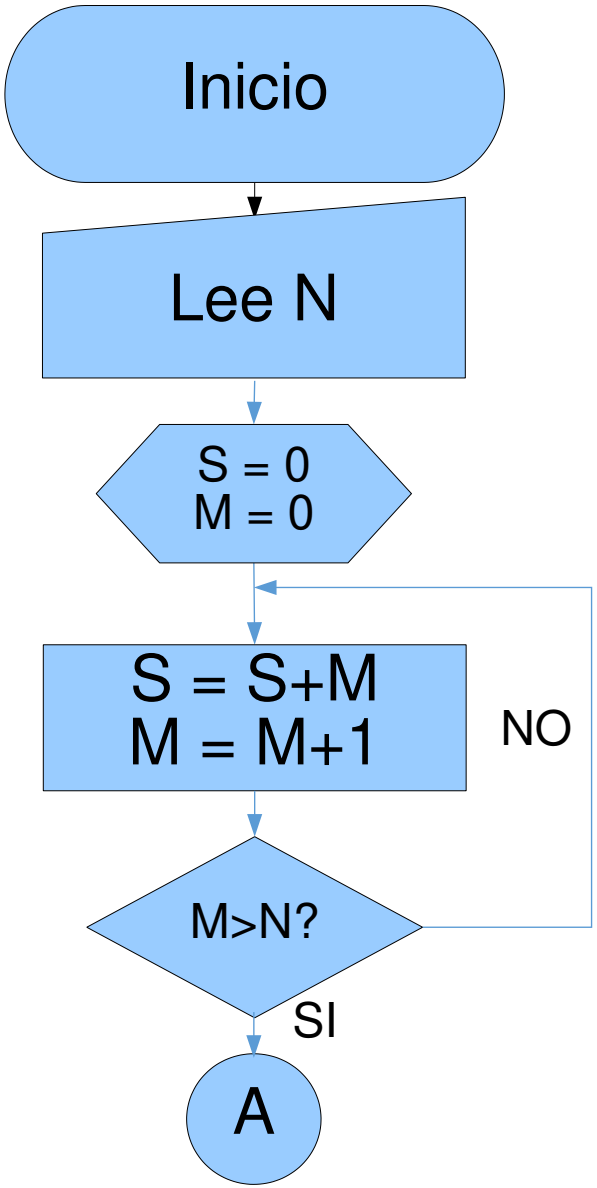


Sentencias de Control: Ciclos (Bucles)



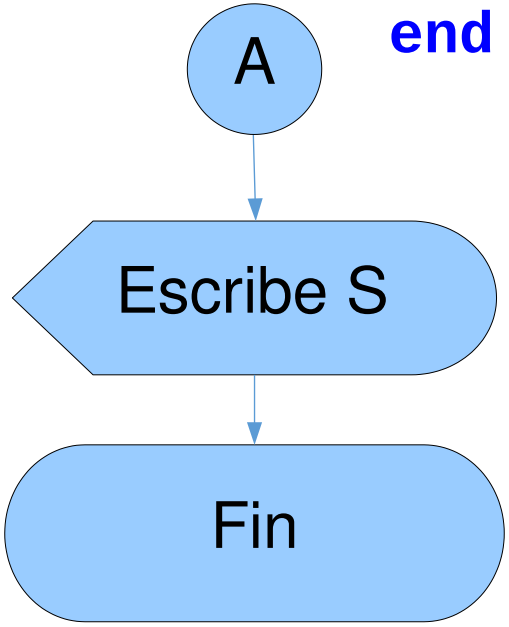

```
%OCTAVE/MATLAB
while cond1
   bloque1
end
bloque2
```

Sentencias de Control: Ciclos (Bucles)

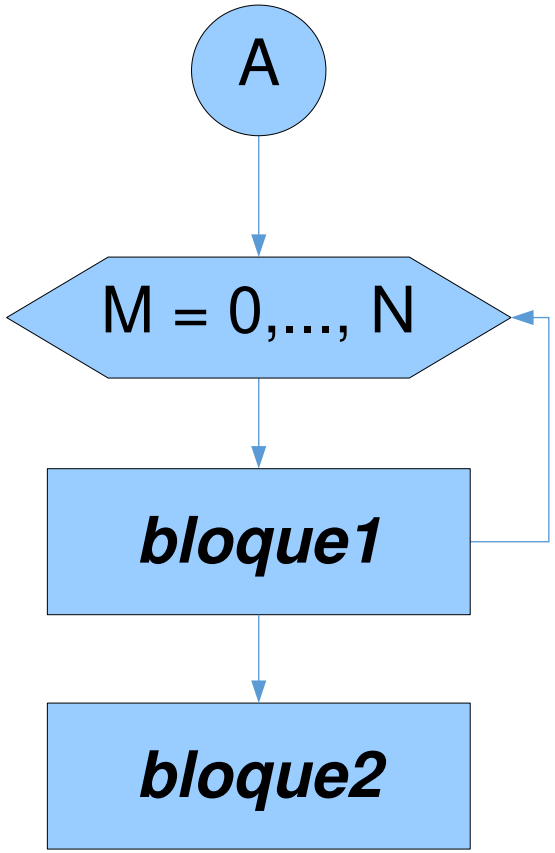


Hasta que el valor de **M** sea mayor que **N** el programa repetirá las operaciones actualizando **S** y **M**.

```
%OCTAVE/MATLAB
while M<=N
    S=S+M;
    M=M+1;
end
```

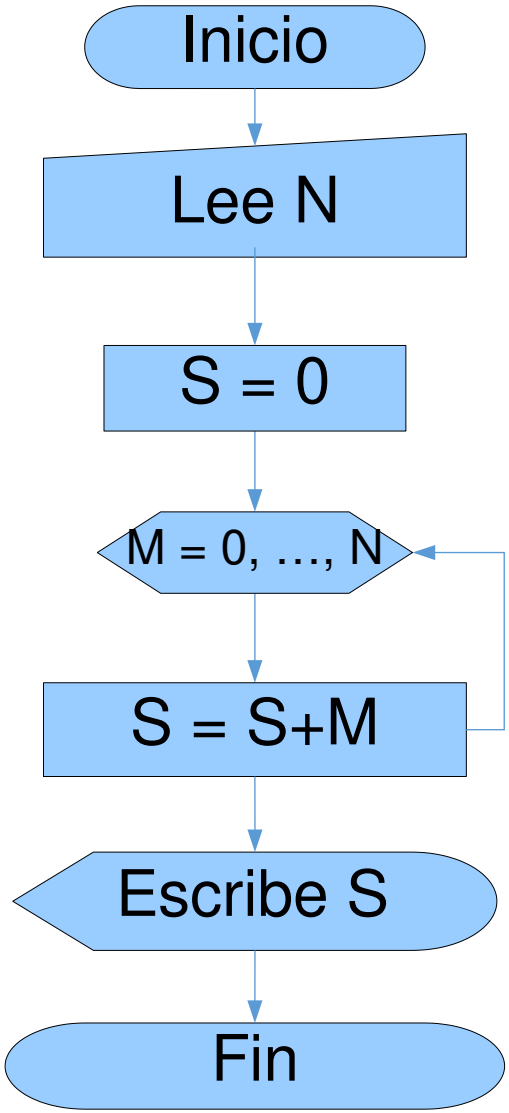


Sentencias de Control: Ciclos (Bucles)




```
%OCTAVE/MATLAB
for M=0:N
   bloque1
end
bloque2
```

Sentencias de Control: Ciclos (Bucles)



Para los valores de **M**: 0, 1, 2, 3,..., **N**-1, **N** el programa sumará a **S** el valor de **M**.

```
%OCTAVE/MATLAB
for M=0:N
    S=S+M;
end
```

Sentencias de Control: Anidamiento

Cualquier sentencia de control puede combinarse con otra de su mismo tipo

```
%OCTAVE/MATLAB
X=input('Escribe un numero natural positivo: ');
if X<0
   if round(X)~=X
      disp('El numero debe ser natural y positivo');
   else
      disp('El numero debe ser positivo');
   end
else
   if round(X)~=X
      disp('El numero debe ser natural');
   else
      disp(['El numero dado es: ' num2str(X)]);
   end
end
```

Sentencias de Control: Anidamiento

o de cualquier otro, dando lugar a estructuras anidadas:

```
%OCTAVE/MATLAB
X=input('Escribe un numero natural positivo: ');
X=abs(X);
if round(X)~=X
   disp('El numero debe ser natural');
else
   S=0;
   for i=1:X
      S=S+i;
      while S<=X
         disp(['La suma es: ' num2str(S)]);
      end
   end
end
```

Ejercicios: Operadores Lógicos

Utilizando operadores lógicos transformar o hacer:

Un programa que pida 3 números y diga cual es el mayor

Un programa que pida la hora y muestre si es por la mañana (6<hora<=12), por la tarde (12<hora<=20) o de noche(20<hora<=6).

Un programa que recoja dos valores (i y j). Que escriba “POSITIVOS” si los dos son positivos, “NEGATIVOS” si los dos son negativos o “MEZCLADOS” si uno es positivo y el otro no.

Un programa que solicite los coeficientes de una ecuación cuadrática, la clasifique en función del discriminante y devuelva las soluciones. Las soluciones complejas se darán en función de su parte real e imaginaria.

Un programa que determine si un año es bisiesto. Un año es bisiesto si es múltiplo de 4 pero no de 100 a excepción de los múltiplos de 400 que siempre son bisiestos.

Ejercicios: Ciclos

Utilizando ciclos (while y for) hacer un programa:

Que solicite 4 calificaciones y diga si está aprobado o no.

Para calcular el factorial de un número.

Para imprimir por pantalla los primeros N términos de la sucesión de Fibonacci, donde el número será introducido manualmente por el usuario.

Que pida un número N y despliegue todas las combinaciones de dos números que sumados den N.

Que despliegue la tabla de multiplicar (0*X,...,10*X) de un número X.

Que recoja números del usuario hasta que se introduzca un 0 y devuelva la media de los números introducidos.

Que pida un número y vaya decrementando su valor en una unidad hasta que llegue a cero.

Qida un numero y compruebe si ese numero es primo.

Que pida un número hasta que este número sea mayor que cero, indicando con un mensaje que el número introducido no es mayor de cero.

Ejercicios: Ciclos

Que calcule la raíz en el intervalo [0 2] del polinomio x^3-3x^2-2x+6 con una precisión de 1e-5 a través del método de la bisección.

Que calcule la raíz del polinomio anterior con el método de Newton-Raphson con la misma tolerancia tomando como condición inicial los extremos del intervalo [0 2]. ¿Qué soluciones obtienes? ¿Coinciden con las anteriores?

Fundamentos de Computación

Tipos de datos
Cálculo matricial con Octave
Operando con strings

Cálculo matricial

Como hemos visto, Octave considera las variables numéricas (y strings) como matrices de las dimensiones correspondientes (p.e: escalar = Matriz 1x1).

```
% Definiendo matrices:
A=[1 2 3];% Vector fila.
size(A)% Dimensiones de la matriz (1, 3).
B=[1; 2; 3];% Vector columna.
size(B)% Dimensiones de la matriz (3, 1).
length(A)% Longitud del vector: 3.
length(B)% Longitud del vector: 3.
C=[1 2 3;4 5 6;7 8 9;10 11 12];% Matriz 4x3.
size(C)
% Matrices especiales:
A=zeros(3,5);% Matriz de ceros.
B=ones(3,5);% Matriz de unos.
C=eye(3);% Matriz identidad de dimensión 3.
D=diag([3 5 6]);%Matriz diagonal.
```

Cálculo matricial

Una vez definida la matriz, podemos seleccionar elementos o submatrices:

```
% Definiendo matrices:
A=rand(10,10);% Matriz aleatoria 10x10.
A(1,:) % Primera fila.
A(:,1) % Primera columna.
A(3,5)% Elemento de la fila 3 y columna 5.
A(1:3,3:5)% Submatriz de filas 1,2,3 y columnas 3,4,5.
A(3,5) = 7;% Redefino el elemento (3,5).
A(1:3,3:5) = rand(3,3);% Redefino la submatriz.
```

El comando ***end*** permite referirnos a los elementos desde el final:

```
A(end,:) % Última fila.
A(:,end) % Última columna.
A(end,end)% Elemento de la última fila y columna.
A(3:end,5:end-1)% Submatriz.
```

En ***strings*** se pueden seleccionar partes del texto del mismo modo:

```
Direccion='C/Benito Vercimuelles';
Direccion(10:end)% Muestro sólo Vercimuelles.
```

Cálculo matricial

```
% Operaciones matriciales:
A=rand(3,4);% Matriz aleatoria 3x4.
B=rand(3,4);% Matriz aleatoria 3x4.
C=A+B;% Suma de matrices.
D=A*B;% ERROR: Las dimensiones no concuerdan.
D=A*B';% A multiplicado por la traspuesta de B (B').
A=rand(3,3);% Matriz aleatoria 3x3.
A^2 % Eleva la Matriz al cuadrado (=A*A).
C=A\B % Division izquierda.
C=A/B' % Division derecha.
A=rand(3,4);% Matriz aleatoria 3x4.
B=rand(3,4);% Matriz aleatoria 3x4.
C=A.*B;% Opera elemento a elemento.
D=A./B;% Opera elemento a elemento.
F=A.\B;% Opera elemento a elemento.
G=A.^2;% Opera elemento a elemento.
H=A.^B;% Opera elemento a elemento.
```

Cálculo matricial: Operador :

```
% El operador ':' genera secuencias de valores:
X=1:10;% [1 2 3 4 5 6 7 8 9 10].
X=1:2:10;% [1 3 5 7 9].
X=1:1.5:10;% ???.
X=10:-1:1;% ???.
% Dichas secuencias pueden utilizarse para seleccionar
elementos de una matriz o vector:
X=1:10;% [1 2 3 4 5 6 7 8 9 10].
X(1:2:10)% Devuelve los elementos [1 3 5 7 9].
X(10:-1:1) % Invertirá los elementos del vector.
A=rand(10,10);%
A(1:2:10,:)% Devuelve las filas [1 3 5 7 9].
A(10:-1:1,:) % Invertirá las filas (flipud(A)).
```

Funciones matriciales:

En los lenguajes interpretados los bucles son muy ineficientes, de modo que, cuando es posible, deben usarse funciones matriciales:

```
% Sumando filas/columnas de matrices:
A=rand(100,10);% Matriz aleatoria 100x10.
sumaFilas=sum(A,1) % Suma de las filas.
sumaCols=sum(A,2) % Suma de las columnas.
mediaFilas=mean(A,1) % Media de las filas.
mediaCols=mean(A,2) % Media de las columnas.
maxFilas=max(A,[],1) % Máximo de las filas.
minCols=min(A,[],2) % Mínimo de las columnas.
```

- Revisa otras funciones (***std***, ***var***, ***corr***, etc...) con el comando ***help***.
- Implementa las operaciones anteriores y compara los tiempos con los comando ***tic*** y ***toc***.

Funciones matriciales:

Una parte importante es la localización de elementos dentro de una matriz, sea numérica o de caracteres:

- Define dos matrices ***A*** y ***B*** de las mismas dimensiones y compáralas, ¿qué obtienes?

```
A=rand(3,3);% Matriz aleatoria 3x3.
B=randn(3,3);% Matriz aleatoria 3x3.
A<B % ¿Qué se obtiene?.
find(A<B) % ¿Qué se obtiene?.
% Buscando elementos dentro de una matriz numérica:
% Funciones find y sub2ind/ind2sub:
A=rand(100,10);% Matriz aleatoria 100x10.
indgt05=find(A>0.5) % Indices en la matriz
[rgt05,cgt05]=ind2sub(indgt05) % [Filas, Cols]
```

Cálculo matricial: Ejercicios

Considerar las funciones ***det*** (determinante), ***rank*** (rango), ***poly*** (polinomio característico) y ***roots*** (raíces de un polinomio) para hayar la dimensión del subespacio imagen, el núcleo y calcular la expresión diagonal de la aplicación lineal dada por la matriz:

A = [2 1 4;3 0 -1;4 -1 5];

Implementa con bucles el producto matricial y evalúa la diferencia entre el tiempo de cálculo con bucles y con el operador * de Octave (funciones ***tic*** y ***toc***).

Resuelve la ecuación matricial:

A=[1 1 0;0 -1 0;2 2 1];

B=[1 2 3;0 0 1;1 1 0];

E=[1 1 0;0 0 1;1 1 0];

A*X*B^(-1) = 2*E;

Dada la matriz A=[1 2 3 4;2 3 4 1;3 4 1 2;4 1 2 3] y el escalar r=2, comprueba las propiedades del determinante.

Operando con Strings

Cómo hemos visto, hay operaciones que se expresan de forma natural con variables numéricas pero cuyo funcionamiento difiere al considerar cadenas de caracteres.

```
%OCTAVE/MATLAB
numero=input('Escribe un numero: ');
texto=input('Escribe un texto: ');
if numero == 0
   disp('El operador == se usa para comparar numeros\n');
end
'pepito' == 'juanito' % Problema con las dimensiones
'pepito' == 'janito'  % [0 0 0 1 1 1]
disp('Para cadenas de caracteres debe usarse strcmp\n');
strcmp('pepito','juanito')
if strcmp(texto,'0')
   disp('La funcion strcmp permite comparar strings\n');
end
```

Otras funciones para manejar strings son: strmatch, findstr, strcat, strvcat, etc

Cadenas de Control

Cualquier sentencia de control puede combinarse con otra de su mismo tipo

```
%OCTAVE/MATLAB
X=input('Escribe un numero natural positivo: ');
if X<0
   if round(X)~=X
      disp('El numero debe ser natural y positivo');
   else
      disp('El numero debe ser positivo');
   end
else
   if round(X)~=X
      disp('El numero debe ser natural');
   else
      disp(['El numero dado es: ' num2str(X)]);
   end
end
```

Cadenas de Control

En el caso anterior es sencillo pero, ¿qué ocurre cuando tengo que mezclar muchas variables de tipos diferentes?:

```
%OCTAVE/MATLAB
Edad=24;
Nombre='Pedro Piqueras';
DNI='77.777.777-Z';
Direccion='Benito Vercimuelles';
Numero=45;
Piso=3;
Letra='C';
```

¿Cómo construyo el mensaje: "***Pedro Piqueras, de 24 años de edad, DNI: 77.777.777-Z y con domicilio en la Calle Benito Vercimuelles Nº 45, 3ºC, nació en el año 1992***"?

Cadenas de Control

```
Edad=24;
Nombre='Pedro Piqueras';
DNI='77.777.777-Z';
Direccion='C/Benito Vercimuelles';
Numero=45;
Piso=3;
Letra='C';
```

¿Cómo construyo el mensaje: "***Pedro Piqueras, de 24 años de edad, DNI: 77.777.777-Z y con domicilio en la Calle Benito Vercimuelles Nº 45, 3ºC, nació en el año 1992***"?

```
% Con la función disp:
disp([Nombre ', de ' num2str(Edad) ' años de edad, DNI: '
DNI ' y con domicilio en la Calle ' Direccion ' Nº '
num2str(Numero) ', ' num2str(Piso) 'º' Letra ', nació en
el año ' num2str(2016-Edad)]);
```

¿Hay algún modo más fácil de componer mensajes con variables de tipos diferentes?

Cadenas de Control

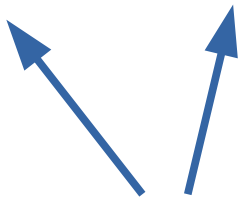
```
Edad=24;
Nombre='Pedro Piqueras';
DNI='77.777.777-Z';
Direccion='C/Benito Vercimuelles';
Numero=45;
Piso=3;
Letra='C';
```

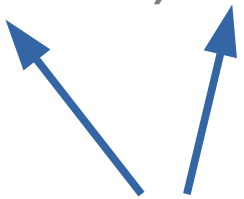
¿Cómo construyo el mensaje: "***Pedro Piqueras, de 24 años de edad, DNI: 77.777.777-Z y con domicilio en la Calle Benito Vercimuelles Nº 45, 3ºC, nació en el año 1992***"?

```
% Con la función sprintf y cadenas de control:
Text = sprintf('%s, de %d años de edad, DNI: %s y con
domicilio en la Calle %s Nº %d, %dº%s, nació en el año
%d', Nombre, Edad, DNI, Direccion, Numero, Piso, Letra,
2016-Edad);
disp(Text)
```

Cadenas de Control

```
Edad=24;
Nombre='Pedro Piqueras';
Text = sprintf('%s, de %d años de edad', Nombre, Edad);
disp(Text)
```





Dos cadenas de control

Dos variables

La función fprintf, que veremos con más detalle al trabajar con ficheros, puede utilizarse para mostrar mensajes por pantalla. Su funcionamiento es "equivalente" a las dos líneas finales del código anterior:

```
Edad=24;
Nombre='Pedro Piqueras';
Text = fprintf('%s, de %d años de edad', Nombre, Edad);
```

¿Qué contiene la variable ***Text***?

Cadenas de Control

Las ***cadenas de control*** son cadenas de caracteres especiales que usan funciones como ***sprintf*** (***sscanf***) o ***fprintf*** (***fscanf***) para dar formato a un mensaje o identificar el tipo de dato que se quiere leer de un mensaje. Llevan uno o varios caracteres de conversión precedidos por el símbolo ***%***.

Estos son los principales:

c → char (carácter)

s → string (cadena de texto)

d → int (entero decimal)

f → float (coma flotante, número real)

Aunque existen otros para casos particulares:

o → int (entero octal)

x → int (entero hexadecimal)

i → int (entero decimal, octal o hexadecimal)

u → int (entero unsigned, sin signo)

Cadenas de Control: Modificadores

Permiten controlar de el aspecto de escritura (también se pueden usar para lectura -scanf-)

Se introducen entre el % y el carácter de conversión:

```
text=sprintf('Son %d EUR\n', 345)          -> Son 345 EUR
text=sprintf('Son %5d EUR\n', 345)         -> Son   345 EUR
text=sprintf('Son %+5d EUR\n', 345)        -> Son  +345 EUR
text=sprintf('Son %-5d EUR\n', 345)        -> Son 345   EUR
text=sprintf('Son %05d EUR\n', 345)        -> Son 00345 EUR

text=sprintf('Son %f EUR\n', 345.2364)     -> Son 345.236400 EUR
text=sprintf('Son %7.2f EUR\n', 345.2364) -> Son  345.24 EUR
text=sprintf('Son %.2f EUR\n', 345.2364)   -> Son 345.24 EUR
text=sprintf('Son %10.3f EUR\n',345.2364) -> Son   345.236 EUR
```

Cadenas de Control: Modificadores

Permiten controlar de el aspecto de escritura (también se pueden usar para lectura -scanf-)

Se introducen entre el % y el carácter de conversión:

```
nom = 'Perico';
Nota = 6.73;
tx=sprintf('%%|%s %.2f|\n', nom, nota)       %|Perico 6.73|
tx=sprintf('%%|%10s %.2f|\n', nom, nota)     %|    Perico 6.73|
tx=sprintf('%%|%-10s %.2f|\n', nom, nota)    %|Perico     6.73|
tx=sprintf('%%|%10.3s %.2f|\n', nom, nota)   %|       Per 6.73|
tx=sprintf('%%|%.3s %.2f|\n', nom, nota)     %|Per 6.73|
```

Realizar un programa que pida 2 números y calcule el cuadrado de todos los número enteros entre en el intervalo cerrado definido por ambos números.
Por cada numero aparecerá una linea con el mensaje:
`El cuadrado del numero 2 es 4.`

Operando con Strings

¿Cómo se extiende la búsqueda a cadenas de caracteres?

```
%% OCTAVE/MATLAB
texto='El operador == se usa para comparar numeros';
findstr(texto,'ra')
%% Matriz de nombres:
nombres=strvcat('Jose','Juan','Sixto','Juan')
strmatch('Juan', nombres)
find(strmatch('Juan', nombres))
```

Cadenas de Control: Ejercicios

Modificar el programa anterior para que pida también la potencia (un número real) y repita la operación pero utilizando esa potencia en lugar del cuadrado.

Modificar el programa anterior para que muestre un menú con tres opciones: 1. Introducir intervalo, 2. Introducir exponente y 3. Mostrar los resultados.

El valor por defecto del exponente será 2 y el del intervalo [0 10].

Modificar el programa anterior para que repita la operación hasta que el usuario introduzca un 0.

Modificar el programa anterior para que sólo muestre: la opción 2 cuando se haya definido el intervalo y la opción 3 cuando se hayan definido el intervalo y el exponente.