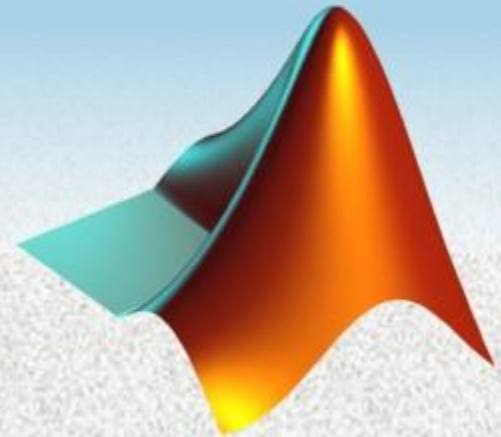




**UNIVERSIDAD  
NACIONAL DE  
INGENIERÍA**

# MATLAB

**R2016a**



Maria Pimentel Herrera  
[uni.kernel@gmail.com](mailto:uni.kernel@gmail.com)

# Información sobre las dimensiones

- Las órdenes **length** y **size** ofrecen información sobre el tamaño de vectores y matrices. Conviene recordar que en la versión 5.0 de MATLAB el tamaño máximo de una matriz es de 16384 (= 128 x 128) elementos.

- `size(A)`

*Genera un vector con las dimensiones de la matriz  $A$ .*

- `size(A, 1)`

*Devuelve el número de filas de la matriz  $A$ .*

# Información sobre las dimensiones

- `size(A, 2)`

*Devuelve el número de columnas de la matriz  $A$ .*

- `[m n]=size(A)`

*Asigna a  $m$  el número de filas de  $A$  y a  $n$  el de columnas.*

- `length(v)`

*Devuelve el número de coordenadas del vector  $v$ .*

- `length(A)`

*Devuelve el valor de  $\max(\text{size}(A))$ .*



# Aplicación

```
>> A=ones(2,5), v=[1 2 3 4 5]
```

```
>> size(A)
```

```
>> size(A,1)
```

```
>> size(A,2)
```

```
>> [m n]=size(A)
```

```
>> length(v)
```

```
>> length(A)
```

# Operaciones con vectores y matrices

*Un vector de  $n$  coordenadas se puede considerar como una matriz  $1 \times n$ . Por esta razón vamos a describir simultáneamente las operaciones con vectores y matrices.*

# Operaciones algebraicas

- Si A y B son matrices con dimensiones adecuadas y  $\lambda$  un escalar, las operaciones algebraicas en MATLAB se efectúan con las siguientes órdenes:

- $A(m,n)+B(m,n)$

*Suma A y B.*

- $A(m,n)-B(m,n)$

*Resta A y B.*

- $A(m,n)*B(n,k)$

*Multiplica A por B.*

*Estas matrices no son conforme para el producto, las columnas de A deben ser iguales a las filas de B      $A(m,n)*B(m,n)$  error*

# Operaciones algebraicas

- $k * A$

*Multiplica por  $k$  todos los elementos de  $A$ .*

- $A^n$

*Eleva la matriz  $A$  al número entero  $n$ .*

- $A.'$

*Genera la traspuesta de  $A$ .*

- $A'$

*Genera la conjugada traspuesta de  $A$ .*

- *Obsérvese que si  $A$  es una matriz real,  $A'$  y  $A.'$  coinciden.*

# Operaciones algebraicas

- `inv(A)`

*Calcula la inversa de A.*

- *Si u y v son dos vectores con el mismo número de coordenadas*

- `dot(u, v)`

*Calcula el producto escalar.*

- `cross(u, v)`

*Calcula el producto vectorial; es necesario que u y v tengan 3 coordenadas.*



# Aplicación

```
>> A=[1 2; 3 4]
```

```
>> B=[-1 4; 1 -2]
```

```
>> C=[1 2 3; 4 5 6]
```

```
>> A+B
```

```
>> 3*A
```

```
>> A*C
```

# Aplicación

```
>> C'
```

```
>> inv(B), B^(-1)
```

```
>> C' * (A-2*B)
```

```
>> u=[-1 2 3]; v=[5 -7 9];
```

```
>> dot(u,v)
```

```
>> cross(u,v)
```

Desarrollar  
página 38 del  
manual



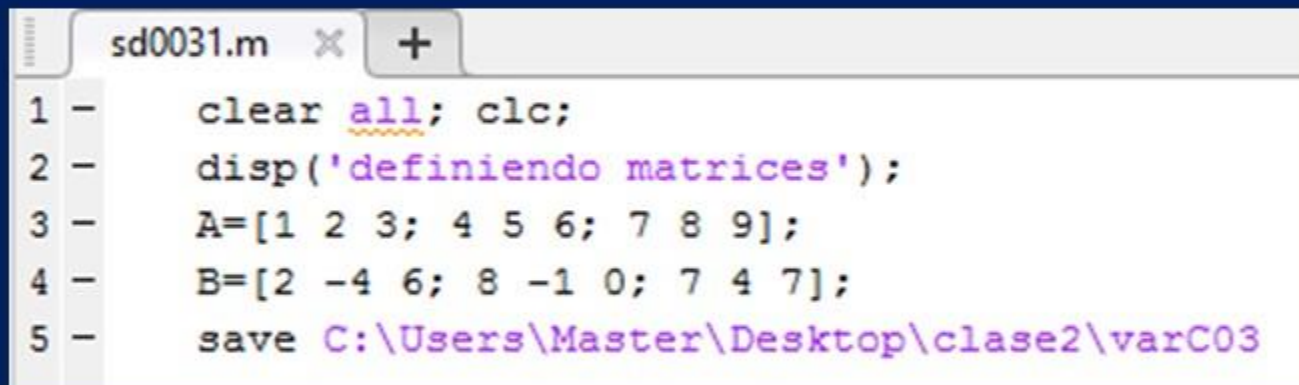
# Operadores aritméticos

Operador	Función	Definición
+	plus	Suma
-	minus	Resta
*	mtimes	Multiplicación
'	transpose	Traspuesta
^	mpower	Potenciación
\	mldivide	División izquierda
/	mrdivide	División derecha



# M-File: sd0031.m

- New Script

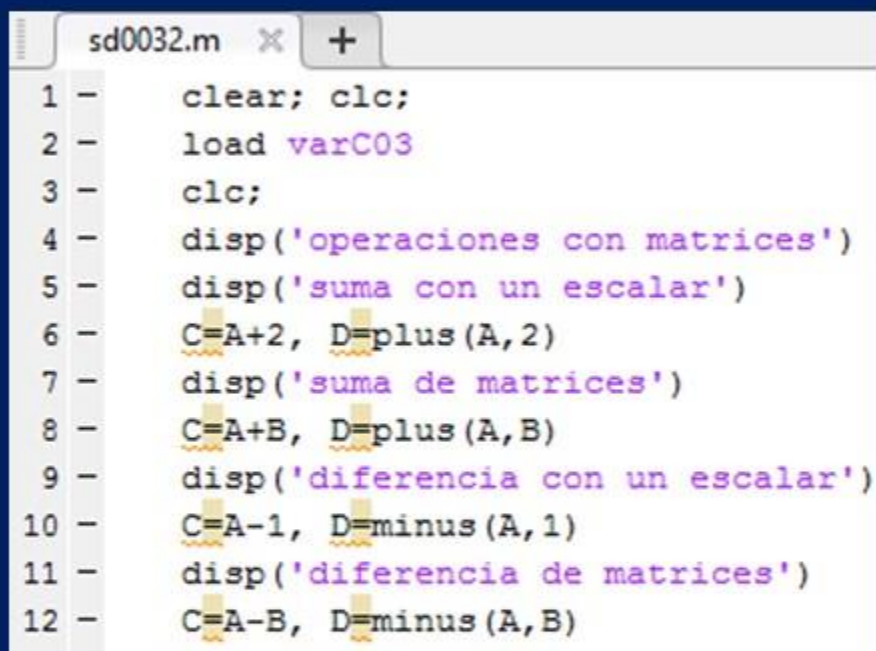


```
sd0031.m  x  +
1 -      clear all; clc;
2 -      disp('definiendo matrices');
3 -      A=[1 2 3; 4 5 6; 7 8 9];
4 -      B=[2 -4 6; 8 -1 0; 7 4 7];
5 -      save C:\Users\Master\Desktop\clase2\varC03
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0031

# M-File: sd0032.m

- New Script

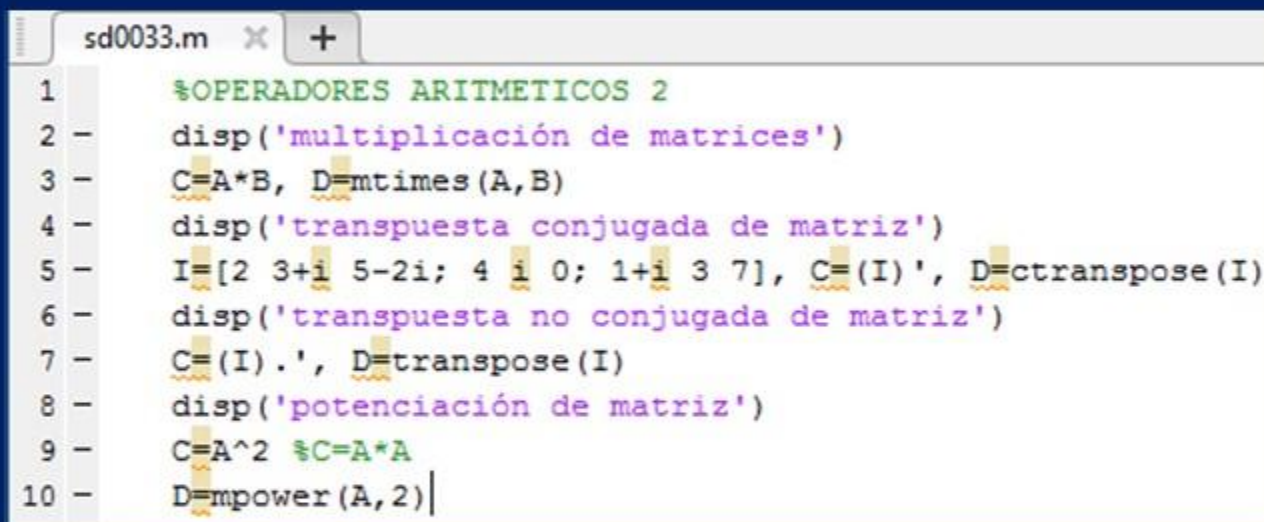


```
1 - clear; clc;
2 - load varC03
3 - clc;
4 - disp('operaciones con matrices')
5 - disp('suma con un escalar')
6 - C=A+2, D=plus(A,2)
7 - disp('suma de matrices')
8 - C=A+B, D=plus(A,B)
9 - disp('diferencia con un escalar')
10 - C=A-1, D=minus(A,1)
11 - disp('diferencia de matrices')
12 - C=A-B, D=minus(A,B)
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0032

# M-File: sd0033.m

- New Script

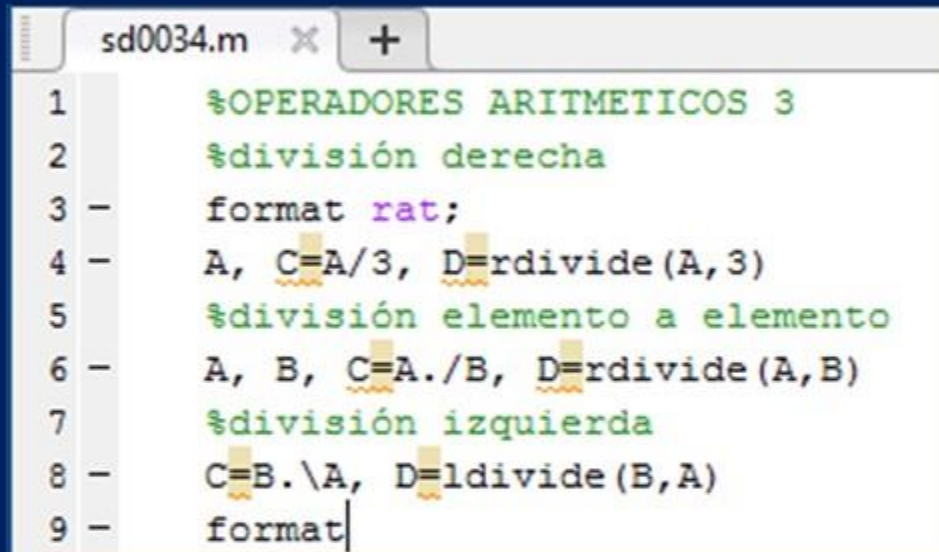


```
sd0033.m x +
1      %OPERADORES ARITMETICOS 2
2 -    disp('multiplicación de matrices')
3 -    C=A*B, D=mtimes(A,B)
4 -    disp('transpuesta conjugada de matriz')
5 -    I=[2 3+i 5-2i; 4 i 0; 1+i 3 7], C=(I)', D=ctranspose(I)
6 -    disp('transpuesta no conjugada de matriz')
7 -    C=(I).', D=transpose(I)
8 -    disp('potenciación de matriz')
9 -    C=A^2 %C=A*A
10 -    D=mpower(A,2)
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0033

# M-File: sd0034.m

- New Script



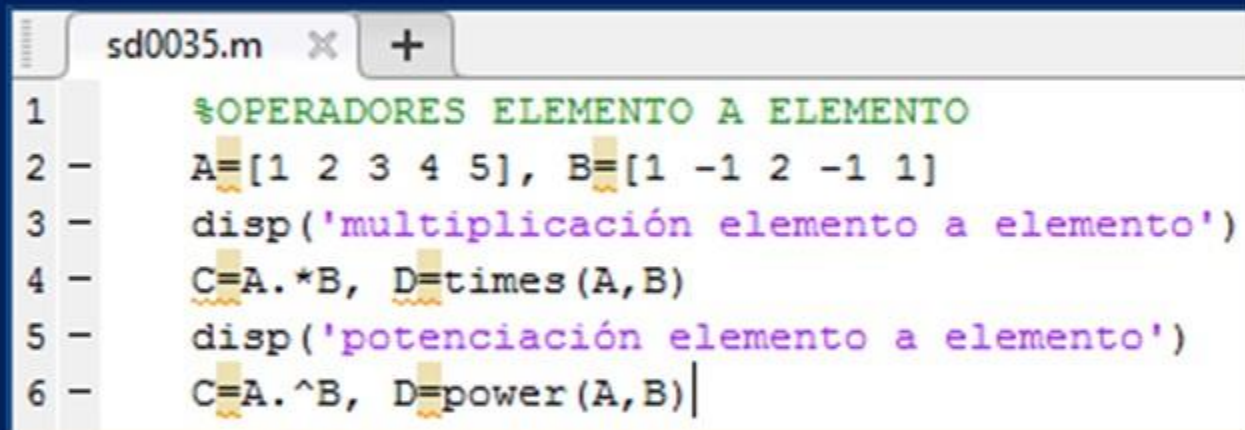
```
sd0034.m  x  +
1      %OPERADORES ARITMETICOS 3
2      %división derecha
3      - format rat;
4      - A, C=A/3, D=rdivide(A,3)
5      %división elemento a elemento
6      - A, B, C=A./B, D=rdivide(A,B)
7      %división izquierda
8      - C=B.\A, D=ldivide(B,A)
9      - format
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0034



# M-File: sd0035.m

- New Script



```
sd0035.m  x  +  
1      %OPERADORES ELEMENTO A ELEMENTO  
2 -    A=[1 2 3 4 5], B=[1 -1 2 -1 1]  
3 -    disp('multiplicación elemento a elemento')  
4 -    C=A.*B, D=times(A,B)  
5 -    disp('potenciación elemento a elemento')  
6 -    C=A.^B, D=power(A,B)|
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0035

# Sistema de ecuaciones lineales

Operadores para resolver sistema de ecuaciones lineales

Operator	Linear Equation	Syntax	Equivalent Syntax Using \
Backslash (\)	$A * x = B$	$x = A \setminus B$	Not applicable
Slash (/)	$x * A = B$	$x = B / A$	$x = (A' \setminus B')'$

# Solución de sistemas de ecuaciones lineales $Ax=b$

donde:

$x$  y  $b$  son vectores columna ( $b$  conocido)

$A$  es una matriz cuadrada invertible

$$A(n,n) * x(n,1) = b(n,1)$$

Solución

$$\text{inv}(A(n,n)) * A(n,n) * x(n,1) = \text{inv}(A(n,n)) * b(n,1)$$

$$x(n,1) = \text{inv}(A(n,n)) * b(n,1)$$

# Aplicación

Resolver el sistema

$$\begin{array}{rclcl} 2x - y + z & = & 3 \\ x + y & = & 3 \\ y - 3z & = & -7 \end{array}$$

Hallar  $x, y, z$

Solución:

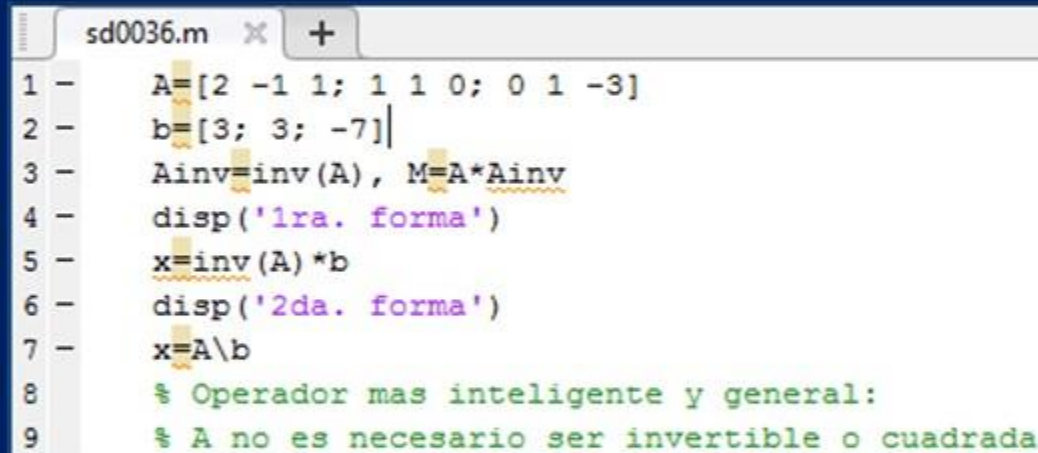
$$A = \begin{bmatrix} 2 & -1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & -3 \end{bmatrix}$$

$$b = \begin{bmatrix} 3 \\ 3 \\ -7 \end{bmatrix}$$



# M-File: sd0036.m

- New Script



```
sd0036.m  X  +
1 - A=[2 -1 1; 1 1 0; 0 1 -3]
2 - b=[3; 3; -7]
3 - Ainv=inv(A), M=A*Ainv
4 - disp('1ra. forma')
5 - x=inv(A)*b
6 - disp('2da. forma')
7 - x=A\b
8   % Operador mas inteligente y general:
9   % A no es necesario ser invertible o cuadrada
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0036

# Solución de sistemas de ecuaciones lineales $xA=b$

donde:

$x$  y  $b$  son vectores fila ( $b$  conocido)

$A$  es una matriz cuadrada invertible

$$x(1,n) * A(n,n) = b(1,n)$$

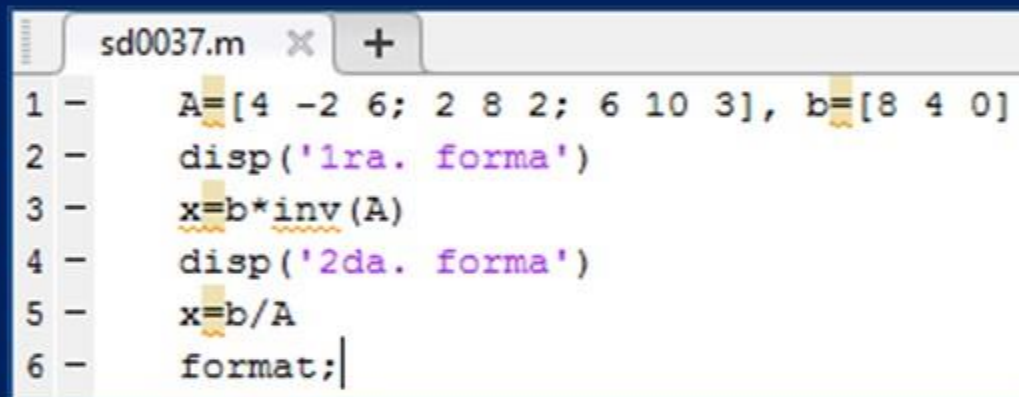
Solución

$$x(1,n) * A(n,n) * \text{inv}(A(n,n)) = b(1,n) * \text{inv}(A(n,n))$$

$$x(1,n) = b(1,n) * \text{inv}(A(n,n))$$

# M-File: sd0037.m

- New Script



```
sd0037.m  x  +
1 -      A=[4 -2 6; 2 8 2; 6 10 3], b=[8 4 0]
2 -      disp('1ra. forma')
3 -      x=b*inv(A)
4 -      disp('2da. forma')
5 -      x=b/A
6 -      format;
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0037

# Definición de variables



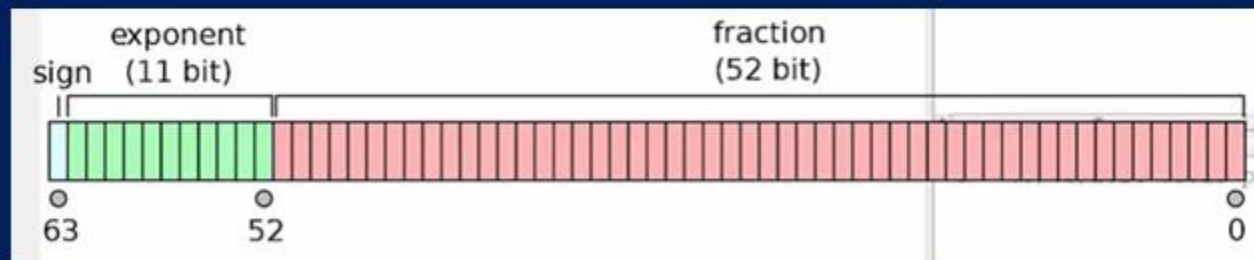
# Double

- Matlab es un programa preparado para trabajar con vectores y matrices como caso particular también trabaja con variables escalares - matrices de dimensión 1
- Siempre trabaja con doble precisión, es decir guardando en 8 bytes = 64 bits
- Double = números reales de doble precisión

Double ➡ 8 bytes ➡ 64 bits

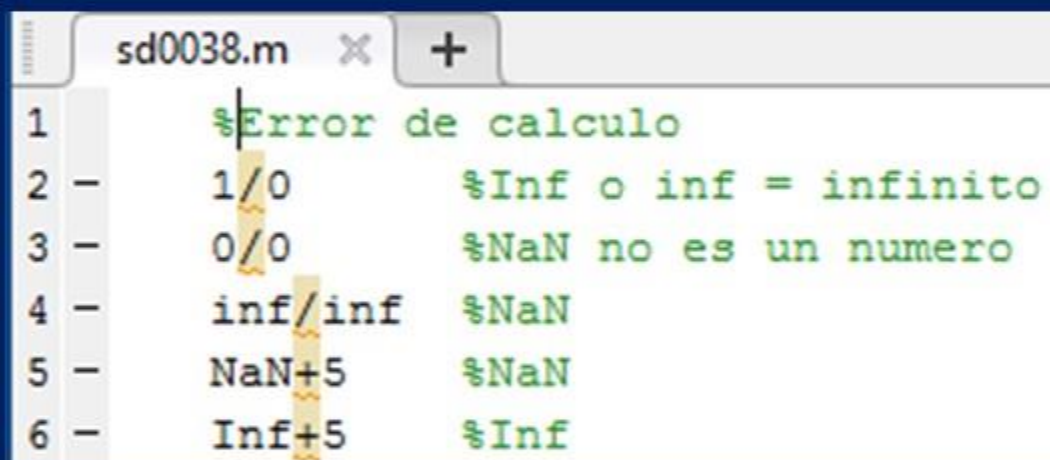
Double  $\Rightarrow$  8 bytes  $\Rightarrow$  64 bits

- Double = números reales de doble precisión
- 1er bit es para el signo 0=+, 1=-
- siguientes 11 bits son para la potenciación
- los restantes 52 bits es para la fracción



# M-File: sd0038.m

- New Script



```
sd0038.m  X  +
1      %Error de calculo
2 -    1/0      %Inf o inf = infinito
3 -    0/0      %NaN no es un numero
4 -    inf/inf  %NaN
5 -    NaN+5    %NaN
6 -    Inf+5    %Inf
```

- Clic Run o pulsar F5 o en la ventana de comando escribir sd0038

# Operaciones útiles de como flotante

- `eps` = devuelve la diferencia entre 1 y el numero de coma flotante inmediatamente superior, es decir representa el número más pequeño que puede sumar a 1.

$$(1.0+eps) > 1.0$$

- representa la exactitud relativa de la aritmética del computador da una idea de la precisión o del numero de cifras almacenadas

```
eps    % 2.2204e-016
```

```
%devuelve el número más pequeño con  
el que se puede trabajar
```

```
realmin    % 2.2251e-308
```

```
%devuelve el número más grande con  
el que se puede trabajar
```

```
realmax    % 1.7977e+308
```



# Integer, float y logical

```
% ahorrar memoria 8, 16, 32 y 64  
% bits con signo o sin signo  
a=int32(5)    % entero de 32 bit  
A=[4 5 6; 7 8 9] %Double  
% entero de 32 bit ocupa la mitad  
B=int32(A)  
whos
```

Número de bytes	Número de bits	Tipo de entero (Con signo)	Tipo de entero (Sin signo)
1	8	int8	uint8
2	16	int16	uint16
4	32	int32	uint32
8	64	int64	uint64

```
% valor mas pequeño que puede formarse  
intmin('int8')      %-128  
intmin('int16')     %-32768  
intmin('int32')     %-2147483648  
intmin('int64')     %-9223372036854775808
```

```
% valor mas grande que puede formarse  
intmax('int8')      %127  
intmax('int16')     %32767  
intmax('int32')     %2147483647  
intmax('int64')     %9223372036854775807
```

# M-File: sd0037.m

- New Script
- Clic Run o pulsar F5 o en la ventana de comando escribir sd0037