

# Matlab básico

## INTRODUCCIÓN

Matlab es el nombre abreviado de “MATrix LABoratory” y es una herramienta con la cual se pueden realizar distintos cálculos tanto de números escalares, así como arreglos de estos (matrices y vectores). También hay distintas aplicaciones en las que puede ser utilizada esta plataforma, así como sus diferentes toolbox, el principal ejemplo es Simulink.

En la Figura 1. Se puede observar la ventana inicial de Matlab, con diferentes colores se resaltan las partes más importantes y son explicadas más adelante.

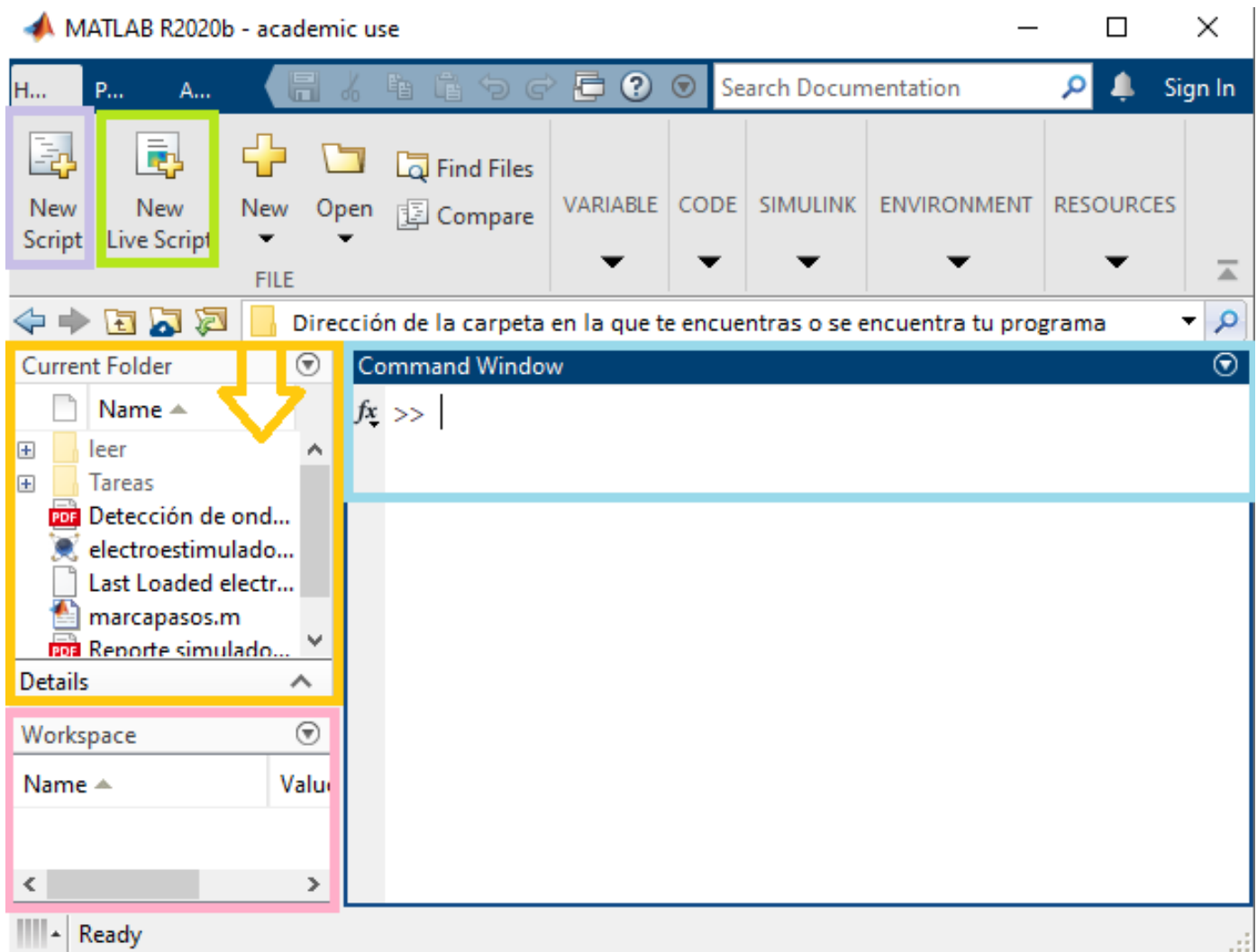


Figura 1. Ventana principal de Matlab

**Command Window (Ventana de comandos):** En este espacio se escriben y muestran todas las operaciones llevadas a cabo. Cada `>>` representa una línea de código nueva y para realizarla existen distintas funciones que serán vistas más adelante. Para cambiar de línea y ejecutar la ya existente se utiliza la tecla enter. Para limpiar la ventana de comandos se utiliza el comando **clc**.

**Workspace:** En este espacio se muestran todas las variables establecidas durante el uso de Matlab. Cada variable puede ser visualizada dándole doble clic, generando una nueva ventana en la que se podrá visualizar la variable completa (sobre todo los arreglos numéricos). Para vaciar el Workspace se utiliza el comando **clear all**.

**Current Folder:** Como lo menciona en la barra de dirección, la carpeta seleccionada en esta será la mostrada en este apartado, mostrando todos los documentos incluidos en esta. Es importante seleccionar la carpeta indicada al momento de correr un programa.

**New Script y Live Script:** Ambos son utilizados para abrir y generar archivos de Matlab (.m) y (.live) para realizar un código. Estos archivos pueden ser guardados, editados y pausados para poder evaluar su funcionamiento.

La última función importante en la ventana inicial de Matlab es el **Command History**, en las versiones más antiguas aparecía al igual que el Workspace, como una ventana adicional, ahora en las más recientes para poder acceder a el es necesario utilizar la flecha hacia arriba (en el teclado) y así poder visualizar todas las líneas de comando ya escritas anteriormente en la plataforma. Estas pueden ser seleccionadas y editadas para volver a ser utilizadas.

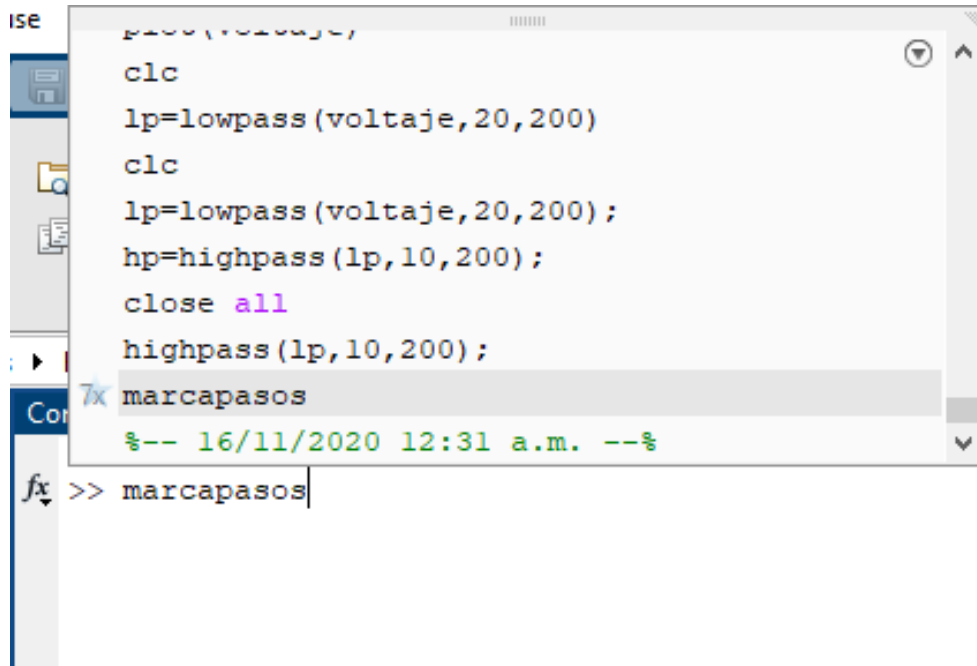


Figura 2. Vista del Command History

## VARIABLES

Para establecer una variable es necesario nombrarla, es decir establecer una letra, conjunto de letras o letras y números con la que esta será identificada. La variable será expresada después de este nombre y un signo igual (=) por lo que es importante tomar en cuenta las siguientes consideraciones (estas también aplican al momento de crear una nueva función o un script):

- No se deben utilizar únicamente números, ya que esto significa un error. **Ej. 1=10.** Al ser utilizado Matlab para realizar operaciones matemáticas, esta expresión no es correcta.
- No se pueden utilizar espacios, si es necesario separar o dar espacio en el nombre establecido se debe utilizar un guion bajo. **Ej. primera\_prueba**
- No se pueden nombrar variables o scripts utilizando comandos o funciones ya establecidos por Matlab, esto generará que el comando deje de funcionar como es establecido y se vuelva una variable o programa únicamente.
- Las variables se pueden guardar hasta con 63 caracteres alfanuméricos, sin acentos o símbolos.

**\*Si queremos que no se muestre la variable, operación, etc. únicamente se utiliza un punto y coma (;) al final de la línea.**

## Ejemplo

```
>> a=1

a =

     1

>> a1=1

a1 =

     1

>> a_b=1

a_b =

     1
```

Figura 3. Ejemplos de variables

```
>> l=10
l=10
↑
Error: Incorrect use of '=' operator. To assign a value to
a variable, use '='. To compare values for equality, use
'=='.

>> a b=10
Unrecognized function or variable 'a'.
```

Figura 4. Errores en establecimiento de variables

```
>> exp(2)

ans =

     7.3891

>> exp=2

exp =

     2

>> exp(2)
Index exceeds the number of array elements (1).

'exp' appears to be both a function and a variable. If this is
unintentional, use 'clear exp' to remove the variable 'exp' from
the workspace.
```

Figura 5. Ejemplo del uso de comandos establecidos por Matlab

## OPERACIONES BÁSICAS

Para realizar operaciones básicas se utilizan los símbolos y comandos mostrados en la siguiente tabla:

Tabla 1. Símbolos y comandos para operaciones básicas

Operación	Símbolo o comando	Expresión en Matlab
<b>SUMA</b>	+	a+b
<b>RESTA</b>	-	a-b
<b>MULTIPLICACIÓN</b>	*	a*b
<b>DIVISIÓN</b>	/	a/b
<b>POTENCIA</b>	^	a^ b
<b>RAIZ CUADRADA</b>	sqrt	sqrt(a)

Al realizar una operación larga se debe tomar en cuenta la jerarquía de operaciones, ya que al igual que al utilizar una calculadora, se deberán utilizar paréntesis para respetar este orden y obtener el resultado deseado.

El orden de precedencia es el siguiente:

1. Potencia
2. Multiplicación y división
3. Suma y resta

### Tips:

- Al momento de utilizar paréntesis abre y cierra el paréntesis antes de escribir el contenido, así aseguras tener los paréntesis completos.
- Se recomienda utilizar paréntesis separando cada operación a realizar cuando se comienza a utilizar la plataforma, para así evitar errores en los resultados.
- Establecer las variables a utilizar antes de realizar las operaciones puede ayudar a que llevarlas a cabo sea más sencillo utilizando la variable en lugar de números.

## Ejemplo

$$x = a^b + b \frac{4b}{c} - \sqrt{ab}$$

a=2.67

b=12

c=9.76

```
>> a=2.67;
>> b=12;
>> c=9.76;
>> x=(a^b)+(b*(4*b)/c)-sqrt(a*b)

x =

1.3131e+05
```

Figura 6. Resultado de la operación de ejemplo en Matlab.

Existen más comandos para realizar operaciones, como lo son las funciones trigonométricas, números establecidos como  $\pi$  o  $e$ , así como números complejos.

Tabla 2. Otros comandos para operaciones

Función	¿Qué hace?
<b>pi</b>	Número pi (3.1416)
<b>inf</b>	Infinito
<b>nan</b>	Magnitud no numérica
<b>i y j</b>	Números complejos
<b>abs(x)</b>	Valor absoluto
<b>exp(x)</b>	Número Euler. $e^x$
<b>realmin</b>	Número real positivo más pequeño utilizable
<b>gcd(m,n)</b>	Máximo común divisor
<b>lcm(m,n)</b>	Mínimo común múltiplo
<b>log(x)</b>	Logaritmo natural
<b>log2(x)</b>	Logaritmo base 2
<b>log(10)</b>	Logaritmo base 10
<b>mod(x,y)</b>	Módulo después de una división
<b>rem(x,y)</b>	Residuo de una división de enteros
<b>nthroot(x,n)</b>	Raíz n-sima de x
<b>sign(x)</b>	Signo del argumento si x es un número real
<b>realmax</b>	Número real positivo más grande utilizable
<b>ceil(x)</b>	Redondea a infinito
<b>fix(x)</b>	Redondea a cero
<b>floor(x)</b>	Redondea a menos infinito
<b>round(x)</b>	Redondea al entero más próximo
<b>sin(x)</b>	Seno
<b>cos(x)</b>	Coseno
<b>tan(x)</b>	Tangente
<b>csc(x)</b>	Cosecante
<b>sec(x)</b>	Secante
<b>cot(x)</b>	Cotangente
<b>sind(x)*</b>	Seno(grados)
<b>sinh(x)*</b>	Seno hiperbólico (radianes)
<b>asin(x)*</b>	Arcoseno (radianes)
<b>asind(x)*</b>	Arcoseno(grados)
<b>asinh(x)*</b>	Arcoseno hiperbólico

\*En estas opciones puede ser sustituida la función trigonométrica sin por cualquiera de las demás mencionadas anteriormente

El resultado automático de una operación se muestra como **ans**, pero es recomendable nombrar el resultado con una variable diferente, ya que estas se sobrescriben y se puede perder información deseada.

Al realizar las operaciones y establecer variables, estas se pueden visualizar de diferentes maneras. La manera preestablecida en Matlab es en base 10 y con cuatro decimales después del punto, pero en la Tabla 3. Se muestran más formatos de visualización de números, tanto decimales como otras bases (hexadecimal, double, etc.)

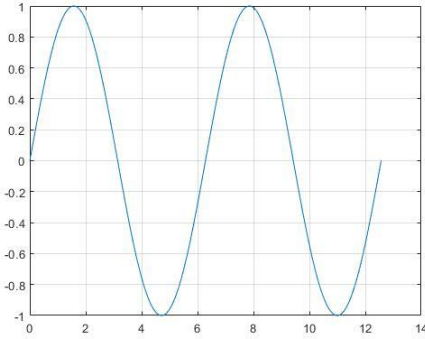
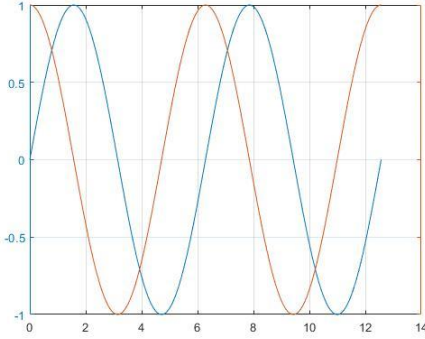
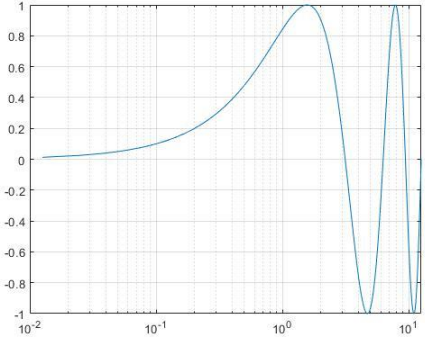
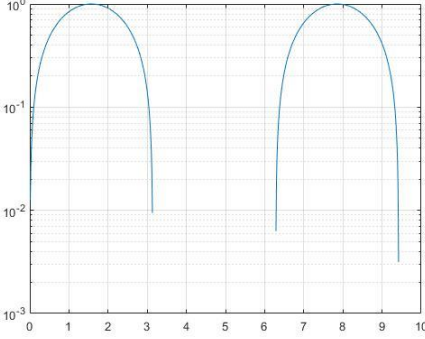
*Tabla 3. Formatos utilizados en Matlab*

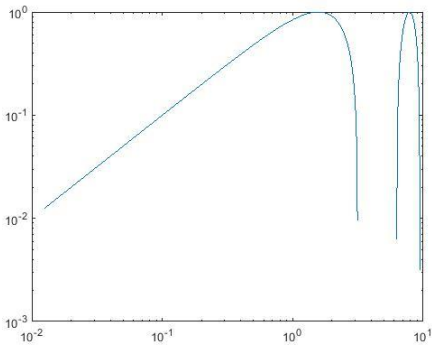
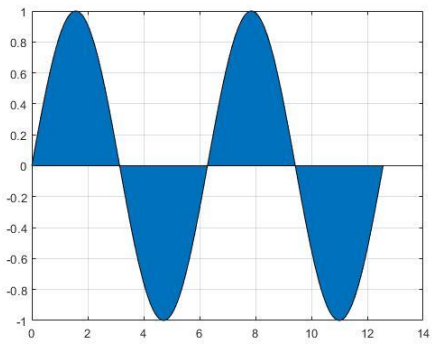
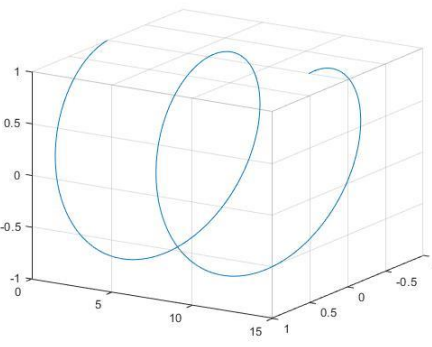
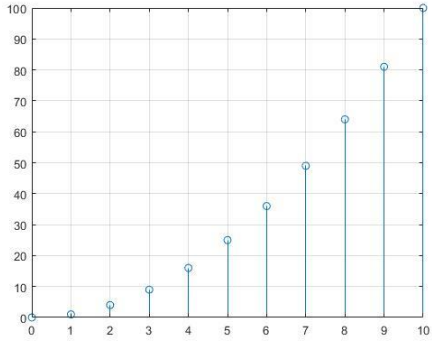
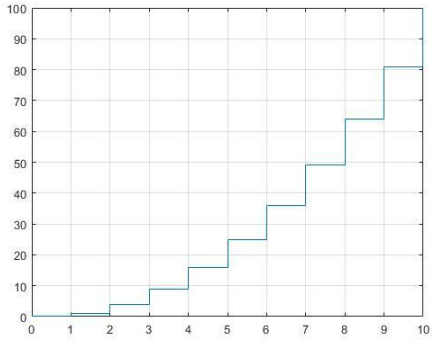
Tipo (comando)	Resultado	Ejemplo
<b>Format short</b>	Formato coma fija con 4 dígitos después de la coma (formato establecido)	3.1416
<b>Format long</b>	Formato coma fija con 14 o 15 dígitos después de la coma	3.14159265358979
<b>Format short e</b>	Formato coma flotante con 4 dígitos después de la coma	3.1416e+000
<b>Format long e</b>	Formato coma flotante con 14 o 15 dígitos después de la coma	3.14159265358979e+000
<b>Format short g</b>	La mejor entre coma fija o flotante con 4 dígitos después de la coma	3.1416
<b>Format long g</b>	La mejor entre coma fija o flotante con 14 o 15 dígitos después de la coma	3.14159265358979
<b>Format short eng</b>	Notación científica con 4 dígitos después de la coma y un exponente de 3	3.1416e+000
<b>Format long eng</b>	Notación científica con 16 dígitos significantes y un exponente de 3	3.14159265358979e+000
<b>Format bank</b>	Formato coma fija con 2 dígitos después de la coma	3.14
<b>Format hex</b>	Hexadecimal	400921fb54442d18
<b>Format rat</b>	Aproximación racional	355/112
<b>Format +</b>	Positivo negativo o espacio en blanco.	+

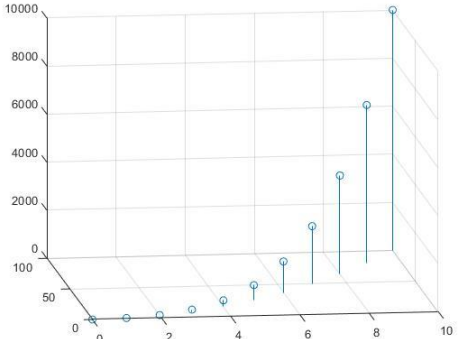
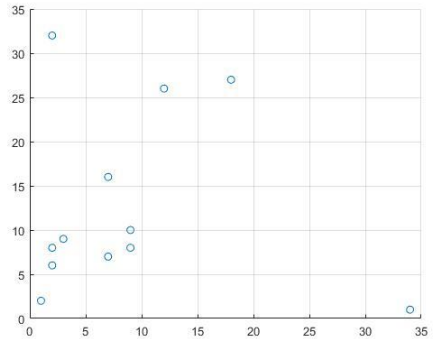
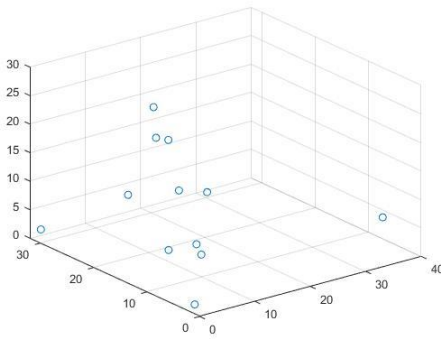
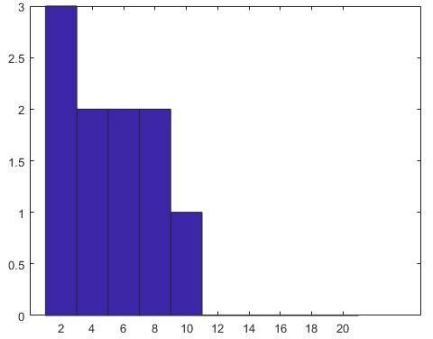
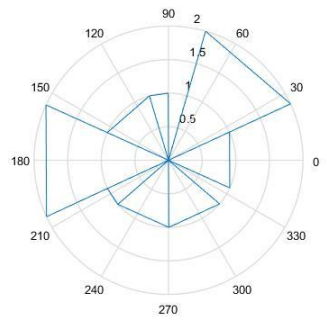
# GRÁFICAS

Existen distintas maneras y tipos de gráficas (2D y 3D) en Matlab, a continuación, se muestran los comandos utilizados y ejemplos.

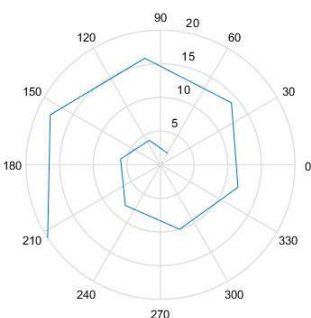
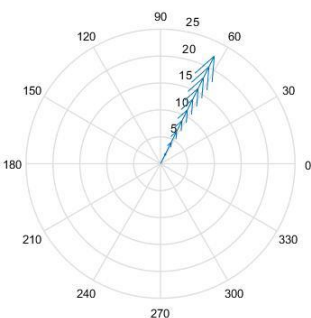
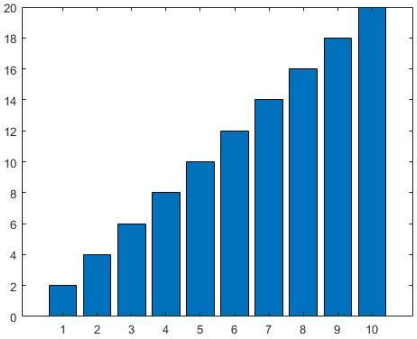
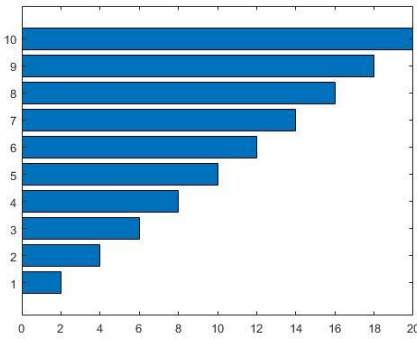
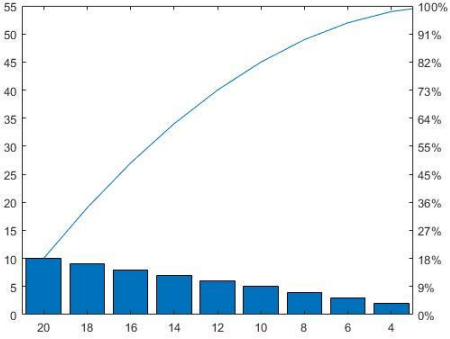
Tabla 4. Comandos para los diferentes tipos de gráficas

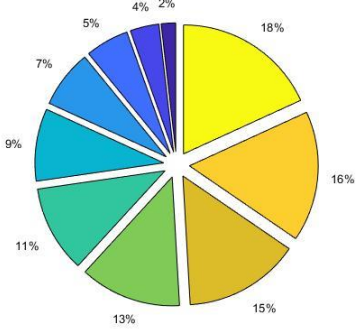
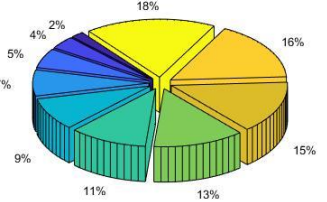
Comando	Función	Ejemplo
<b>plot(x,y)</b>	Realiza la gráfica en 2d de una función o dos valores establecidos(x,y).	
<b>plotyy(x,y,x2,y2)</b>	Realiza la gráfica en 2d de dos función o cuatro valores establecidos (x1,y1,x2,y2).	
<b>semilogx(x,y)</b>	Realiza la gráfica en 2d de una función o dos valores establecidos (x,y) con una escala logarítmica en el eje x.	
<b>semilogy(x,y)</b>	Realiza la gráfica en 2d de una función o dos valores establecidos (x,y) con una escala logarítmica en el eje y.	

<b>loglog(x,y)</b>	Realiza la gráfica en 2d de una función o dos valores establecidos (x,y) con una escala logarítmica.	
<b>area(x,y)</b>	Realiza la gráfica en 2d de una función o dos valores establecidos (x,y) marcando el área de la gráfica.	
<b>plot3(x,y,z)</b>	Realiza la gráfica en 3d de una matriz o tres valores establecidos (x,y,z).	
<b>stem(x,y)</b>	Realiza la gráfica de funciones discretas(x,y)	
<b>stairs(x,y)</b>	Realiza la gráfica de escalera de una función (x,y)	

<p><b>stem3(x,y,z)</b></p>	<p>Realiza la gráfica en 3d de una matriz o tres valores establecidos (x,y,z).</p>	
<p><b>scatter(x,y)</b></p>	<p>Realiza la gráfica de dispersión de una función(x,y).</p>	
<p><b>scatter(x,y,z)</b></p>	<p>Realiza la gráfica de dispersión de una matriz o función 3D (x,y,z).</p>	
<p><b>hist(x,y)</b></p>	<p>Realiza el histograma de una dos variables, matriz o función (x,y).</p>	
<p><b>Rrose(x,y)</b></p>	<p>Realiza la rosa de los vientos de una función o dos variables (x,y).</p>	



<b>polar(x,y)</b>	Realiza la gráfica polar de una función o dos variables (x,y)	
<b>compass(x,y)</b>	Realiza la gráfica de dirección de una función o dos variables (x,y)	
<b>bar(x,y)</b>	Realiza la gráfica de barras de una función o dos variables (x,y)	
<b>barh(x,y)</b>	Realiza la gráfica de barras horizontal de una función o dos variables (x,y)	
<b>pareto(x,y)</b>	Realiza la gráfica de Pareto de una función o dos variables (x,y)	

<b>pie(x,y)</b>	Realiza la gráfica de pastel de una función o dos variables (x,y)	
<b>pie3(x,y)</b>	Realiza la gráfica de pastel 3D de una función o dos variables (x,y)	

Al realizar las gráficas existen distintas herramientas que nos ayudan a modificar las propiedades de las gráficas y estos se muestran a continuación:

Tabla 5. Comandos para modificar propiedades de las gráficas.

Comando	Descripción	Ejemplo
<b>axis</b>	Modifica las propiedades de los ejes	<code>axis([0 1000 0 1])</code> Crea un eje x de 0 a 1000 y un eje y de 0 a 1.
<b>clf</b>	Borra la ventana Figure	
<b>close</b>	Cierra la ventana Figure	
<b>figure</b>	Crea o selecciona una ventana Figure	<code>figure()</code> Crea una nueva figura con un número sucesivo a la ya existente <code>figure(4)</code> Selecciona o crea la figura 4.
<b>grid</b>	Coloca o quita una cuadrícula	<code>grid on</code> Coloca la cuadrícula <code>grid off</code> Quita la cuadrícula
<b>gtext</b>	Coloca texto con el mouse	<code>gtext('texto')</code>
<b>hold</b>	Mantiene la gráfica actual y permite graficar más de dos gráficas en una misma figura.	<code>hold on</code> Permite graficar manteniendo la gráfica actual <code>hold off</code> Elimina la/s grafica/s actuales al volver al graficar
<b>subplot</b>	Crea subgráficas dentro de una misma figura	<code>subplot(x,y,z)</code> x-filas en las que se dividirá la figura y-columnas en las que se dividirá la figura z-posición en la que se realizará la gráfica (va de 1 a x*y) *Cada gráfica cuenta con características independientes
<b>title</b>	Coloca el título de la gráfica	<code>title('título')</code>
<b>xlabel</b>	Coloca una etiqueta en el eje x	<code>xlabel('eje x')</code>
<b>ylabel</b>	Coloca una etiqueta en el eje y	<code>ylabel('eje y')</code>
<b>legend</b>	Coloca una descripción de cada gráfica que se muestra en una figura dependiendo sus características	<code>Legend('primera gráfica','segunda gráfica')</code> *Siempre se debe tomar en cuenta el orden en el que se realizaron las gráficas

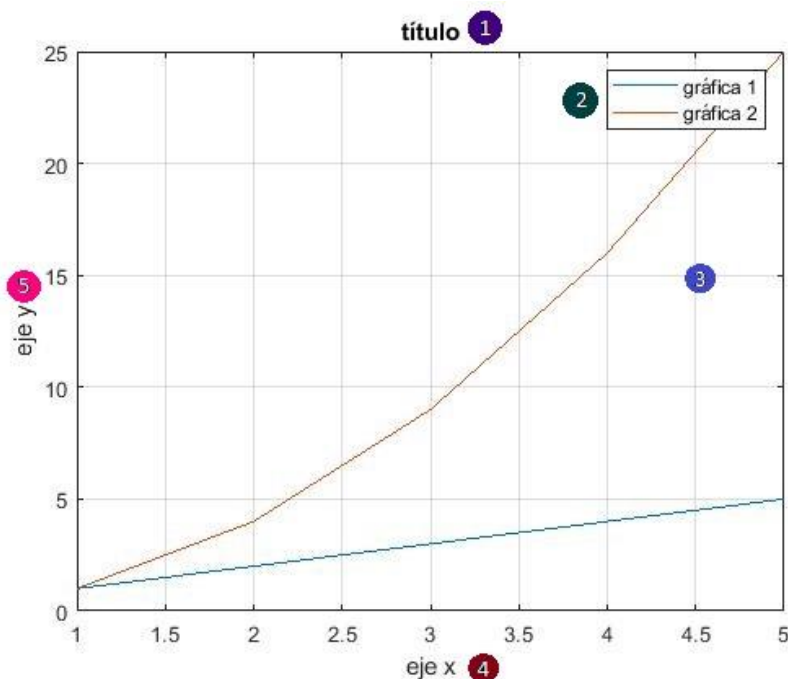


Figura 7. Ejemplo de gráfica con propiedades modificadas

1. title
2. legend
3. grid
4. xlabel
5. ylabel

```
>> x=[1:10];
y=x;
y2=x.^2;
plot(x,y)
title('título')
hold on
plot(x,y2)
xlabel('eje x')
ylabel('eje y')
grid on
legend('gráfica 1','gráfica 2')
```

Figura 8.1. Código utilizado para modificar propiedades

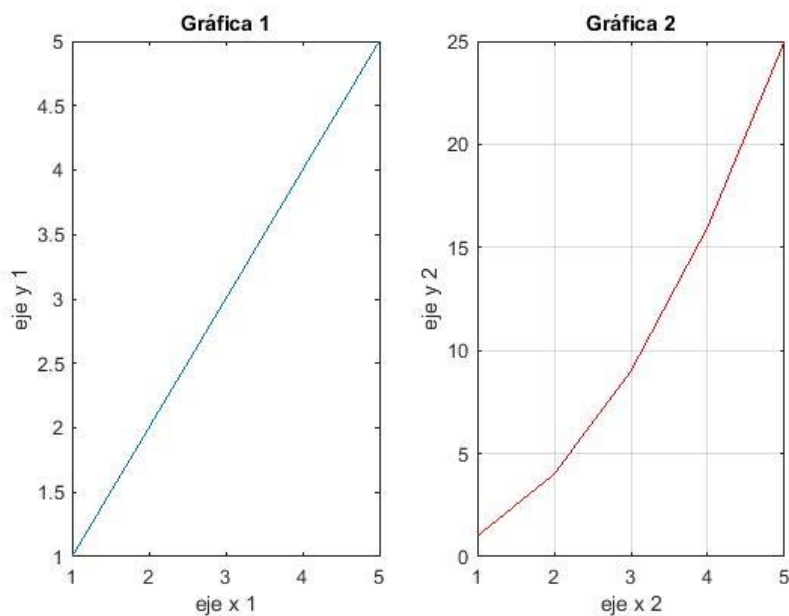


Figura 9. Ejemplo de subplot con propiedades modificadas.

```
>> subplot(1,2,1)
plot(x,y)
title('Gráfica 1')
xlabel('eje x')
ylabel('eje y 1')
subplot(1,2,2)
plot(x,y2)
title('Gráfica 2')
xlabel('eje x 2')
ylabel('eje y 2')
grid on
```

Figura 8.1. Código utilizado para subplot

Se utilizaron los comandos ya mencionados anteriormente para poder modificar y obtener distintas características en la figura deseada. Además se muestra la manera de utilizar la división de figura utilizando el comando subplot. Como se menciona, cada gráfica cuenta con sus propias características y puede ser editado individualmente. En este ejemplo se muestra una división en 1 fila y 2 columnas, por lo que  $z=2$ .

Al igual que las propiedades, las características de la gráfica pueden ser modificadas también, cambiando color, grosor de línea, tipo de gráfico y algunas de las propiedades de dos maneras distintas, utilizando comandos y la segunda utilizando el Property Editor. Se muestran ambas formas a continuación:

Tabla 6. Comandos para modificar características de las gráficas

tipodegráfico(x,y,'especificadores de línea', 'Propiedades','Valores')			
	Símbolo/Comando	Significado	Forma de uso
Color	y	amarillo	plot(x,y,'símbolo de color seleccionado') plot(x,y,'m')
	m	magenta	
	c	cyan	
	r	rojo	
	g	verde	
	b	azul	
	w	blanco	
	k	negro	
Estilo de línea	-	línea continua	plot(x,y,' símbolo de estilo de línea seleccionado') plot(x,y,'--')
	:	línea a puntos	
	-.	línea barra punto	
	--	líneas discontinuas	
Marcadores	.	marcador de puntos	plot(x,y,' símbolo de marcador seleccionado') plot(x,y,'o')
	o	marcador de círculos	
	x	marcador de equis	
	+	marcador de símbolo más	
	*	marcador de asteriscos	
	s	marcador de cuadros	
	d	marcador de diamantes	
	^	marcador de triángulo	
	v	marcador de triángulo invertido	
	<	marcador de triángulo a la izquierda	
	>	marcador de triángulo a la derecha	
	p	marcador de estrella de 5 puntas	
	h	marcador de estrella de 6 puntas	
	LineWidth	Modifica el grosor de la línea de la gráfica	plot(x,y,'LineWidth',3) *Mayor número = mayor grosor

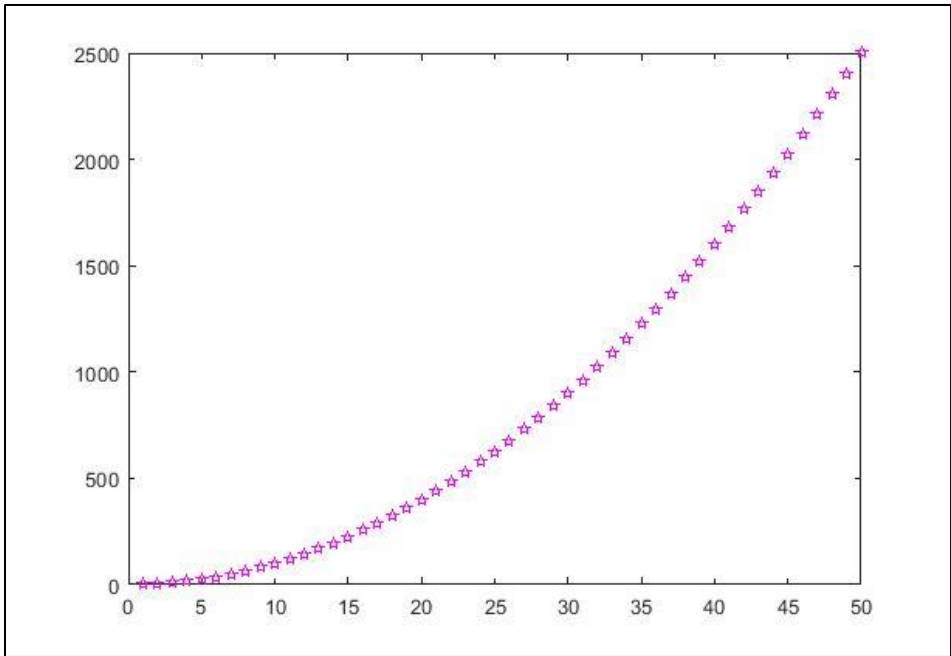
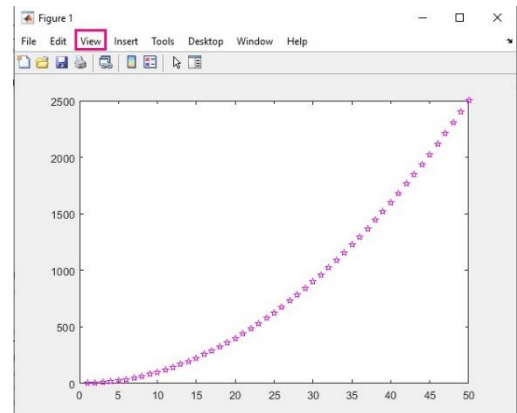


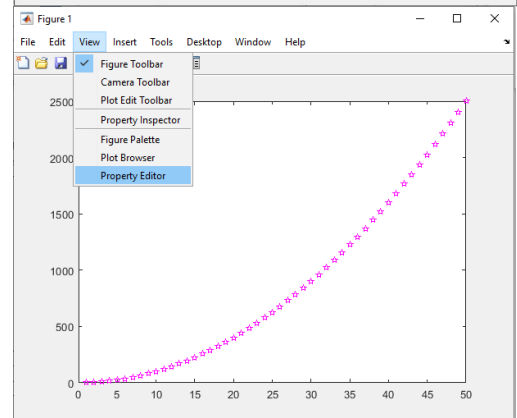
Figura 9. Ejemplo de gráfica modificada utilizando comandos (plot(x,y,'mp'))

Para modificar manualmente estas y más propiedades de la gráfica se deben seguir los siguientes pasos:

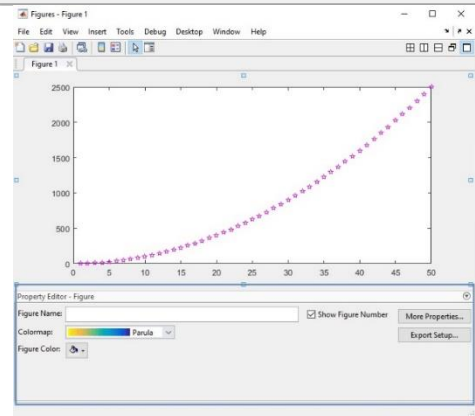
1. Seleccionar el menú **view** de la figura a modificar



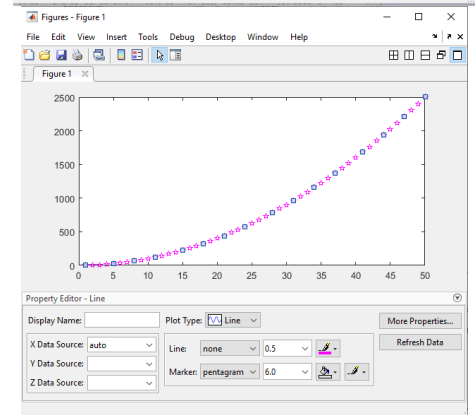
2. Seleccionar **Property Editor**



3. Se abrirá la siguiente ventana (puede aparecer en cualquier lado de la gráfica o en una ventana aparte)



4. Seleccionar la característica a editar (línea, fondo, título, cuadrícula, etc)



5. Modificar la característica deseada (**color**, **tipo de línea o marcador**, **ancho de línea**, etc)

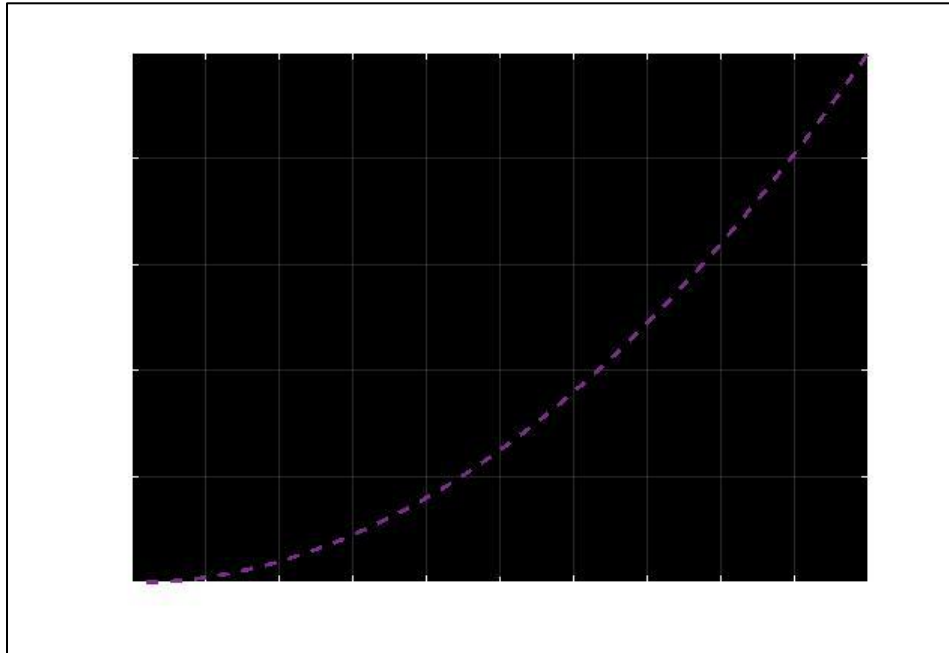
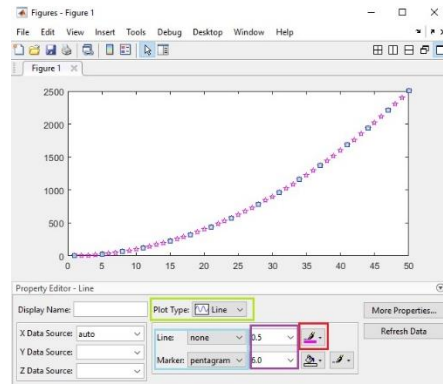


Figura 10. Ejemplo de gráfica modificada con el Property Editor

Para graficar es necesario establecer los datos que se quieren visualizar, por lo que debemos considerar el intervalo en el que se va a generar la gráfica (variable independiente) y la función o los valores evaluados de la variable independiente. La manera en la que se pueden establecer los valores a graficar son las siguientes:

### 1. Vectores

Para definir las variables a graficar en forma de vector es necesario únicamente introducir los datos deseados entre corchetes. Existen diferentes maneras de establecer un intervalo deseado, ya sea para la variable dependiente o la independiente.

Ej.

- $x=[1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10]$  De esta manera se debe introducir valor por valor deseado
- $x=[A:i:B]$  Se genera un vector de A a B con espaciado de i. Si no se establece una i el espaciado será de 1.
- $x=\text{linspace}(A,B,n)$  Se genera un vector de n datos en el intervalo de A a B con espaciado  $(B-A)/(n-1)$ . Si no se establece una n se generarán 100 datos con un espaciado igual entre cada dato.

Las siguientes formas son utilizadas más comúnmente para definir la función de la variable dependiente y es necesario evaluarla en el intervalo de la variable independiente después de ser definida.

## 2. Polinomios

Los polinomios se definen igual que los vectores, únicamente que la posición en el vector establecerá el coeficiente que tendrá la variable. Además, los polinomios tienen comandos específicos para su aplicación que se verán más adelante.

Ej.  $x^2 \rightarrow P = [1 \ 0 \ 0]$  de esta manera se establece la función

Se utiliza `polyval` para evaluar la función en el intervalo establecido

**`y=polyval(P,x)`**

## 3. Ecuaciones

Se pueden establecer de distintas maneras las ecuaciones de la función a utilizar, la más sencilla es simplemente definir nuestro intervalo primero y después establecer la función con la variable del intervalo, así automáticamente se va a evaluar y entregará el vector a utilizar para la gráfica.

Ej. `x=[1:10]`

`y=x.^2`

La siguiente opción para establecer una función es utilizando variables simbólicas, donde se utiliza el comando `syms` para poder definir la función y después poder evaluarla con `subs`.

Ej. `syms x`

`y=x^2`

`x=[1:10]`

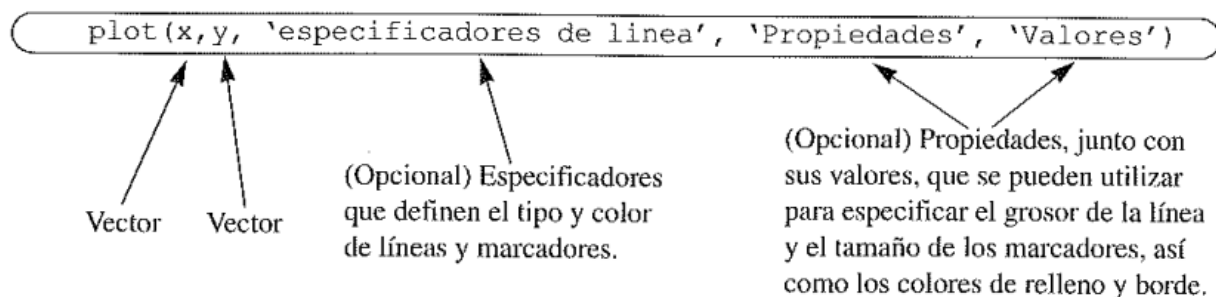
**`subs(y)`**

La última manera es utilizando el comando `inline`, que establecerá la función requerida y puede ser evaluada únicamente utilizando el nombre de la variable y el intervalo requerido entre paréntesis.

Ej. `y=inline('x^2')`

`y=y(1:10)`

Para realizar una gráfica se debe seguir el siguiente formato en el código, utilizando los comandos ya mencionados en las tablas anteriores:



Se muestran a continuación la obtención de una gráfica con la misma función utilizando las diferentes formas ya mencionadas, además de los comandos mencionados para modificar las propiedades y características de la imagen:

$x = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10]$

$y = x^2$

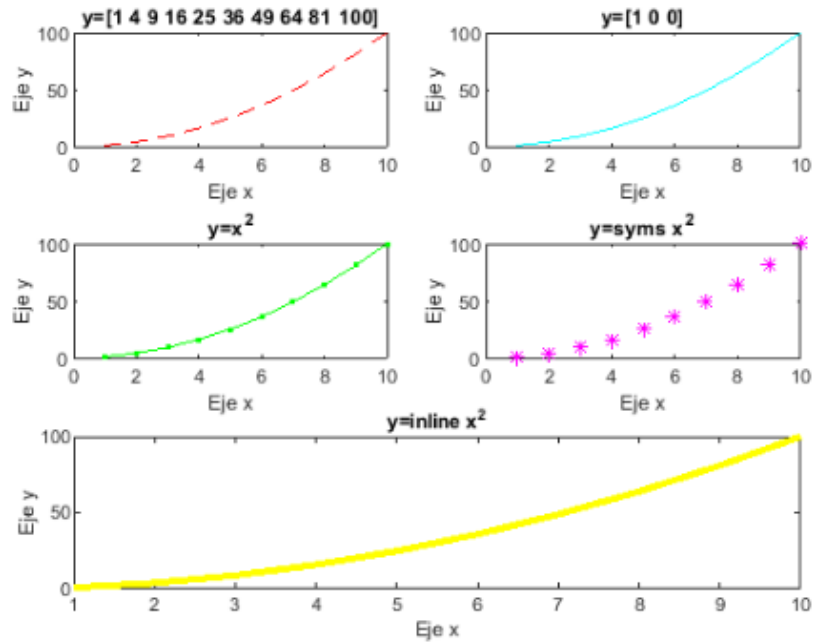
## CÓDIGO EN MATLAB

## RESULTADOS

```
x=[1 2 3 4 5 6 7 8 9 10];
y=[1 4 9 16 25 36 49 64 81 100];
subplot(3,2,1)
plot(x,y,'r--')
title('y=[1 4 9 16 25 36 49 64 81 100]')
xlabel('Eje x')
ylabel('Eje y')
x=[1:10];
p=[1 0 0];
y=polyval(p,x);
subplot(3,2,2)
plot(x,y,'c')
title('y=[1 0 0]')
xlabel('Eje x')
ylabel('Eje y')
x=[1:10];
y=x.^2;
subplot(3,2,3)
plot(x,y,'.-g')
title('y=x^2')
xlabel('Eje x')
ylabel('Eje y')
syms x
y=x.^2
x=[1:1:10];
y=subs(y);
subplot(3,2,4)
plot(x,y,'m*')
title('y=syms x^2')
xlabel('Eje x')
ylabel('Eje y')
x=[1:1:10];
f=inline('x.^2')
y=f(x);
subplot(3,1,3)
plot(x,y,'y','LineWidth',3)
title('y=inline x^2')
xlabel('Eje x')
ylabel('Eje y')
```

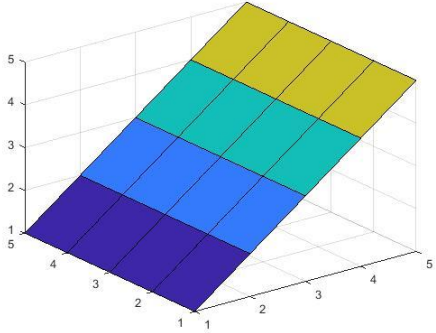
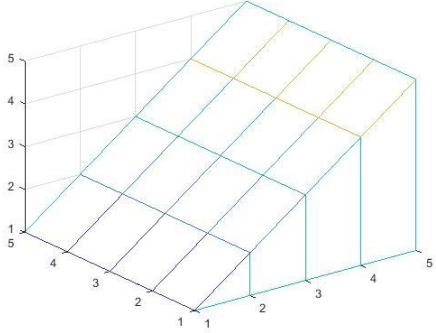
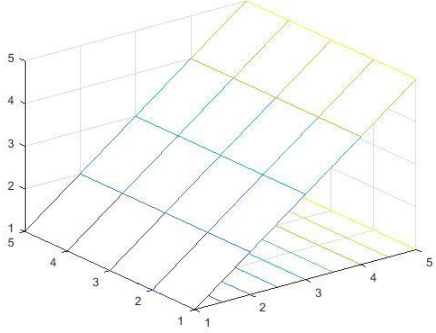
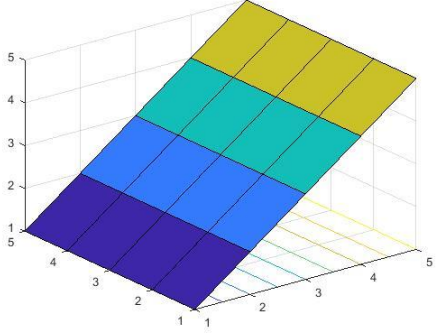
$y = x^2$   
f =

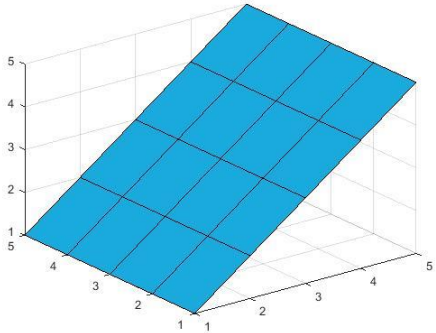
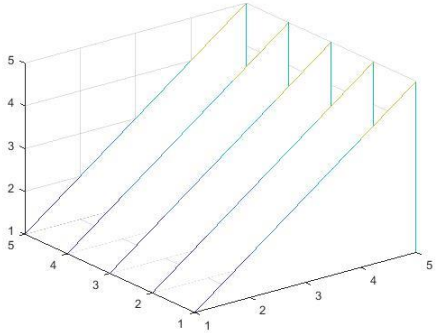
Inline function:  
 $f(x) = x.^2$





## GRÁFICAS 3D

Comando	Función	Ejemplo
<b>surf(x,y,z)</b>	Realiza una gráfica de superficie en 3D	
<b>meshz(x,y,z)</b>	Realiza una gráfica de malla con cortina en 3D	
<b>meshc(x,y,z)</b>	Realiza una gráfica de malla con contorno en 3D	
<b>surfc(x,y,z)</b>	Realiza una gráfica de superficie con contorno en 3D	

<b>surf(x,y,z)</b>	Realiza una gráfica de superficie con alumbrado en 3D	
<b>waterfall(x,y,z)</b>	Realiza una gráfica de cascada en 3D	

Para realizar una gráfica en 3D se necesitan 3 variables, donde z es una matriz por lo que los pasos a seguir son los siguientes:

1. Se utiliza meshgrid para establecer los ejes x,y
2. Se utiliza la ecuación de la función con dos variables, pudiendo utilizar las formas mencionadas anteriormente de ecuación.
3. Graficar usando cualquiera de las función

### Ejemplo:

**x=[1:10]**

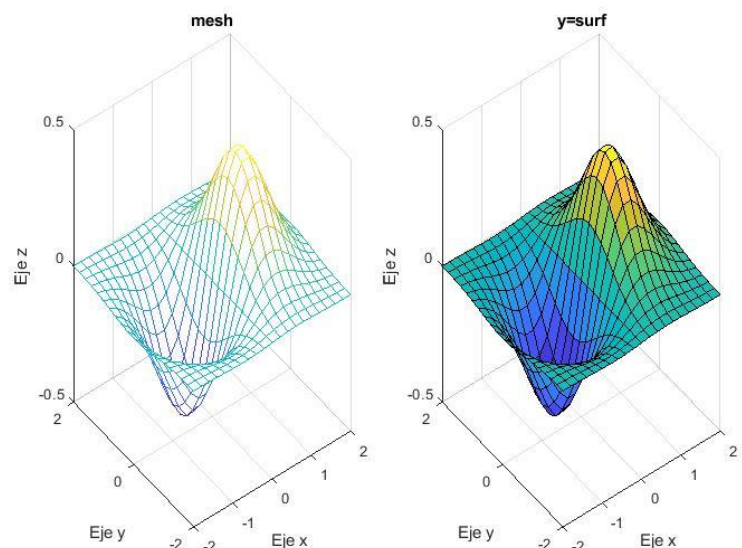
**y=[1:10]**

**z=x $e^{-x^2+y^2}$**

#### CÓDIGO MATLAB

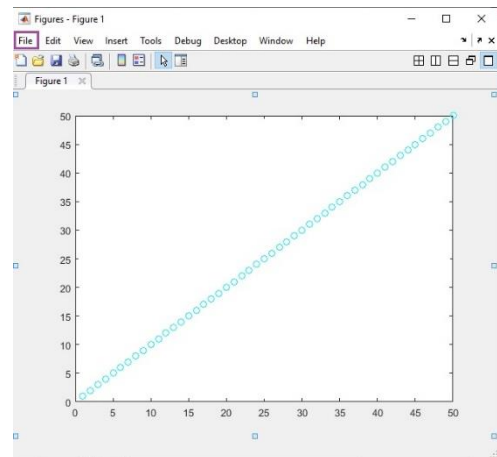
```
[x,y] = meshgrid(-2:.2:2);
z = x .* exp(-x.^2 - y.^2);
subplot(1,2,1)
mesh(x,y,z)
title('mesh')
xlabel('Eje x')
ylabel('Eje y')
zlabel('Eje z')
subplot(1,2,2)
surf(x,y,z)
title('y=surf')
xlabel('Eje x')
ylabel('Eje y')
zlabel('Eje z')
subplot(3,1,3)
```

#### RESULTADOS

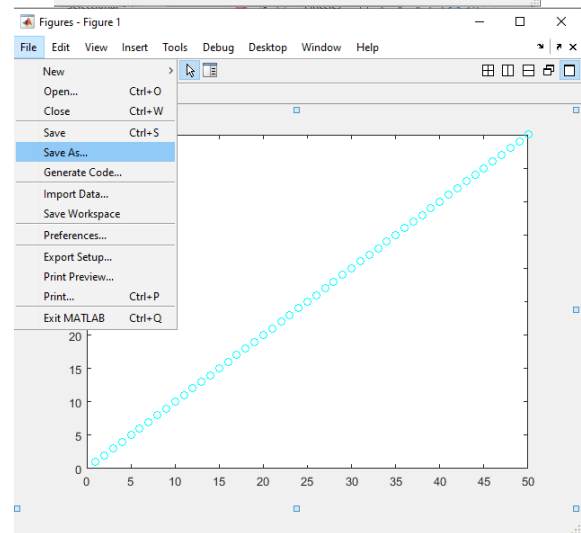


Por último es importante conocer los pasos para guardar una figura como imagen, esto para poder ser utilizada de forma externa de manera más limpia y profesional que utilizando una captura de pantalla. Los pasos a seguir son los siguientes:

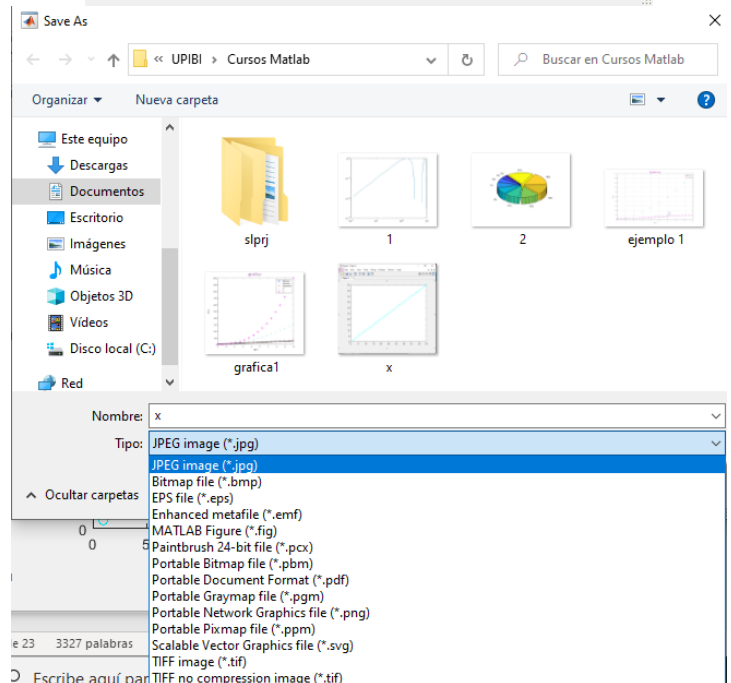
1. Seleccionar menú file en la figura a guardar



2. Seleccionar **Save As**



3. Se abrirá el siguiente menú donde se seleccionara la carpeta en la que se guardará la imagen, seleccionamos tipo JPEG image y se le dará el nombre deseado.



4. Guardar

## VECTORES Y MATRICES

Para crear un vector introducimos los valores deseados separados por espacios (o comas) todo ello entre corchetes [ ]. Si lo que queremos es crear una matriz se realizará de la misma manera únicamente que las filas son separadas con puntos y comas (;).

Generalmente usaremos letras mayúsculas cuando nombremos a las matrices y minúsculas para vectores y escalares. Esto no es imprescindible y Matlab no lo exige, pero resulta útil.

Existen otras maneras de construir vectores matrices, estas son mostrados en las siguientes tablas:

Comando	Descripción
<b>[a:b]</b>	Crea un vector que comienza en el valor a y acaba en el valor b aumentando de 1 en 1.
<b>[a:c:b]</b>	Crea un vector que comienza en el valor a y acaba en el valor b aumentando de c en c.
<b>linspace(a,b,c)</b>	Genera un vector linealmente espaciado entre los valores a y b con c elementos.
<b>linspace(a,b)</b>	Genera un vector linealmente espaciado entre los valores a y b con 100 elementos.
<b>logspace(a,b,c)</b>	Genera un vector logarítmicamente espaciado entre los valores $10^a$ y $10^b$ con c elementos.
<b>logspace(a,b)</b>	Genera un vector logarítmicamente espaciado entre los valores $10^a$ y $10^b$ con 50 elementos.

Ej.

```
>> v=[1 2 3 4 5 6 7 8 9 10]
```

```
v =
```

```
1     2     3     4     5     6     7     8     9    10
```

```
>> v=[1:10]
```

```
v =
```

```
1     2     3     4     5     6     7     8     9    10
```

```
>> v=[1:0.5:5]
```

```
v =
```

```
1.0000    1.5000    2.0000    2.5000    3.0000    3.5000    4.0000    4.5000    5.0000
```

```
>> v=linspace(1,5,10)
```

```
v =
```

```
1.0000    1.4444    1.8889    2.3333    2.7778    3.2222    3.6667    4.1111    4.5556    5.0000
```

```
>> v=logspace(1,5,10)
```

```
v =
```

```
1.0e+05 *
```

```
0.0001    0.0003    0.0008    0.0022    0.0060    0.0167    0.0464    0.1292    0.3594    1.0000
```

Comando	Definición
<b>zeros (n) / (m,n)</b>	Crea una matriz de $n \times n$ o $m \times n$ de ceros.
<b>ones (n) / (m,n)</b>	Crea una matriz de $n \times n$ o $m \times n$ de unos.
<b>rand (n) / (m,n)</b>	Crea una matriz de $n \times n$ o $m \times n$ de números aleatorios con distribución uniforme.
<b>randn(n) / (m,n)</b>	Crea una matriz de $n \times n$ o $m \times n$ de números aleatorios con distribución uniforme.
<b>eye (n) / (m,n)</b>	Crea una matriz identidad (unos en la diagonal y ceros el resto) de $n \times n$ o $m \times n$ .
<b>magic(n)</b>	Crea una matriz cuadrada de $n \times n$ de enteros de modo que sumen lo mismo las filas y las columnas

Ej.

```
>> M=zeros(3)

M =

     0     0     0
     0     0     0
     0     0     0

>> M=ones(3,2)

M =

     1     1
     1     1
     1     1

>> M=rand(5)

M =

    0.8147    0.0975    0.1576    0.1419    0.6557
    0.9058    0.2785    0.9706    0.4218    0.0357
    0.1270    0.5469    0.9572    0.9157    0.8491
    0.9134    0.9575    0.4854    0.7922    0.9340
    0.6324    0.9649    0.8003    0.9595    0.6787

>> M=eye(3,4)

M =

     1     0     0     0
     0     1     0     0
     0     0     1     0

>> M=magic(4)

M =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

Teniendo un vector establecido podemos acceder a los elementos individuales o un conjunto de ellos utilizando subíndices, así como es mostrado en la siguiente tabla:

Comando	Función
<b>v(n)</b>	Devuelve el n elemento del vector v
<b>v(end)</b>	Devuelve el último elemento del vector v
<b>v(n:m)</b>	Devuelve los elementos n a m del vector v
<b>v(n:a:m)</b>	Devuelve los elementos n a m del vector v con un intervalo a
<b>v([a b c])</b>	Devuelve los elementos a b y c del vector v

Ej.

```
>> v=[1:10];
>> v(5)

ans =

    5

>> v(end)

ans =

   10

>> v(2:6)

ans =

    2    3    4    5    6

>> v(1:2:7)

ans =

    1    3    5    7

>> v([3 5 8])

ans =

    3    5    8
```

Para las matrices es importante considerar que estas cuentan con dos dimensiones, filas y columnas M(f,c), por lo que sus elementos pueden ser obtenidos de la siguiente manera:

Comando	Función
<b>M(f,c)</b>	Devuelve el elemento del vector ubicado en la fila f y la columna c
<b>M(f,:)</b>	Devuelve la fila f completa. Los dos puntos significan que se toman todos los valores de las columnas existentes en la matriz M
<b>M(:,c)</b>	Devuelve la fila c completa. Los dos puntos significan que se toman todos los valores de las filas existentes en la matriz M
<b>M(f,n:m)</b>	Devuelve los elementos n a m de la fila f en la matriz M
<b>M([a b] , c)</b>	Devuelve de las filas a y b de la matriz M, la columna c.
<b>M(end, m:n)</b>	Devuelve de la ultima fila de la matriz M las columnas m a n.

\*En estas opciones es posible combinar los comandos utilizados, intercambiando entre filas y columnas.

Ej.

```
>> M=[1 2 3; 4 5 6; 7 8 9];
```

```
>> M(1,2)
```

```
ans =
```

```
2
```

```
>> M(3,:)
```

```
ans =
```

```
7      8      9
```

```
>> M(:,2)
```

```
ans =
```

```
2
```

```
5
```

```
8
```

```
>> M(2,1:2)
```

```
ans =
```

```
4      5
```

```
>> M([2 3],3)
```

```
ans =
```

```
6
```

```
9
```

```
>> M(end,2)
```

```
ans =
```

```
8
```

## Operaciones básicas con arreglos numéricos (vectores y matrices).

Para realizar operaciones con matrices es importante tomar en cuenta la dimensión de los arreglos a utilizar, ya que las operaciones únicamente pueden ser llevadas a cabo vectores con vectores, matrices con matrices o una combinación de vectores con matrices, siempre y cuando las dimensiones coincidan y estos cumplan con las reglas de operaciones con matrices y vectores. Siempre será necesario definir nuestras matrices o vectores utilizando las formas vistas o manualmente, para así poder utilizar los comandos siguientes de una manera más sencilla.

Los símbolos y la manera de expresar las operaciones se muestran en la siguiente tabla:

Tabla 7. Operaciones con arreglos numéricos

Símbolo	Expresión	Operación
+	$A+B$	Suma de arreglos
-	$A-B$	Resta de arreglos
*	$A*B$	Multiplicación de arreglos
.*	$A.*B$	Multiplicación elemento a elemento de arreglos
/	$A/B$	División de arreglos por la derecha
./	$A./B$	División elemento a elemento de arreglos por la derecha
\	$A\backslash B$	División de arreglos por la izquierda
.\	$A.\backslash B$	División elemento a elemento de arreglos por la izquierda
^	$A^n$	Potenciación (n debe ser un número, no una matriz)
.^	$A.^B$	Potenciación elemento a elemento de arreglos
'	$A'$	Trasposición compleja conjugada
.'	$A.'$	Trasposición de arreglos
<b>cross(x,y)</b>	Producto cruz	Producto vectorial entre los vectores x & y
<b>dot(x,y)</b>	Producto punto	Producto vectorial entre los vectores x & y

Ej.

```
>> A=[1 2 3; 4 5 6];
>> B=[1 2; 3 4; 5 6];
>> C=[7 8 9; 5 4 3];
>> A+B
Matrix dimensions must agree.
```

```
>> A+C

ans =

     8     10     12
     9      9      9
```

```
>> A-C

ans =

    -6    -6    -6
    -1     1     3
```

```
>> A*C
Error using *
    -           -

>> A*B

ans =

    22    28
    49    64

>> A.*B
Matrix dimensions must agree.

>> A.*C

ans =

     7    16    27
    20    20    18
```



## Funciones para el análisis de matrices

Función	Definición
<b>det(M)</b>	Determinante de la matriz M
<b>diag(M)</b>	Extrae la diagonal de M como un vector columna
<b>diag(v)</b>	Crea una matriz diagonal con el vector v sobre la diagonal
<b>eig(M)</b>	Valores propios de la matriz M
<b>inv(M)</b>	Matriz M inversa
<b>length(v)</b>	Dimensión de una matriz o un vector
<b>norm(M)</b>	Norma de M
<b>norm(M,n)</b>	Norma-n de M
<b>normest(M)</b>	Estimación de la norma-2
<b>null(M)</b>	Espacio nulo de M
<b>orth(M)</b>	Ortogonalización de M
<b>pinv(M)</b>	Pseudoinversa de M
<b>poly(M)</b>	Polinomio característico de M
<b>rank(M)</b>	Rango de M
<b>rref(M)</b>	Reducción mediante Gauss de M
<b>size(M)</b>	Dimensiones de una matriz (filas y columnas)
<b>trace(M)</b>	Traza de M
<b>tril(M)</b>	Matriz triangular inferior a partir de la matriz A
<b>triu(M)</b>	Matriz triangular superior a partir de la matriz A

Ej.

```
M =                                >> poly(M)

     1     2     3                ans =
     4     5     6                1.0000 -15.0000 -18.0000 -0.0000
     7     8     9

>> det(M)                        >> rank(M)
ans =                             ans =
    6.6613e-16                    2

>> diag(M)                      >> tril(M)
ans =                             ans =
     1     0     0
     5     5     0
     9     8     9

>> eig(M)
ans =
    16.1168
    -1.1168
    -0.0000
```

## POLINOMIOS

Los polinomios son expresiones matemáticas utilizadas para el modelado de problemas científicos y Matlab proporciona las herramientas necesarias para realizar operaciones estándar sobre polinomios, como encontrar sus raíces, evaluarlo y ajustes. Un polinomio es representado en forma de vector ordenando sus coeficientes descendientemente respecto a sus potencias, respetando su signo y cuando no se cuenta con alguna potencia debe ser puesto un 0 en el lugar correspondiente.

$$\text{Ej.} \quad x^5 - 5x^4 + 3x - x^2 - 10 \quad p = [1 \ -5 \ 0 \ -1 \ 3 \ -10]$$

Comenzaremos aprendiendo a obtener el valor de un polinomio con cualquier valor de x de dos maneras diferentes:

1. Se establece el valor de x deseado y se define el polinomio en forma de ecuación. Al momento de escribir el polinomio automáticamente se va a evaluar con el valor de x establecido y nos regresará el resultado deseado.

```
>> x=2;
>> p=x^5-5*x^4+3*x-x^2-10

p =

    -56
```

2. La segunda manera es utilizando el comando polyval(p,x) donde p es el polinomio en forma de vector y x el valor o valores deseados a evaluar en el polinomio.

```
>> x=2;
>> p=[1 -5 0 -1 3 -10];
>> r=polyval(p,x)

r =

    -56
```

En la siguiente tabla se muestran todos los comandos a utilizar con polinomios y sus funciones y más adelante se explica más a detalle el uso de cada uno de estos.

Comando	Función
<b>roots(p)</b>	Cálculo de las raíces de un polinomio
<b>poly(v)</b>	Construye un polinomio con unas raíces específicas
<b>polyval(p,x)</b>	Evalúa un polinomio
<b>corrcoef(y1,y2)</b>	Coeficiente de correlación lineal entre las variables ajustadas
<b>residue</b>	Desarrollo en fracciones parciales (residuos)
<b>polyfit</b>	Ajuste de un polinomio a unos datos
<b>polyder(p)</b>	Derivada de un polinomio
<b>polyint(p)</b>	Integral de un polinomio
<b>conv(p1,p2)</b>	Multiplicación de polinomios
<b>deconv(p1,p2)</b>	División de polinomios

De un polinomio es posible obtener sus raíces, así como es posible de la raíces obtener el polinomio al que corresponden estas:

```
>> p=[1 -5 0 -1 3 -10];
>> r=roots(p)

r =

    5.0316 + 0.0000i
   -0.8241 + 0.8988i
   -0.8241 - 0.8988i
    0.8084 + 0.8265i
    0.8084 - 0.8265i

>> poly(r)

ans =

    1.0000   -5.0000   -0.0000   -1.0000    3.0000  -10.0000
```

Para las operaciones con polinomios se utilizan + para la suma y - la resta, en estas es necesario que los vectores cuenten con las mismas dimensiones, si no no será posible realizar la operación. La multiplicación y división de polinomios es necesario definirlos y utilizando los comandos *conv* y *deconv* se realizan las operaciones. El comando *deconv* devuelve el cociente y el residuo de la división, por lo que las salidas se definen entre corchetes [c,r].

```
>> p1=[1 2 3 4];
>> p2=[5 6 7];
>> p1+p2
Matrix dimensions must agree.

>> p2=[0 5 6 7];
>> p1+p2

ans =

     1     7     9    11

>> p1-p2

ans =

     1    -3    -3    -3

>> p1=[1 2 3 4];
>> p2=[5 6 7];
>> conv(p1,p2)

ans =

         5         16         34         52         45         28

>> [c,r]=deconv(p1,p2)

c =

    0.2000    0.1600

r =

         0         0    0.6400    2.8800
```

Derivación e integración de poliniomios:

```
>> p=[1 -5 0 -1 3 -10];
>> polyder(p)

ans =

         5        -20         0         -2         3

>> polyint(p)

ans =

    0.1667   -1.0000         0   -0.3333    1.5000  -10.0000
```

## Representación gráfica de un polinomio

Un polinomio puede ser gráfico al ser evaluado con un intervalo de valores x definidos, así como también pueden ser mostradas únicamente sus raíces. Los pasos a seguir son los siguientes:

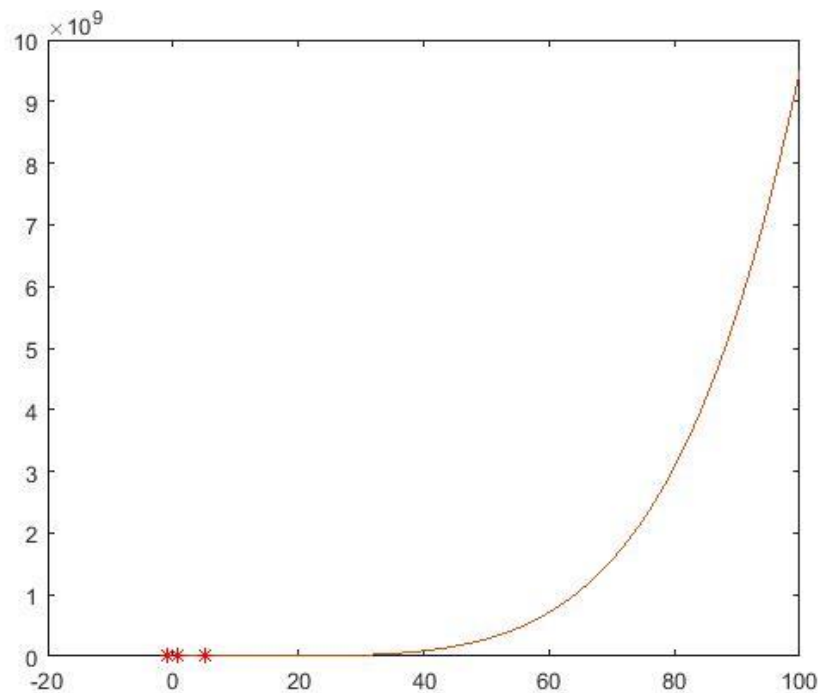
1. Definimos el polinomio p en forma de vector siempre respetando la posición y signos de los coeficientes.
2. Definimos el intervalo x en el que se va a realizar la gráfica
3. Evaluamos el polinomio con `y=polyval(p,x)`
4. Graficamos con `plot(x,y)`
5. Para graficar las raíces es necesario obtenerlas con `r=roots(p)` y las graficamos con `plot(r,0)`

Ej.

### CÓDIGO EN MATLAB

```
>> p=[1 -5 0 -1 3 -10];  
>> x=[1:100];  
>> r=roots(p);  
>> y=polyval(p,x);  
>> plot(x,y)  
>> hold on  
>> plot(r,0,'*r')
```

### RESULTADO



También es posible realizar el ajuste de un polinomio, eligiendo el grado de ajuste que aplique mejor para el caso.

El grado 1 dará una línea recta, el dos un parábola y así sucesivamente. Los pasos a seguir son los mismos que se utilizan para la representación gráfica de polinomios, y para obtener el ajuste se debe seguir lo siguiente:

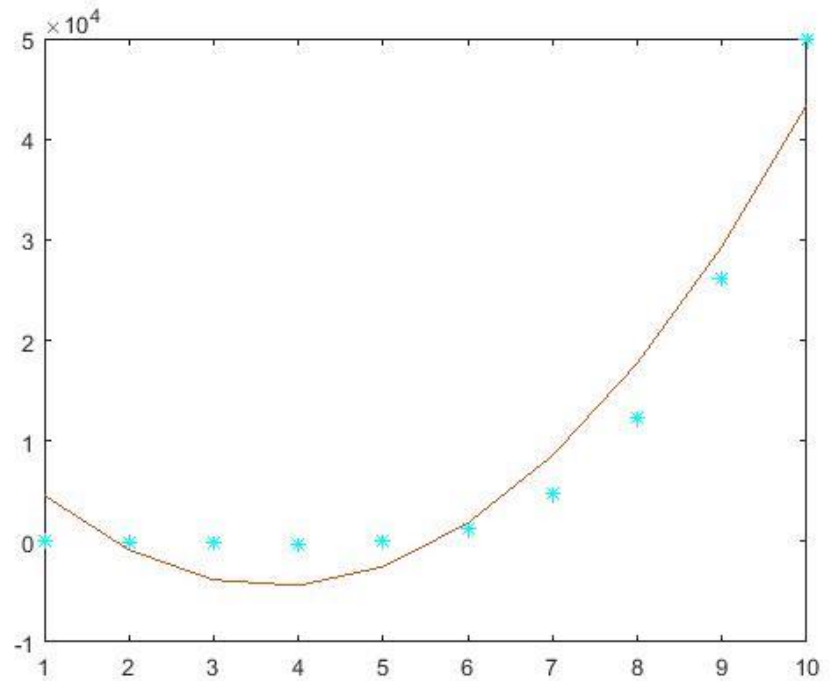
1. Se utiliza el comando `p=polyfit(x,y,n)`, donde x y y son los valores de la gráfica y n será el grado que se desea ajustar
2. Ya obtenido el nuevo polinomio p se realiza la evaluación con el mismo intervalo x con `ya=polyval(p,x)`
3. Se grafica después de usar `hold on` los nuevos datos obtenidos. (Se recomienda que la primera grafica se realice con puntos de dispersión y la gráfica ajustada con una línea continua para que se observe correctamente el ajuste realizado).

Ej.

### CÓDIGO EN MATLAB

```
>> p=[1 -5 0 -1 3 -10];  
x=[1:10];  
y=polyval(p,x);  
plot(x,y,'c*')  
hold on  
pa=polyfit(x,y,2);  
ya=polyval(pa,x);  
plot(x,ya)
```

### RESULTADO



## CICLOS CONDICIONALES

Los ciclos condicionales se tratan de colocar una condición que dependa de un parámetro y mientras que la Condición sea cierta se ejecutará el bloque de Comandos indicado dependiendo la condición del ciclo. Sólo en el caso de que la condición sea falsa o se termine con la condición establecida, el flujo salta el bloque de comandos definidos dentro del ciclo y sigue con el resto del programa. Las condiciones establecidas dentro de un ciclo condicional son terminadas con el comando *end*, que indica que después de este, las líneas de comando ya no forman parte del ciclo.

Para generar estas condiciones dentro de los ciclos condicionales se utilizan los siguientes operadores:

	Operador	Definición
Relacionales	<	Menor que
	<=	Menor o igual que
	>	Mayor que
	>=	Mayor o igual que
	==	Igual a
	~=	Distinto de
Lógicos	&&	Y
		O
	~	No

## While

El comando *while* genera un ciclo en el que mientras se cumpla la condición definida, todos los comandos establecidos dentro de este se repetirán, en el momento que deje de cumplirse el programa terminará o continuará con las siguientes líneas de código.

### **Ej. Mientras x valga=1 pedir x nuevamente**

```
>> x=1;
>> while x==1
x=input('Ingrese un número: ')
end
Ingrese un número: 1

x =

     1

Ingrese un número: 1

x =

     1

Ingrese un número: 2

x =

     2

>> |
```

## For

El comando *for* genera un ciclo establecido por una variable en un intervalo de a:b definidos por el usuario. Es decir que lo que se encuentre dentro de este ciclo se repetirá n veces que se encuentren dentro de este intervalo.

### **Ej. Realizar la suma de 1 a 10**

<pre>&gt;&gt; s=0; for i=1:10 s=s+i end</pre>	<pre>s =      6</pre>	<pre>s =     21</pre>	<pre>s =</pre>
<pre>s =      1</pre>	<pre>s =     10</pre>	<pre>s =     28</pre>	<pre>s =     45</pre>
<pre>s =      3</pre>	<pre>s =     15</pre>	<pre>s =     36</pre>	<pre>s =     55</pre>
			<pre>&gt;&gt;  </pre>

## If

El comando if define una condición en la que si se cumple el argumento se realizará lo establecido dentro de esta. Para establecer diferentes condiciones se utiliza else y else if.

- Else indica que se cumplirá lo definido mientras todo lo no establecido dentro del argumento inicial if se cumpla.
- Con else if es posible determinar distintas condiciones para que se cumplan diferentes opciones dentro de estas.

Ej. Si x=1 y=10, si x=2, y=20 y si x= cualquier otro valor, y=0.

```
>> x=input('Ingrese un número: ')
if x==1
    y=10
elseif x==2
    y=20
else
    y=30
end
Ingrese un número: 1
x =
     1
y =
    10

>> x=input('Ingrese un número: ')
if x==1
    y=10
elseif x==2
    y=20
else
    y=30
end
Ingrese un número: 2
x =
     2
y =
    20

>> |
```

```
>> x=input('Ingrese un número: ')
if x==1
    y=10
elseif x==2
    y=20
else
    y=30
end
Ingrese un número: 3
x =
     3
y =
    30

>> |
```

# FUNCIONES

Las funciones permiten definir funciones análogas a las de Matlab, con su nombre, argumentos y valores de salida. La primera línea que no sea comentario debe empezar por la palabra function, seguida por los valores de salida (entre corchetes [ ] y separados por comas si hay más de uno), el signo igual (=) y el nombre de la función seguido de los argumentos (entre paréntesis ( ) y separados por comas):

**function [a,b,c] = nombre\_función (x,y,z)**

En las líneas siguientes escribimos los argumentos de salida a partir de los de entrada. El nombre de la función y el nombre del archivo deben ser idénticos y no empezar por cifra sino por letra. Todas las variables dentro de una función se aíslan del espacio de trabajo de Matlab. Las únicas conexiones entre las variables dentro de una función y el espacio de trabajo de Matlab son las variables de entrada y salida.

## **Ej. Realice una función para encontrar la raíz cuarta de un número**

Para realizar una función es necesario seguir los siguientes pasos:

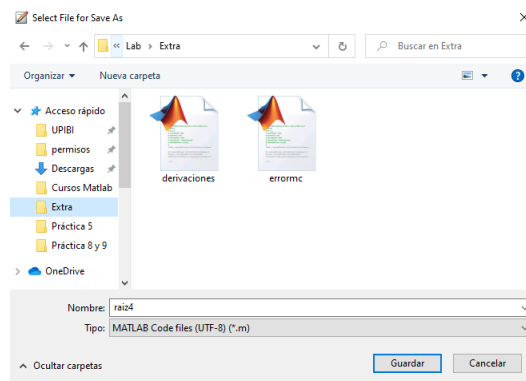
1. Definir cuáles son las entradas, salidas y el nombre de la función

**function [a,b,c] = nombre\_función (x,y,z)**

2. Establecer las líneas de código que serán ejecutadas por la función

```
function [r]=raiz4(n)
r=n^(1/4)
```

3. Guardar la función con el mismo nombre que se le estableció



4. Para correr la función se escribe en el command window el nombre y entre paréntesis la(s) entrada(s) requeridas. \*Es necesario establecer la misma carpeta en la que se guardó en la barra de direcciones de Matlab, si no se realiza este paso no será posible utilizar la función.

```
>> raiz4(16)

ans =

     2

>> |
```



## EJERCICIOS

1. Calcula el resultado de las siguientes operaciones, mostrándose en diferentes variables:

$$1+5/6$$

$$5*3*2/3$$

$$5^2+7^3$$

$$(1+4)/(5-9)+3^3*(1+4)$$

$$9+4+2+1+3+7$$

2. Limpia las variables y la pantalla

3. Define las variables x=12, y=56. Realiza su suma, resta, multiplicación y división y muéstralo en variables s, r, m y d.

4. Realiza las siguientes operaciones:

$$s+r-m$$

$$r+d*s$$

$$s/m+d/r$$

$$s*r+s/m-d$$

Muestre los resultados en formato long, short y hexadecimal

5. Calcula el coseno de  $\pi$

6. Calcula la raíz cuadrada, la raíz cuarta y la raíz quinta de 64

7. Establece las siguientes variables y realiza las siguientes operaciones:

$$a=12.65 \quad b=3\pi \quad c=a+b$$

$$x1 = \frac{a - \sqrt{b^2 - a}}{\cos\left(\frac{a}{c}\right)}$$

$$x2 = \sqrt{\frac{b \times c}{\sqrt{\left(\frac{a}{c}\right) + b \times a}}} - \frac{a}{b}$$

$$x3 = \frac{\cos(a) - \sin(c)}{\tan(b)}$$

$$x4 = \frac{\ln\left(\frac{a+b}{\sin(c)}\right)}{|a-b-c|}(\sqrt{4ab})$$

$$x5 = \left(\sqrt{4a - \frac{b}{c}}\right)(\cos(a) - \sin(b))$$

8. Un proyectil se dispara con un ángulo  $\theta$  y una velocidad inicial  $v_0$ . Desarrolla un programa que solicite al usuario, Ángulo de disparo y velocidad inicial (m/s<sup>2</sup>) para obtener el alcance horizontal, la altura máxima y el tiempo de vuelo del proyectil, desde que sale hasta que impacta con el suelo. Definir  $v=600\text{m/s}$  y  $\theta=60$

$$x_{\max} = \frac{v_0^2}{g} \sin(2\theta)$$

$$y_{\max} = \frac{v_0^2}{2g} \sin^2 \theta$$

$$t_{\text{vuelo}} = 2 \frac{v_0}{g} \sin \theta$$

9. Realizar las siguientes gráficas:

$$f(x) = 2 \sin^3(x) \cos^2(x) \quad y \quad g(x) = e^x - 2x - 3, \quad x \in [-1.5, 1.5]$$

$$f(x) = \log(x+1) - x \quad y \quad g(x) = 2 - 5x, \quad x \in [0, 5]$$

$$f(x) = 6 \sin(x) \quad y \quad g(x) = 6x - x^3, \quad x \in [-\pi/2, \pi/2]$$

$$f(x) = e^{-x^2+2} \sin(x/2) \quad y \quad g(x) = -x^3 + 2x + 2, \quad x \in [-1, 2]$$

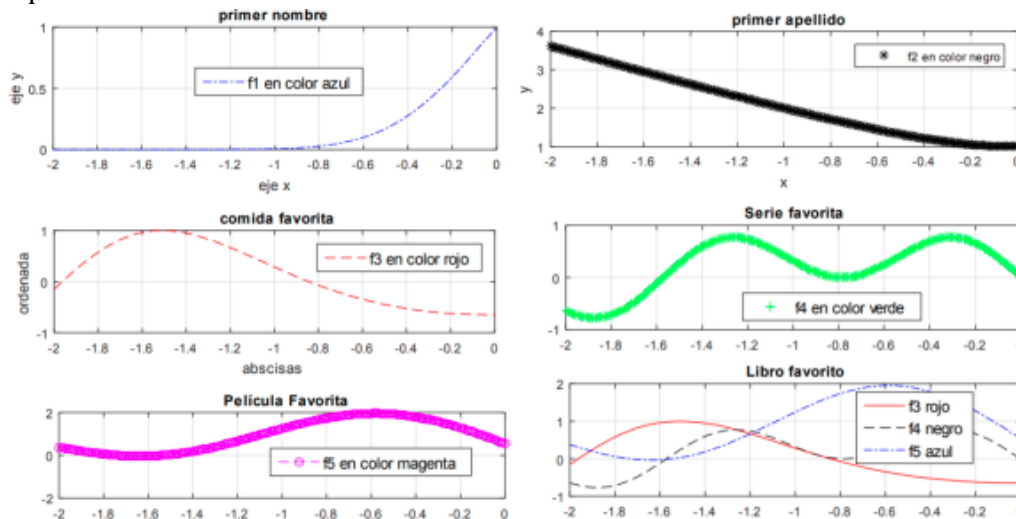
$$f(x) = \sqrt{r^2 - x^2}, \quad \text{para } r = 1 \text{ y } r = 4$$

10. Sean las funciones:

$$f_1(x) = e^{-3x^2+2x} \quad f_2(x) = \sqrt{x^2 - 2x + 1} \quad f_3(x) = \cos(x^2 + 4) \quad f_4(x) = (\cos(2x))(\sin(-4x))$$

$$f_5(x) = e^{-0.1x} \cos(x+1) + \sin(3x+\pi)$$

Construya una figura usando los comandos de graficas de Matlab de manera que se muestre una gráfica por función y una última gráfica con las siguientes características de distribución, colores, y leyendas. Personalice los títulos de acuerdo con sus datos personales como se indica en cada caso. Considere  $x = [-2\pi:0.01:2\pi]$ . Obtenga f1 y f2 estableciendolas como función, f3 y f4 estableciendolas con variables simbólicas y f5 y f6 utilizando linspace.



11. Grafica en 3D de superficie para los siguientes valores:

$$z = \frac{\sin \sqrt{x^2 + y^2}}{\sqrt{x^2 + y^2 + 0.1}}$$

Malla de -10:0.5:10. Cada tipo de gráfica 3D ocupa una ventana.

1	2	3
4	5	6
7		

12. Crea el vector  $v = (1,2,3,4)$
13. Crea el vector  $w = (5,6,7,8)$
14. Calcula el vector traspuesto de  $v$ :
15. Crea un vector llamado  $v2$  donde sus elementos vayan desde el 2 al 17 creciendo de 3 en 3
16. Crea un vector  $v3$  donde sus elementos vayan desde el 2 al 20 y que en total tenga 10 elementos
17. Averigua cuál es el cuarto elemento del vector  $v3$
18. Sean dos vectores  $v$ ,  $w$  y  $k = 3$ . Calcular:

$$v + w$$

$$wk$$

$$w-v$$

$$kv$$

$$v/k$$

$$2vw$$

$$v*w$$

19. Genera un vector  $v$  de 1 a 100 con espaciado de 3 en 3
20. Genera 3 vectores del mismo tamaño del vector original  $v$ .
21. Obtén los datos de 20 al final del vector  $v$ .
22. Crea la matriz  $M=[1\ 2\ 3\ 4;5\ 6\ 7\ 8;9\ 10\ 11\ 12]$
23. Calcula la traspuesta de la matriz  $M$
24. Halla la fila 2 de la matriz  $M$
25. Calcula el rango de  $M$ :
26. Crea la matriz identidad de tamaño 4, 6 y 8.
27. Crea la matriz de ceros de tamaño  $3 \times 3$
28. Crea la matriz cuadrada de unos de tamaño  $5 \times 5$
29. Calcula el determinante de  $M$
30. Definir las siguientes matrices:

$$A = \begin{pmatrix} 2 & 6 \\ 3 & 9 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad C = \begin{pmatrix} -5 & 5 \\ 5 & 3 \end{pmatrix}$$

Realizar las siguientes operaciones:

$$A+B$$

$$A-C+B$$

$$A*B$$

$$C*A$$

$$A*B+B*A$$

Crear una matriz de ceros de  $6 \times 6$  y generar la siguiente matriz (que tiene sobre la diagonal las matrices  $A$ ,  $B$ ,  $C$ ) sin introducir elemento a elemento:

$$G = \begin{pmatrix} 2 & 6 & 0 & 0 & 0 & 0 \\ 3 & 9 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 3 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & -5 & 5 \\ 0 & 0 & 0 & 0 & 5 & 3 \end{pmatrix}$$

Realizar sobre G las siguientes operaciones, guardando todos los resultados en variables distintas:

Borrar la última fila y la última columna de G.

Extraer la primera submatriz  $4 \times 4$  de G.

Extraer la submatriz  $\{1, 3, 6\} \times \{2, 5\}$  de G.

Reemplazar  $G(5, 5)$  por 4.

31. De las siguientes matrices:

$$A = \begin{bmatrix} 1 & 3 \\ -3 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 5 & -1 & 6 \\ 7 & 7 & 9 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} -1 & 3 & 7 \\ -7 & 5 & -2 \end{bmatrix}$$

Obtenga las siguientes operaciones y si no es posible explique por qué.

A + B

AB

BC

B + D

32. Genere los siguientes polinomios y defina que grado corresponde a cada uno:

$$5x^2+10$$

$$7x^8+4x^5-x^6+2x^3+6x^2-1+3x$$

$$3x^3+x^4-8x^5+x^2+7x^9-x-10x^8-13$$

$$6x^{20}+3x^{17}-5x^{10}-7+9x^9+7x$$

$$4x^5+3+5x^4+6x-12x^3-9x^2-18x$$

33. Evalúe los polinomios anteriores con:

$$x=[1:100]$$

$$x2=[-20:2:20]$$

$$x3=[-2\pi:0.01:2\pi]$$

$$x4=2$$

$$x5=19$$

34. Obtenga las raíces de los siguientes polinomios y grafique.

$$x^2+x-12$$

$$x^3-4x^2+x+6$$

$$x^4-5x^2+4$$

$$x^3+4x^2+3x$$

$$x^3-2x^2-5x+6$$

35. Realice las siguientes operaciones con polinomios:

$$P1=5x^3-4x^2+x+6 \quad P2=x^5-5x^3+4x^2-3 \quad P3=x^3+4x^2+3x \quad P4=x^3-2x^2-5x+6$$

$$P1+P2$$

$$P2-P4$$

$$P1*P4$$

$$P2*P3$$

$$P2*P1$$

$$P1+P2+P3$$

$$P2+P4-P1$$

$$P2/P3$$

$$P3/P4$$

$$P1/P3$$

Integral de P2, P3+P2, P1-P4, P2/P1 Y P2\*P3

Derivada de P1, P2+P3, P3/P2, P4-P1 y P1\*P2

36. Realice las gráficas y el ajuste de 5 polinomios de los resultados de las operaciones anteriores y obtenga su coeficiente de correlación
37. Desarrolla un programa que muestre los números múltiplos de 3 y 7 comprendidos entre 1 a 300, además muestre el promedio de dichos números y cuántos de ellos son pares y cuántos impares. IMPORTANTE: El primer número múltiplo de 3 y 7 es el 21.
38. Desarrolla un programa que permita sumar los primeros  $n$  términos de la serie:
- $$3^n + 6^n + 9^n + 12^n + \dots$$
39. Desarrolla un programa que muestre los primeros números primos comprendidos entre 2 y  $n$ .
40. Generar e imprimir la tabla de multiplicar de un número ingresado por teclado hasta un número  $n$  también ingresado por el usuario.
41. Si un número  $X$  entero ingresado por el usuario es divisible por 5. El programa debe imprimir uno de los dos siguientes mensajes:
- i.-  $X$  es divisible por 5
  - ii.-  $X$  no es divisible por 5.
42. Leer un número hasta que este se encuentre entre los valores 17 y 23. Si un número ingresado no está en ese rango el programa pide el ingreso de otro número.
43. Determinar si un número de tipo entero de entrada es par o impar.
44. Determinar el número mayor y el menor de  $n$  números de entrada y entregar la diferencia entre ellos.
45. Construya un algoritmo que sume los primeros números pares ingresados por teclado hasta que su suma sea mayor o igual a 150.
46. Crear un algoritmo que calcule  $ab$ , tal que  $a$  y  $b$  son mayores que 0.
47. Crear un algoritmo que calcule  $12 + 22 + 32 + \dots + Nn$  para un  $N$  dado por el usuario, con  $N > 0$ .
48. Crear un algoritmo que calcule  $1! + 2! + 3! + \dots + N!$ , para un  $N$  dado por el usuario, con  $N > 0$ .
49. Crear un algoritmo que calcule  $(1+1) + (2+(1+2)) + (3+(1+2+3)) + \dots + (N+(1+2+\dots+N))$ , para un  $N$  dado por el usuario, con  $N > 0$ .
50. Crear un algoritmo que calcule  $1N + 2N-1 + 3N-2 + \dots + N1$ , para un  $N$  dado por el usuario, con  $N > 0$ .
51. Elabore un programa que muestre como dado un número entero y positivo (validarlo), cumple con la conjetura de Collatz. Imprima la serie hasta que llegue a 1.
- Conjetura de Collatz:
- “Si el número es par se divide entre dos”
  - “Si el número es impar se multiplica por tres y se le suma 1”
- Tras aplicarle la operación anterior, según corresponda, Repetir el proceso con el número obtenido hasta obtener 1.
- El programa debe preguntar al usuario si desea realizar otro cálculo.
52. Haga un programa determine e imprima el binario de un número natural ingresado por teclado (validarlo). El programa debe preguntar al usuario si desea determinar otro número binario.
53. Escriba un programa que pida al usuario un número y si el que introduce por teclado es menor que 100, que vuelva a solicitarlo. El programa debe imprimir el número de intentos que requirió el usuario para introducir la cantidad correcta.
54. Para cada una de las siguientes series, elabore un programa que calcule su valor, dado un valor de  $n$  válido (validar antes  $n$ ):
- a)  $-1 + 2 - 3 + 4 - 5 + \dots + n$ , tal que  $n$  es par y positivo.
  - b)  $\frac{1!+2!+3!+\dots+n!}{1+2+3+\dots+n}$ , tal que  $n$  es un número entero positivo.
  - c)  $1^{1!} + (2^{1!} + 2^{2!}) + (3^{1!} + 3^{2!} + 3^{3!}) + \dots + (n^{1!} + n^{2!} + n^{3!} + \dots + n^{n!})$
  - d)  $1^n + 2^{n-1} + 3^{n-2} + \dots + n^1$

Los programas deben preguntar al usuario si desea realizar otro cálculo.

55. Escriba un programa que, dado un número natural  $n$  (validarlo), imprima los números primos comprendidos entre 1 y  $n$ .
56. Escriba un programa que imprima los primeros  $n$  números primos solicitados por el usuario.
57. Realiza un programa que pida al usuario un número, (valida número entero y positivo) para calcular su cuadrado usando sólo sumas, muestre en pantalla su correspondiente renglón.

$$\begin{aligned}
 1^2 &= 1 \\
 2^2 &= 1 + 2 + 1 = 4 \\
 3^2 &= 1 + 2 + 3 + 2 + 1 = 9 \\
 4^2 &= 1 + 2 + 3 + 4 + 3 + 2 + 1 = 16 \\
 5^2 &= 1 + 2 + 3 + 4 + 5 + 4 + 3 + 2 + 1 = 25
 \end{aligned}$$

58. Realiza un programa que vaya pidiendo números hasta que se introduzca un número negativo y nos diga cuantos números se han introducido y el promedio de los pares. El número negativo sólo se utiliza para indicar el final de la introducción de datos pero no se incluye en el calculo.
59. Realiza un programa que solicite un número 'n' al usuario (valida positivo y entero), pregunte si desea saber su factorial, o si desea convertirlo a BINARIO.
60. Realiza un programa que muestre los números pares, y cuántos, comprendidos entre dos números introducidos por el usuario y validados como enteros, positivos y distintos entre sí, el programa debe empezar por el menor de los enteros introducidos.
61. Elabore un programa que determine si un año es bisiesto. Un año es bisiesto si es múltiplo de 4 (por ejemplo 1984). Los años múltiplos de 100 no son bisiestos, salvo si ellos son también múltiplos de 400 (2000 es bisiesto, pero; 1800 no lo es).
62. Elabore un programa que determine si una rana que sube por una escalera de la siguiente forma, pisa un escalón dado por el usuario. Verificar que éste sea entero y positivo: "la rana sube cinco escalones en un salto pero desciende dos".
63. Elabore un programa que permita al usuario seleccionar una operación entre dos números: suma, resta multiplicación o división y la realice. En caso de la división, el divisor debe ser diferente de cero de lo contrario envíe un mensaje de error.
64. Programa que determine si dados tres magnitudes determine si corresponden a los lados de un triángulo. De ser así que diga qué tipo de triángulo es.  
Tip: "En un triángulo, cada lado debe ser menor que la suma de los otros dos lados." Probar esto para todas las posibles combinaciones.
65. Elabore un programa que dadas las calificaciones de  $n$  estudiantes por el usuario, calcule el promedio de aprobación (cálculo del promedio tomando en cuenta sólo los que aprobaron), promedio de reprobación (cálculo del promedio tomando en cuenta sólo los que reprobaron) y promedio grupal.
66. Calcule la sumatoria

$$\sum_{i=1}^n \frac{(-1)^i}{i!}$$

Hasta un valor de  $n$ , entero y positivo, dado por el usuario.

67. Elaborar un programa que pida al usuario  $n$  números, imprima cuántos son enteros, de éstos cuántos son positivos y múltiplos de 7 y cuántos de los no enteros son negativos

68. Elabore un programa que calcule la suma de los números impares comprendidos entre los números 1 y  $n$ , donde  $n$  es un número natural (entero positivo) dado por el usuario.
69. Un número natural  $nn$  se denomina perfecto cuando es igual a la suma de todos sus divisores, excepto él mismo. Realice un código que establezca si un número es perfecto.
70. Escriba un programa que lea un número de tres cifras y forme el mayor número posible con las cifras del número ingresado. El número ingresado debe tener el mismo signo que el ingresado
71. Hacer una función en Matlab que determine si en un vector hay elementos repetidos. La función deberá tener como único argumento el vector a examinar y debe devolver un numero 1 si existen elementos repetidos y un 0 si no existen elementos repetidos.
72. Elabore una función en Matlab que se llame "Matriz3p", tal que dada una matriz M como único argumento, entregue 3 variables de salida [x,y,v], donde x es la suma de todos los elementos de la matriz, y es la multiplicación de todos los elementos de la primera columna, y v un vector con todos los elementos de la matriz organizados por renglones uno a continuación del otro.
73. Elabore una FUNCION en MATLAB llamada interseccion, que teniendo como únicas entradas dos conjuntos de datos (números reales) definidos por dos arreglos lineales A y B; construya y despliegue utilizando ciclos repetitivos fijos la intersección  $A \cap B$  de los dos conjuntos de datos.
74. En la bitácora de un laboratorio se toman las mediciones de pH los días lunes, miércoles y viernes, durante 4 semanas. Los datos se anotan de manera consecutiva en forma de lista por los supervisores. Elabore un PROGRAMA en MATLAB que realice lo siguiente:  
 Pida la lista de datos de pH de las cuatro semanas Y LOS GUARDE EN UN VECTOR llamado v, VALIDANDO que cada dato sea un valor posible de pH (0-14).  
 Una vez guardado el vector de datos y utilizando las estructuras de CICLOS, el programa debe reorganizar automáticamente los datos en un arreglo de 4 renglones por 3 columnas, llenándola POR RENGLONES donde cada renglón corresponde a cada semana, y las columnas a los días de medición (Lun, Mie o Vie).  
 Calcular los promedios semanales de las medidas de pH y almacenarlas en un vector llamado promph y calcular el promedio general de los datos y guardarlo en la variable m. No se puede utilizar el comando mean o sum, debe hacerlo utilizando ciclos for o while.  
 Calcular la desviación estándar  $\sigma$  de todos los datos de pH, utilizando la expresión:

$$\sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (v(i) - m)^2} \quad m = \frac{1}{n} \sum_{i=1}^n v(i)$$

75. Elabore un programa que dada una matriz mxn entregue:  
 Vector con los números mayores a 8  
 Vector con múltiplos de 5  
 Vector llamado operaciones con el valor máximo de la matriz, el mínimo, la suma del primer vector, la resta del segundo vector y la suma de los valores menores a 8  
 El vector operaciones ordenado de mayor a menor  
 El vector operaciones ordenado de menor a mayor