Guía de Actividades Nº 1

1. Busque las siguientes palabras con la opción Help <palabra> y con lookfor <palabra>:

   a) format
   b) precedence
   c) Help
   d) Dialog

2. Busque, con la ayuda del soft, los siguientes conceptos:

   a) formato de impresión y visualización de un número.
   b) operadores Booleanos.
   c) operadores aritméticos.
   d) raíz cuadrada de un número real y complejo

3. Realice las siguientes operaciones:

a)	(-2 +3 -5)(4-3+2):(15+4 - 21)	6
b)	$((2-3)(4-3+5))^3((3-2)+(6-8-5))$	1296
c)	$5 - (4 - 3 + 2*7 + 5) + [(3 - 6)^2 (7-9)^2]$	21
d)	$[5 + 3 * 2 : 6- 4][4:2- 3 + 6]:[7- 8 : 2 - 2]^2$	10
e)	$6 - 4 * 3 : 2 - 7 * 2 + 8 - 6 : 3- 5^2 + 3$	-30
f)	$[(2 - 7)(4-5) + 2]^3$	343
g)	$[(2 - 7)(4-5)]^3 + 2^3$	133
h)	$\frac{4^3 - 3 + 2}{4 - 21}$	- 3.7058824
i)	$- 4^2 + (5 + 10)^{1/2} : 2^{1/2} - \frac{10}{25 - 15}$	- 14.261387
j)	(2 - 2i) + 4 + (i - 5)	(1 - i)
k)	i^2	-1

4. Exprese el apartado h) e i) del ejercicio anterior en forma racional.

5. Convierta a binario y hexadecimal los siguientes números:

   a) 127
   b) 389
   c) 753
   d) 928
   e) 10078
   f) 180001

6. Convierta a decimal los siguientes números binarios:

   a) 1001
   b) 111001
   c) 1000001
   d) 11111000011

7. Defina z1= 3 + 2i y z2= -3+7i. Realice las siguientes operaciones:

   a) z1+z2
   b) z1. z2
   c) z1– z2
   d) z1:z2
   e) |z1| + |z2|
   f) $\overline{z1} + \overline{z2}$

8. Ingrese los siguientes polinomios:

a) $P = 2x^3 - 5x + 10$
b) $Q = x^4 + 3x^2 + 1$
c) $R = 10x^6 + 3x^7 - 4x$
d) $S = -x^6 + x - 4$
e) $T = x^2 - 4$
f) $P1 = x^2 - x$
g) $Q1 = 2z^3 + 4z + 8$
h) $R1 = x^4 - x^2 + 3$

9. Defina los polinomios de variable y raíces indicadas;

a) A variable: x raíces: -1,1
b) B variable: z raíces: $\sqrt{2}$, $\sqrt{3}$
c) C variable: t raíces: 0, m = 3
d) D variable: x raíces: -2,1,4
e) E variable: x raíces: i , – i
f) F variable: x raíces: 1, i, –i
g) G variable: w raíces: .1,1
h) H variable: z raíces: .1,1

10. Realice las siguientes operaciones entre los polinomios definidos en los ejercicios 8 y 9.

a) $2P - \frac{4}{5}Q + 2R$
b) $((Q1)^2 - B)/H$

11. Evalúe los polinomios del ejercicio 9 en los siguientes puntos:

a) A en $x = 1, 3, -1/2$
b) B en $z = -\sqrt{2}, 0, 3/2$
c) C en $t = 0, 1/2, \sqrt{3}$
d) D en $x = \pi, -3, e$
e) E en $x = 1 - 2i, -5, \frac{1}{\sqrt{2}}$
f) F en $x = 2$
g) G en $w = 0, e^2$
h) H en $z = 1, (2 - 3i)^{1/2}$

12. Sea los polinomios $H = 3a^2 - 2a + 7$, $F = 6a^3 - 5a + 2$; $G = 5a^2 + 4a - 3$, se pide:

a) $H + F + G$
b) $H - 2F + G$
c) $H - F - 3$
d) $HF - 3G$
e) $H^2 - FG$
f) F/G

13. Resuelva simbólicamente

a) $\dfrac{1 - x^2}{1 + x}$

b) $\dfrac{2x-5}{5 - 2x} + \dfrac{1}{x+1}$

c) $\dfrac{3x^2 + 2x + 1}{x^2 - 5}$

d) $\dfrac{x^2 + 5x - 6}{x^2 + 6x}$

e) $\dfrac{1}{1 - x^2 + 3x^3} + \dfrac{x}{x+1}$

f) $\dfrac{2x}{1 - x^2} + \dfrac{1}{x+1}$

g) $\dfrac{(x+1)^2}{x^2 - 6x + 9} \div \dfrac{3x + 3}{x - 3}$

h) $\dfrac{x + 1}{1 - x} \cdot \dfrac{2 - x - x^2}{5x}$

14. Encuentre el polinomio cociente y resto de dividir P por Q, si:

a) $P = x^5 - 3x^3 - 2$ $\quad Q = x^4 - 2x$

b) $P = x^{10} - x^5 + 7$ $\quad Q = x^4 - 2x^3 + 4$

c) $P = 5x^5 - x^2 - 3$ $\quad Q = x + 1$

d) $P = x^3 - x^5 + 7$ $\quad Q = x^{10} - 2x^3 + 4$

15. Calcule todas las raíces de los siguientes polinomios, usando la orden adecuada:

a) $x^2 - 6x + 9$

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

c) $2x^7 - 3x^3 - 2x + 10$

d) $x^4 - x^2 + 1$

e) $2x^3 - 5x^2 + 3$

f) $-x^5 + 2x^3 - 5$

g) $x^3 - x^2$

h) $(x^2 - 3)^3(x^2 + x + 1)$

i) $(x + 2x^3 - 5x^4)(x^2 + 2) + (x^6 - x^3)$

16. Descomponer en fracciones racionales simples las siguientes expresiones:

a) $\dfrac{x^2}{x^2 - 1.5x + 0.5}$

b) $\dfrac{x^2 - x + 1}{x^3 - x^2}$

c) $\dfrac{x^2 - 2x - 1}{x^2 - 1}$

d) $\dfrac{4x^2 + 13x - 9}{x^3 + 2x^2 - x}$

17. Ingrese las siguientes matrices:

a) $A = \begin{pmatrix} 1 & -2 & 3 & -4 \end{pmatrix}$

b) $B = \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix}$

c) $C = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

d) $D = \begin{pmatrix} 2 & 5 \\ -2 & 3 \\ 4 & 0 \end{pmatrix}$

e) $E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

f) $F = \begin{pmatrix} 1 & 0 & -5 \\ 1 & 1 & 1 \end{pmatrix}$

g) $G = \begin{pmatrix} 1 & -3 & 4 & 6 \\ 0 & 2 & 7 & 11 \\ -4 & 7 & 8 & 9 \end{pmatrix}$

h) $H = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 3 & -2 & 1 \end{pmatrix}$

i) $I = \begin{pmatrix} 1 & -2 & 8 & 0 & 2 \\ 0 & 3 & 4 & 5 & 7 \\ -3 & 5 & 7 & 3 & 1 \\ 1 & 0 & 1 & 3 & -6 \end{pmatrix}$

18. Utilice los comandos adecuados para generar las siguientes matrices:

a) $A = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$

b) $B = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

c) $C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

d) $D = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$

e) $E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

f) $F = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$

g) G = rand (2,4)

h) H = rand(5,7)

19. Genere los siguientes vectores:

a) (1 3 5 7 9 11)

b) (1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8)

c) (6 10 14 18 22)

d) (-10 -5 0 5 10 15 20)

20. Que obtiene cuando ingresa las siguientes órdenes?

a) x = 4 : 0.2 : 5
b) z = -2: 1 : 5
c) z = -2: 5
d) A = [1:8:20 ; -5: 2:0]
e) B = 1:8:20 , -5: 2:0
f) G = 1:8:20 ; -5: 2:0

21. Sean las matrices:

$$M = \begin{pmatrix} 1 & 2 & 3 & 4 \\ -1 & -2 & -3 & -4 \\ 5 & 6 & 7 & 8 \\ -5 & -6 & -7 & -8 \end{pmatrix} \quad A = \begin{pmatrix} 1 & -2 & 3 \\ 2 & -2 & 3 \\ 3 & -3 & 3 \end{pmatrix} \quad B = \begin{pmatrix} 2 & 3 & 6 \\ 3 & 3 & 6 \\ 6 & 6 & 6 \end{pmatrix} \quad C = \begin{pmatrix} 2 & 4 \\ 0 & 4 \\ 1 & 4 \end{pmatrix}$$

a) Muestre la cuarta fila, segunda columna y el elemento (3,2) de la matriz M
b) Muestre la submatriz que formada por las columnas 2 y 4 de la matriz M.
c) Muestre la submatriz que formada por las filas 2 ,3 y 4 de la matriz M.
d) Muestre la submatriz que se obtiene de M omitiendo la fila 2 y las columnas 1 y 3.
e) Muestre la matriz que se obtiene agregando en M los elementos 0 1 2 3 como fila y luego como columna.
f) Muestre la matriz que se obtiene poniendo la matriz C junto a la matriz A, es decir, AC. Use las órdenes adecuadas para indicar el tamaño de la nueva matriz y la cantidad de elementos.
g) Muestre la matriz que se obtiene poniendo la matriz Identidad de orden 3 junto a la matriz A, es decir, AI. Use las órdenes adecuadas para indicar el tamaño de la nueva matriz y la cantidad de elementos.
h) Muestre la matriz que se obtiene poner la matriz B debajo de la matriz A. Use las órdenes adecuadas para indicar el tamaño de la nueva matriz y la cantidad de elementos.
i) Muestre una matriz diagonal de orden 5 de elementos equiespaciados del 1 al 10.
j) Genere un vector que sea 3 veces el vector de los elementos de la diagonal de la matriz A.
k) Muestre la suma de los elementos de la diagonal de B.
l) Genere la matriz T, obtenida de M al intercambiar la fila 1 por la 4 y la 2 por la 3.

Calcule:

m) M -T
n) AB y BA
o) (AB)C y A(BC)
p) $(AB)^2$ y A^2B^2
q) $(A^3)^4$ y A^{12}
r) $(A+B)^2$ y $A^2 + B^2$

22. Defina una matriz D cuadrada no nula. Pruebe que: $D + D^t$ es una matriz simétrica y que $D - D^t$ es una matriz antisimétrica.

23. Sean las matrices $A=\begin{bmatrix} -3 & 2 & 1 \\ -4 & 0 & 1 \end{bmatrix}$ y $B=\begin{bmatrix} 0 & 0 \\ 6 & -1 \\ 0 & 3 \end{bmatrix}$. Calcule: $(A.B)^t$ y $B^t . A^t$.

24. Sea $A=\begin{bmatrix} 3 & 3 \\ -3 & -3 \end{bmatrix}$. Pruebe que A^2 es la matriz nula.

Guía de Actividades Nº 2

1. Invente dos matrices A y B. Pruebe los siguientes comandos:
   a) A.*B y A*B
   b) A./ B , A/ B y C***inv**(B)
   c) A.^B
   d) A^3

2. Sean A y B cualesquiera dos matrices aleatorias [**rand**] de orden 3. Encuentre triu(A), tril(A), triu(B) y tril(B). Calcule los productos triu(A)triu(B) y tri(A)tril(B). Que propiedad presentan dichos productos?

3. Calcule el determinante de las siguientes matrices, usando el comando **det** y concluya si la matriz admite o no inversa:

   [El determinante es un número que caracteriza a una matriz cuadrada. El mismo dependerá de sus elementos y es único. Si el valor del **determinante es distinto de cero**, la matriz admite inversa por lo que es equivalente a la matriz identidad. **Recuerde:** Toda matriz cuadrada que es equivalente a la matriz identidad es inversible. Caso contrario la matriz es **singular**, o lo que es lo mismo, no admite inversa.]

   i) $A = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}$

   ii) $B = \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}$

   iii) $C = \begin{pmatrix} 1 & 2 & 3 \\ 5 & 4 & -1 \\ 0 & 2 & 8 \end{pmatrix}$

   iv) $D = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 0 & -1 \\ 0 & 2 & 1 \end{pmatrix}$

   v) $E = \begin{pmatrix} 2 & 2 & 3 \\ 0 & 4 & -1 \\ 0 & 0 & 1 \end{pmatrix}$

   vi) $F = \begin{pmatrix} 1 & 2 & 4 & 0 \\ 1 & 6 & 2 & 1 \\ 0 & 2 & 0 & 1 \\ 2 & -3 & 1/2 & 0 \end{pmatrix}$

   vii) $G = \begin{pmatrix} 1 & 2 & 8 & -3 \\ 3 & -4 & 2 & 1 \\ 0 & 2 & 0 & 1 \\ 2 & -3 & -7 & 6 \end{pmatrix}$

   viii) $H = \begin{pmatrix} 1 & -2 & 0 & 0 & 0 \\ 2 & 4 & 2 & -2 & 4 \\ 1 & 0 & -3 & 4 & 5 \\ -1 & 1 & 0 & 0 & -2 \\ 0 & 0 & 1 & -1 & 3 \end{pmatrix}$

4. Utilice las matrices anteriores para probar algunas de las propiedades de los determinantes. Compare los resultados y <u>enuncie la propiedad</u>. [Las barras denotan determinante]

   a. $|F\,G|$ y $|F|\,|G|$
   b. $|A + B|$ y $|A| + |B|$
   c. $|A^T|$ y $|A|$
   d. Defina la matriz C1 obtenida de multiplicar la fila 2 de la matriz C por 3. Compare los determinantes de C y C1.
   e. Defina la matriz C2 obtenida de multiplicar la fila 3 de la matriz C1 por 3. Compare los determinantes de C y C2.
   f. $|3H|$ y $3^5|H|$
   g. $|4D|$ y $4^3|D|$
   h. Defina la matriz H1 obtenida a partir de la matriz H intercambiando las filas 3 y 5. Compare sus determinantes.

i. Defina la matriz H2 obtenida a partir de la matriz H1 intercambiando las filas 2 y 4. Compare |H2| con el determinante de H1 y de H. Puede sacar alguna conclusión?

j. Defina la matriz F1 obtenida a partir de la matriz F reemplazando la fila 2 por la fila 2 menos 3 veces la fila 1. Compare |F1| con el determinante de F. ¿qué puede concluir?

5. Calcule mediante el comando **rref** la matriz reducida de las matrices del ejercicio 3. Cuales de ésas matrices son equivalentes? Cuáles admiten inversa?

6. Utilice las operaciones elementales por filas para encontrar la matriz reducida de las siguientes matrices. Verifique sus resultados usando el comando **rref.**

a) $A = \begin{pmatrix} 1 & 1 \\ 2 & 3 \end{pmatrix}$

b) $B = \begin{pmatrix} -1 & 6 \\ 4 & 2 \end{pmatrix}$

c) $C = \begin{pmatrix} 1 & -1 & 1 \\ 2 & 4 & 3 \\ 5 & 6 & -2 \end{pmatrix}$

d) $D = \begin{pmatrix} 2 & 4 & 8 \\ 3 & 5 & 8 \\ -6 & 0 & 4 \end{pmatrix}$

e) $E = \begin{pmatrix} 2 & -4 & -2 \\ 3 & 1 & 6 \end{pmatrix}$

f) $F = \begin{pmatrix} 2 & -7 \\ 3 & 5 \\ 4 & -3 \end{pmatrix}$

Sugerencia: Antes de comenzar redefina la matriz, por ej. RA = A para poder comparar RA con rref(A)

7. Calcule las inversas de las matrices del ejercicio 1. Usando:

a. reducción de la matriz obtenida de adjuntar a cada una de ellas la matriz identidad del orden que corresponda, usando el comando **rref o rrefmovie.**

b. el comando **inv**

8. Pruebe las siguientes propiedades de matrices inversas, de ser posible, usando los resultados anteriores:

a) $(A*B)^{-1}$; $B^{-1}A^{-1}$ b) $(H^{T})^{-1}$; $(H^{-1})^{T}$ c) $|G^{-1}|$; $1/|G|$

9. Este ejercicio explora la forma de las matrices elementales.
[**Recuerde:** una matriz elemental se obtiene a partir de la identidad por aplicación de una y sólo una de las operaciones elementales por filas.]

Defina la matriz A = round(10*rand(4,4) -5) y las matrices elementales (E) que representan las siguientes operaciones por filas: $R_3 \rightarrow 4R_3$, $R_1 \rightarrow R_1 - 3R_2$, $R_1 \leftrightarrow R_4$.

a) Encuentre, para cada E, el producto E*A y compruebe que se realiza la misma operación sobre A.

b) Encuentre la inversa de cada una de las matrices elementales obtenidas. Es la inversa de una matriz elemental una matriz elemental? Justifique

Para cada uno de los siguientes sistemas, dé la matriz aumentada (matriz de coeficientes y la columna de términos independientes) y use el comando **rref** para encontrar la matriz reducida por fila. Usando el Teorema de Rouchè-Fröbenius justifique que los sistemas tienen <u>solución única.</u> [Use la orden **rank** para comprobar el rango de la matriz]
Use la orden adecuada para asignar al vector **X** la solución del sistema.

Compruebe, por medio de las operaciones entre matrices, que dicho vector es solución.

10. $\begin{cases} x_1 - 2x_2 + 3x_3 = 11 \\ 4x_1 + x_2 - x_3 = 4 \\ 2x_1 - x_2 + 3x_3 = 10 \end{cases}$

11. $\begin{cases} -2x_1 + x_2 + 6x_3 = 18 \\ 5x_1 + 8x_3 = -16 \\ 3x_1 + 2x_2 - 10x_3 = -3 \end{cases}$

12. $\begin{cases} x_1 + x_2 - x_3 = 7 \\ 4x_1 - x_2 + 5x_3 = 4 \\ 2x_1 + 2x_2 - 3x_3 = 0 \end{cases}$

13. $\begin{cases} x_1 - 2x_2 + 3x_3 = 0 \\ 4x_1 + x_2 - x_3 = 0 \\ 2x_1 - x_2 + 3x_3 = 0 \end{cases}$

14. $\begin{cases} x_1 - 2x_2 + x_3 + x_4 = 2 \\ 3x_1 + \quad - 3x_3 - x_4 = -8 \\ 4x_2 - x_3 - x_4 = 1 \\ -x_1 + 6x_2 - 2x_3 = 7 \end{cases}$

15. $\begin{cases} x_1 + 2x_2 + 8x_3 - 3x_4 = 2 \\ 3x_1 - 4x_2 + 2x_3 - x_4 = -1 \\ 2x_2 + x_4 = 1 \\ 2x_1 - 3x_2 - 7x_3 + 6x_4 = 0 \end{cases}$

Para cada uno de los siguientes sistemas, dé la matriz aumentada y use el comando **rref** para encontrar la matriz reducida por fila. Concluya, usando el Teorema de Rouchè-Fröbenius, que ningún sistema tiene solución.

16. $\begin{cases} 3x_1 + 6x_2 - 6x_3 = 9 \\ 2x_1 - 5x_2 + 4x_3 = 6 \\ 5x_1 + 28x_2 - 26x_3 = -8 \end{cases}$

17. $\begin{cases} x_1 + x_2 - x_3 = 7 \\ 4x_1 - x_2 + 5x_3 = 4 \\ 6x_1 + x_2 + 3x_3 = 20 \end{cases}$

18. $\begin{cases} x_1 + 2x_2 - 4x_3 = 4 \\ -2x_1 - 4x_2 + 8x_3 = -9 \end{cases}$

19. Las siguientes matrices son matrices aumentadas de sistemas de ecuaciones. Determine sus soluciones.

i) $\left(\begin{array}{ccc|c} 3 & 5 & 1 & 0 \\ 4 & 2 & -8 & 0 \\ 8 & 3 & -18 & 0 \end{array}\right)$

ii) $\left(\begin{array}{cccc|c} 9 & 27 & 3 & 3 & 12 \\ 9 & 27 & 10 & 1 & 19 \\ 1 & 3 & 5 & 9 & 6 \end{array}\right)$

iii) $\left(\begin{array}{ccccc|c} 1 & 0 & 1 & -2 & 7 & -4 \\ 1 & 4 & 21 & -2 & 2 & 5 \\ 3 & 0 & 3 & -6 & 7 & 2 \end{array}\right)$

iv) $\left(\begin{array}{ccccc|c} 6 & 4 & 7 & 5 & 15 & 9 \\ 8 & 5 & 9 & 10 & 10 & 8 \\ 4 & 5 & 7 & 7 & -1 & 7 \\ 8 & 3 & 7 & 6 & 22 & 8 \\ 3 & 2 & 7/2 & 9 & -12 & -2 \end{array}\right)$

a) Use el comando **rref** para encontrar la reducida por filas.

El resto de éste problema trabaja con papel y lápiz

b) Para cada reducida, escriba las ecuaciones equivalentes.

c) Resuelva cada uno de los sistemas.

Suponga que se desean resolver varios sistemas de ecuaciones en los que las matrices de coeficientes son los mismos pero tienen sus vectores de términos independientes distintos. Los mismos se pueden resolver generando una matriz aumentada en la que se agregan tantas columnas de vectores de términos independientes como sistemas se tengan. Es decir una matriz aumentada en la que se contemplan todos los sistemas involucrados.

20. Resuelva los siguientes sistemas:

a) $\begin{cases} x_1 + x_3 + x_2 = 4 \\ 2x_1 + 3x_2 + 4x_3 = 9 \\ -2x_1 + 3x_3 = -7 \end{cases}$ $\quad \begin{cases} x_1 + x_2 + x_3 = 4 \\ 2x_1 + 3x_2 + 4x_3 = 16 \\ -2x_1 + 3x_3 = 11 \end{cases}$

b) $\begin{cases} 2x_1 + 3x_2 - 4x_3 = 1 \\ x_1 + 2x_2 - 3x_3 = 0 \\ -x_2 + 5x_3 - 11x_1 = -7 \end{cases}$, $\begin{cases} 2x_1 + 3x_2 - 4x_3 = -1 \\ x_1 + 2x_2 - 3x_3 = -1 \\ -x_2 + 5x_3 - 11x_1 = -6 \end{cases}$, $\begin{cases} 2x_1 + 3x_2 - 4x_3 = 1 \\ x_1 + 2x_2 - 3x_3 = 2 \\ -x_2 + 5x_3 - 11x_1 = 0 \end{cases}$

21. Resolver los siguientes sistemas, dependiendo del parámetro real m:

a) $\begin{cases} 3x + 3y - z = 0 \\ -4x - 2y + mz = 0 \\ 3x + 4y + 6z = 0 \end{cases}$ $\quad$ b) $\begin{cases} 3x - 5y - 7z = 1 \\ -4x + y + 6z = 2 \\ x + 4y + mz = 7 \end{cases}$

Dos vectores son **ortogonales** si su producto escalar (o interno o punto) es cero.

22. Determine si los siguientes pares de vectores son ortogonales:

a) $\begin{pmatrix} 2 \\ 3 \end{pmatrix}$; $\begin{pmatrix} 3 \\ -2 \end{pmatrix}$

b) $\begin{pmatrix} 2 \\ 3 \end{pmatrix}$; $\begin{pmatrix} -3 \\ 2 \end{pmatrix}$

c) $\begin{pmatrix} -5 \\ 0 \\ 3 \end{pmatrix}$; $\begin{pmatrix} 3 \\ 10 \\ 5 \end{pmatrix}$

d) $\begin{pmatrix} 1 & 2 & 3 & 4 \end{pmatrix}$, $\begin{pmatrix} -4 & -1 & -3 & 4 \end{pmatrix}$

e) $\begin{pmatrix} 1 & 0 & 3 & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & -1 & 0 & 4 \end{pmatrix}$

f) $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$; $\begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix}$

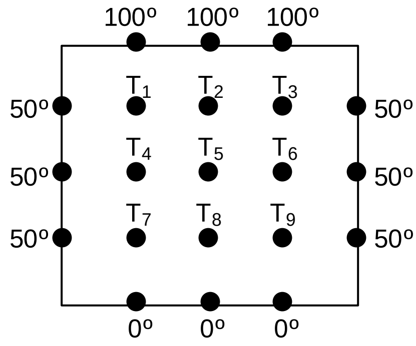
g) $\begin{pmatrix} 1 \\ 4 \\ -7 \end{pmatrix}$; $\begin{pmatrix} 2 \\ 3 \\ 2 \end{pmatrix}$

h) $\begin{pmatrix} 8 \\ -2 \\ 3 \\ -4 \end{pmatrix}$; $\begin{pmatrix} 0.5 \\ -4 \\ 4 \\ 0 \end{pmatrix}$

23. Determine el o los valores de α , si existen, tal que $(1 \quad 2\alpha \quad -1 \quad 5)$ es ortogonal a $(\alpha \quad 3\alpha \quad 7 \quad 1)$.

24. Determine el o los valores de α y β , si existen, tal que $(\beta \quad 2 \quad -1 \quad 3)$ es ortogonal a $(5 \quad \alpha \quad -4 \quad 2)$.

25. **Distribución de calor:** Se tiene una placa rectangular cuyas orillas se mantienen a cierta temperatura. Se tiene interés en encontrar la temperatura en los puntos interiores conocida la temperatura en algunos puntos del borde. Considerar el siguiente diagrama. Se quiere encontrar una aproximación de las temperaturas en los puntos intermedios T1 a T9, suponiendo que la temperatura en un punto interior es el promedio de las temperaturas de los cuatro puntos que lo rodean –arriba, a la derecha, abajo y a la izquierda.



a) Usando las consideraciones anteriores establecer un sistema de ecuaciones que describa las temperaturas, comenzando por T1, después T2, etc. Por ejemplo par T1 se tiene:

$$T_1 = \frac{100 + T_2 + T_4 + 50}{4}$$

Reordene cada una de las ecuaciones. Por ejemplo: $4T_1 - T_2 - T_3 = 150$.

Defina con A la matriz de coeficientes y con b el vector columna de términos independientes.

Construya la matriz ampliada [A b]. Resuelva el sistema con el comando **rref.**

b) Usando la definición de matrices definidas anteriormente, realice A\ b.

c) Resuelva el sistema usando la matriz inversa.

26. Resolver el siguiente **Circuito Eléctrico**:

$$\begin{cases} I_1 - I_2 - I_3 = 0 \\ 5I_1 + 7I_2 = 24 \\ 5I_1 + 3I_3 = 12 \\ -7I_2 + 3I_3 = -12 \end{cases}$$

Ajuste de Polinomios. Si se tienen dos puntos en el plano con coordenadas x distintas, existe una única recta y = ax+b que pasa por ambos puntos. Si se tienen tres puntos en el plano con coordenadas x distintas, existe una única parábola que pasa por ellos. Si se tienen n+1 puntos del plano con coordenadas x distintas, existe un polinomio de grado n único que pasa a través de los n+1 puntos: $y = c_0 + c_1x + c_2x^2 + \ldots + c_nx^n$. Los coeficientes pueden encontrarse resolviendo un sistema de ecuaciones.

27. Encuentre la recta que pasa por (-2, -5) y (4, 7).

28. Encuentre la parábola que pasa por:

a) (-2, 7); (0, 7); (4, 8)

b) (2, 5); (3, 10); (4, -3)

29. Encuentre el polinomio $P(x) = a + bx + cx^2 + dx^3$ que pasa por los puntos:

a) (-1, 2); (0, 4); (1, 10); (2, 26)

b) (-3, 1); (-1, -2); (1, 8); (2, -3)

30. Encuentre P tal que pasa por los siguientes puntos. Indique el grado de los polinomios.

a) (0, 5); (1, -2); (3, 3); (4, -2); (-3, 10).

b) (-1, 2); (0, 4); (1, 10); (2, 26); (-3, -2); (-2, -6)

Guía de Actividades Nº 3: Gráficos en 2D

1. Utilice el comando **plot(x, y**) para los siguientes vectores X e Y:

a) x=[-2 2 6 7] e y = [3 -1 4 6] . Pruebe el comando fill(x,y,'m') y fill(x,y,'g')

b) x=[2 -1 8 7] e y = [10 -1 7 10] . Pruebe también el comando fill(x,y,'b') y fill(x,y,'c')

c) x=[-1 2 3 5; -2 3 7 0] e y=[10 4 7 3; 2 0 6 5]. Analice la gráfica.

d) x=[-1 2 3 5; -2 3 7 0] e y= x.^2. Observe la operación que se usó la operación "**.^**" para elevar cada una de las componentes de x al cuadrado.

2. Genere con la instrucción **plot** un rectángulo de vértices (0,0), (0,3), (5,0), (5,3). Rellene dicho rectángulo de color rojo.

3. Grafique las siguientes funciones en los intervalos indicados, use distintos colores y marcadores de líneas y de puntos:

a) $y = x^5 - 81x$; [-4,4]. Defina primero el vector x como un vector de 20 elementos y luego de 60.

b) $y = [\sin(x)]^2 + 2x\cos(x)$; [-2,6]. Genere primero un vector x de 20 elementos y luego de 70.

c) $y = \dfrac{x}{x^2+1}$ en el intervalo [-4, 4].

d) $y= x.e^{-x^2}$ en [-4,4].

e) $y= e^{-x^2}$ en [-2,2].

f) y= log(x) en el intervalo [-4, 3]. Qué ocurre?

g) y = sinh(x) en [-4,5]

4. Grafique y = 1/x con asteriscos rojos para x = 0:0.1:4. Cambie la opción adecuada y dibuje con una línea continua en azul. Grafique asteriscos para los puntos y únalos con línea continua (use '**r*-**').

5. Grafique en un mismo sistema de coordenadas $y= e^{0.1x}$, y2= ysin(x), y3= y cos(x) con distintos tipos de trazado de líneas y colores para x = 0:0.1:20.

6. Use los comandos **ezplot** y **fplot** para graficar $y=\dfrac{x^3}{(x^2-4)}$ en el intervalo [-8, 9]. Compare los gráficos.

7. Para las siguientes funciones, realice la gráfica incluya título, legenda en los ejes coordenados, grilla y cambie el color de la línea:

a) $f(x)= x(x^2-4)^2$

b) $f(x) = e^{x^2}$

c) $f(x) = \ln(|x -1|)$

d) $f(x) = \dfrac{\sqrt{x}}{x^2+1}$

e) $f(x) = \sin\left(\dfrac{1}{x}\right)$

f) $f(x) = \dfrac{x}{e^{|3x-1|}}$

g) f(x) = tg(sen(x))-sen(tg(x))

h) $f(x) = \begin{cases} x^2 & x < 4 \\ 30e^{-x} & x \geq 4 \end{cases}$

i) $f(x) = \begin{cases} -x & x< 0 \\ 2 & 0 < x < 2 \\ -x^2 & x \geq 2 \end{cases}$

j) $f(x) = \begin{cases} \sqrt{1-x} & x \leq -1 \\ 1- x^2 & -1 < x < 1 \\ \sqrt{1-x} & x \geq 1 \end{cases}$

8. Use el comando **comet** para animar los gráficos, pruebe:

```
t=linspace(0,2*pi,4000);
x=cos(5*t);
y=sin(7*t);
comet(x,y)
```

9. Grafique las siguientes funciones en los intervalos indicados:

a) y = ln(x) [0, 2]

b) y = ln(x -1) [1.1,10]

c) y = log(x) [0, 2]

d) y = -ln(x) [0.1, 4]

10. Grafique los siguientes grupos de funciones en el intervalo [-8, 8]:

a) $\begin{cases} f(x) = |x| \\ f(x) = |x + 2| \\ f(x) = |x - 2| \end{cases}$

b) $\begin{cases} f(x) = |x| \\ f(x) = (1/2)|x| \\ f(x) = 3|x| \end{cases}$

c) $\begin{cases} f(x) = |x| \\ f(x) = -|x + 2| \\ f(x) = - | x | + 2 \end{cases}$

d) $\begin{cases} f(x) = |x| \\ f(x) = |(1/2)x| \\ f(x) = |(1/2)x +1| \end{cases}$

11. Grafique los siguientes grupos de funciones en el intervalo [-2π, 2π]:

a) $\begin{cases} f(x) = sen(x) \\ f(x) = sen(x + \frac{p}{2}) \\ f(x) = sen(x - \frac{p}{2}) \end{cases}$

b) $\begin{cases} f(x) = sen(x) \\ f(x) = sen(2x) \\ f(x) = 2sen(x) \end{cases}$

c) $\begin{cases} f(x) = sen(x) \\ f(x) = sen(0.5x) \\ f(x) = 0.5sen(x) \end{cases}$

d) $\begin{cases} f(x) = sen(x) \\ f(x) = sen(x) + 2 \\ f(x) = sen(x) -2 \end{cases}$

12. Grafique f y g, en los intervalos indicados y en ventanas separadas:

f(t) = cos(-t) +4 en [-2π,0] g(x) = ln(x +1) en [0, 6]

a) Ponga un Nombre a cada ventana.
b) Ponga un Título que indique la función graficada y nombre a los ejes.
c) Cambie el color de fondo, y de la línea.

13. Grafique las funciones f y g del apartado anterior en un mismo sistema de coordenadas, incluyendo las leyendas correspondientes, en el intervalo [0,6].

14. Representar sobre los mismos ejes las gráficas de las funciones sen(x), sen(2x) y sen(3x) en el intervalo [0,2π]. Incluya legenda y títulos. Utilice distintos tipos de línea y de marcadores.

15. Representar sobre los mismos ejes las gráficas de las funciones y= sen(x^2) e y= ln($\sqrt{x}$). Colocar el texto de cada ecuación adecuadamente dentro del gráfico y poner el título al gráfico y nombre a cada uno de los ejes.

16. Presentar en un mismo gráfico las gráficas de seno y coseno en el intervalo [-2π, 2π], colocadas horizontalmente una al lado de la otra con sus respectivos nombres. Mostrar también la representación vertical de ambas (una debajo de la otra) ambas con la retícula.

17. Grafique las funciones **f**, **g**, **h** y **J**, en 4 subventanas como se indica en la figura, colocando en cada una de ellas el título correspondiente.

f(x)= sec(x) + 2, en el intervalo [-4,6]	g(x)= |tagh(x)| en el intervalo [0,6]
h(x)= senh(2x) + 2, en el intervalo [-4,6]	J(x)= - |cosec(x)|, en el intervalo [2,6]

18. Grafique las funciones **f**, **g**, **h** , **J**, **K** y **L** en el intervalo [-2π, 2π]. Utilice 6 subventanas como se indica en la figura, colocando en cada una de ellas el título correspondiente.

f(x)= sen(x)	g(x)= cos(x)	K(x)= tag(x)
h(x)= cosec(x)	J(x)= sec(x)	L(x)= cotg(x)

19. Dibuje **bar**([1 -3 4 5 2 10]

20. Pruebe los siguientes comandos para x=0.5:0.5:4; y= 1./x^2;:

a) bar(x,y)

b) bar(x,y,1.8) % realiza las barras con ancho 1.8

c)barh(x,y)

d) Y=[y' fliplr(y)']; bar(x,Y)

e) bar(x,Y, 'stacked') considere el mismo vector Y definido anteriormente.

21. Pruebe los siguientes comandos (Busque con el help los comandos nuevos):

subplot(3,1,1), bar(rand(10,5),'stacked'), colormap(cool)

subplot(3,1,2), bar(0:.25:1,rand(5),1)

subplot(3,1,3), bar(rand(2,3),.75,'grouped')

22. Realice un histograma con 10 intervalos, en el eje x ponga 'Valor' y en el eje y 'Frecuencia'

x=[1.85 1.86 2.02 2.10 1.96 1.90 1.96 1.97 1.87 1.90 1.95 1.96 1.85 2.13 2.01 2.17 2.06 2.09 1.98 1.89 1.90 1.86 1.96 2.08 2.02 2.13 2.15 2.20 2.09 2.04 2.07 1.87 1.90 1.93 1.86 2.15 2.17 2.01 2.04 2.06 1.89 1.87 1.90 1.97 1.94 1.97 2.00 2.09 2.08 1.87];

23. Realice un histograma, usando el vector anterior, con 8 intervalos, en el eje x ponga 'Valor' y en el eje y 'Frecuencia'.

24. Defina x= -3:0.3:3 y pruebe los siguientes comandos:

a)bar(x, exp(-x.^2))

b)stairs(x, exp(-x.^2))

c)y = randn(10000,1); hist(y,x) % randn es una matriz de números al azar con distribución normal.

d)y=randn(10000,1); hist(y,20) % Utiliza 20 barras para realizar el histograma

e)y=randn(10000,1); hist(y,70) % Utiliza 70 barras para realizar el histograma

25. Pruebe los siguientes comandos, para x=[1 2 1 4 8 12];:

a) pie(x) b) pie(x, [1 0 1 1 0 0]

26. Pruebe los siguientes comandos:

a) pie([2 4 3 5])

b) pie3([2 4 3 5])

c) pie([2 4 3 5], [1 1 0 0])

d) pie3([2 4 3 5], [1 1 0 0])

e) pie([2 4 3 5], [1 1 0 0],{'Norte','Surh','Este','Oeste'})

f) pie3([2 4 3 5], [1 1 0 0],{'Norte','Sur','Este','Oeste'})

Un subconjunto de puntos en $R^{\mathbf{n}}$ se define en forma paramétrica si **es la imagen** de una función vectorial de variable vectorial f: $R^m \to R^{\mathbf{n}}$. En particular, una función vectorial **f** de variable real es una función f: $R \to R^{\mathbf{n}}$, es decir $f(x)= (f_1(x), f_2(x),...,f_n(x))$ donde f_i se llaman funciones coordenadas. Las variables x e y dependen de un parámetro t.

El comando es **plot** con todas sus propiedades, donde previamente se debe definir la variación del parámetro t y no ya de la variable independiente como se estaba realizando.

27. Grafique en forma paramértrica:

a) Una recta que pasa por los puntos (-1, 2) y (3, 7)
b) Una circunferencia de radio 4 con centro en (0,0)
c) Una circunferencia de radio 2 con centro (-1, 2)
d) Una elipse de semieje **x** igual a 3 y de semieje **y** igual a 5, centrada en el origen.
e) Una elipse de semieje **x** igual a 3 y de semieje **y** igual a 5, centrada en (2,-4).

28. Representar la curva (Epicicloide), con el parámetro t variando entre 0 y 2π, y coordenadas paramétricas: x= 5cos(t) – cos(5t), y= 5sen(t) –sen(5t).

29. Realizar la gráfica de la cicloide cuyas ecuaciones son: x= t-2en(t),y= 1-2cos(t), con t entre -3π y 3π.

30. En cada uno de los siguientes ejercicios, x y y están definidos en términos de un parámetro t sobre un intervalo dado. Use los comandos de MATLAB para graficar y vs x. Coloque nombre a cada eje y un título al gráfico.

a) x= 2 cos(t) + sin(t), y = 2sin(t) - sin(2t), $[0,2\pi]$
b) x = sin(3t), y = cos(3t), $[0,2\pi]$
c) x= 2*cos(t)+sin(t), = 2*sin(t)-2*cos(t), $[0,2\pi]$
d) $f(t)=\left(\frac{t(t^2-1)}{t^2+1}, \frac{2(t^2-1)}{t^2+1}\right)$ en [-5,5]

31. Dibujar las siguientes curvas definidas en forma paramétrica por las siguientes funciones f: R $\rightarrow R^2$. Para los apartados a),b),c) dibujar los vectores velocidad (usar el comando quiver):

    a) $f(t) = (2 \cos^3 t;\ 2 \text{ sen}^3 t);\ [-\pi, \pi]$

    b) $f(t) = (3 \text{ sen } t;\ 2 \text{ sen}(2t));\ [-\pi,\pi]$

    c) $f(t) = (\cos(t), \text{sen}(t))\ ;\ [0,\pi]$

    d) $f(t) = \left(\frac{t}{p}\left(12\left(\frac{t}{p}\right)^2 - 9\right)\ ,\ \left(\left(\frac{t}{p}\right)^2 - 1\right)16\left(\frac{t}{p}\right)^2 + 2 \right)$ con $t \in [-3,3]$

    e) $f(t) = (1.5\cos t(\cos t + 1);\ 2 \text{ sen}(2t))\ ,\ [-\pi, \pi]$

    f) $f(t) = (e^{0.25t} \text{ sen}(2t),\ e^{0.25t} \cos(2t))\ ,\ [-\pi, \pi]$

    g) $f(t) = \frac{2}{3}\left(t \cos\left(\frac{7}{2}t\right), t \text{ sen}\left(\frac{7}{t}\right)\right)\ ,\ [-\pi, \pi]$

    h) $f(t) = \left(\frac{2}{3} \cos\left(\frac{7}{2}t\right), \text{sen}\left(\frac{7}{t}\right)\right)\ ,\ [-\pi, \pi]$

32. Graficar las siguientes expresadas en coordenadas polares (usar el comando **polar**), considere en todos los casos la variación de θ en el intervalo $[-\pi, \pi]$

    a) $r = 7 - 7 \text{ sen}(\theta)$

    b) $r = 3 - 6 \text{ sen}(\theta)$

    c) $r = \text{sen}(6\theta)$

    d) $r = \cos(8\theta)$

    e) r= sqrt(4(cos(5θ)))

    f) r= sqrt(4abs(cos(3θ)))

Guía de Actividades Nº 4: Gráficos en 3D

1. Realiza en las gráficas (usando el comando **plot3**) en forma paramétrica de:
   a) la recta que pasa por el punto (-1, 2 -3) y tiene dirección (1, -2, 3)
   b) la recta cuya ecuación es $x = \frac{y+1}{2} = \frac{z-3}{2}$
   c) la recta definida por $\begin{cases} x + 3y - 5z = 4 \\ 2x + y + 2z = -3 \end{cases}$

2. Realiza la gráfica de los siguientes planos. Poner título, nombre a los ejes, grilla.
   a) x + 3y - 2z=1
   b) x - z= 4
   c) x + y + z = 3
   d) 5x + 3y +z = 0
   e) z = 0
   f) pasa por el punto (0, 2, -1) y es paralelo al xy.
   g) pasa por el punto (-3, 4, 5) y tiene dirección normal (-1, 4, 2)

3. Para cada apartado realiza las gráficas de los planos en una misma ventana (use **hold on**). Estudia la posición relativa de los mismos. Resuelve es sistema de ecuaciones y analiza la compatibilidad del mismo.
   a) 2x + 3y - 2z=0 ; 5x –y +4z = 0
   b) x + y – z = -10 ; x + y – z = 0
   c) x + 2y – z = 4 ; x + z = 5
   d) 2x + 3y -4z = 2 ; x + y = 2 ; 3x + 2y + 3z = -1

4. Representar las gráficas de las siguientes funciones de 2 variables, utilizando los comandos surf, mesh y sus variantes. Dibujar también las curvas de nivel.
   a) $z = 1 - x^2 - y^2$
   b) $z = -\sqrt{x^2 + y^2}$
   c) $z = x^4 + y^4 - x^2y^2$
   d) $z = x^4 + y^4 - 3x^2y^2$
   e) $z = \frac{1}{9+x^2+y^2}$
   f) $z = -\sqrt{|x\,y|}$
   g) $z = \frac{\operatorname{sen}\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$

h) $z = \dfrac{\cos\left(\dfrac{x^2 + y^2}{9}\right)}{2 + x^2 + y^2}$

i) $z = \dfrac{xy}{x^2 + y^2}$

j) $z = \dfrac{1 + \operatorname{sen}(xy)}{x^2 + y^2}$

k) $z = \operatorname{sen}(x)\operatorname{sen}(y)$

l) $z = \cos\left(\dfrac{xy}{x^2 + y^2 + 1}\right)$

m) $z = \dfrac{x^3 y - x y^3}{x^2 + y^2}$

n) $z = e^{-(x^2 + y^2)}$

o) $z = x e^{-(x^2 + y^2)}$

p) $z = 0.2y^2 - 4|x|$

5. Dibujar las superficies generadas por **cylinder(R(t),30)**, en cada uno de los siguientes casos:

a) $R(t) = t;$ $t \in [-1; 1]$

b) $R(t) = t^2;$ $t \in [-1; 1]$

c) $R(t) = 2 + \operatorname{sen}(t);$ $t \in [-2\pi; 2\pi]$

d) $R(t) = e^t;$ $t \in [-3; 3]$

e) $R(t) = \sqrt{t}$, $t \in [0; 2]$

6. Pruebe las siguientes órdenes. Cambie efectos, luz, brillo, etc.

a.
```
subplot(1,3,1), [cs,h] = contour(peaks); clabel(cs,h,'labelspacing',72)
subplot(1,3,2), cs = contour(peaks); clabel(cs)
subplot(1,3,3), [cs,h] = contour(peaks);
clabel(cs,h,'fontsize',15,'color','r','rotation',0)
```

b.
```
% Malla y definicion de la funcion
x = -2:0.2:2; y = -2:0.2:2;
[X,Y] = meshgrid(x,y);
f = X.* exp(-X.* X-Y.* Y);
[px,py] = gradient(f,0.2,0.2); % Gradiente de la función
% Visualizacion de la función
subplot(2,2,1); mesh(X,Y,f)
subplot(2,2,2); surfc(X,Y,f)
subplot(2,2,3); pcolor(X,Y,f); axis image; colorbar
subplot(2,2,4); quiver(X,Y,px,py); axis image;
hold on; contour(X,Y,f,20); colorbar
```

OPERADORES y FUNCIONES en MATLAB

Operadores relacionales

Los operadores relacionales ejecutan comparaciones elemento a elemento entre dos matrices y devuelven una matriz del mismo orden cuyos elementos son **T (verdadero)** o **F (falso)** según se cumpla o no la comparación.

OPERADOR	FUNCIÓN
= =	determina si dos expresiones son iguales
a **<** b	determina si a es **menor** que b
a**<=** b	determina si a es **menor o igual** que b
a **>** b	determina si a es **mayor** que b
a**>=** b	determina si a es **mayor o igual** que b
a **~=** b	determina si a **no es igual** a b

A modo de ejemplo, pruebe las siguientes comparaciones entre las matrices **a** y **b**:

```
-->a=[1 1 0 2; -2  3  4  0]
 a  =
!   1.    1.    0.    2. !
! - 2.    3.    4.    0. !
-->b=[1 2 2 3; -2  3 -0  2]
 b  =
!   1.    2.    2.    3. !
! - 2.    3.    0.    2. !
-->a==b
 ans  =
! T F F F !
! T T F F !
```

```
-->a~=b
 ans  =
! F T T T !
! F F T T !
-->a>b
 ans  =
! F F F F !
! F F T F !
-->a<=b
 ans  =
! T T T T !
! T T F T !
```

Cuando se compara un vector o una matriz con un escalar, la comparación se realiza con todos los elementos del vector o de la matriz. Pruebe usando la matriz **a** definida en el caso anterior el comando **a < -1**.

El signo **=** se usa para realizar asignaciones. Esto es `x = 2`, asigna el número `2` a la variable `x`.

Operadores lógicos

Los operadores lógicos sirven para combinar o negar expresiones relacionales. Estos son:

OPERADOR	FUNCIÓN
A & B	Conjunción lógica (and). Es verdadera sólo si ambos **A** y **B** son verdaderos.
A | B	Disyunción lógica (or). Es verdadera si alguno de los dos, **A** o **B**, es verdadero.
~ A	Negación lógica(not). Es verdadera sólo si **A** es falsa.

ARCHIVOS -M

El concepto de función, lo mismo que el concepto de variable, es de fundamental importancia para trabajar con un software matemático. Evidentemente, el concepto teórico de función matemática es fijo e independiente del software con el que trabajemos. Pero la forma de efectuar y operar las funciones es muy exclusiva del paquete con el cual estemos trabajando. MATLAB tiene un modo muy fácil para definir y ejecutar una función.

Hasta el momento estuvimos utilizando funciones definidas por MATLAB como lo son entre otras, **poly**: para ingresar un polinomio, **roots:** para encontrar las raíces de un polinomio, **inv:** para encontrar la inversa de una matriz, etc, para trabajar en el entorno interactivo. Pero, el programa ofrece también la posibilidad de definir funciones a '*medida*'. Entonces, estudiamos ahora cómo podemos aumentar la funcionalidad del lenguaje MATLAB para nuestras aplicaciones definiendo funciones.

Usualmente Matlab es utilizado en modo comando (o interactivo), en cuyo caso se escribe un comando en una línea y se procesa de inmediato. Sin embargo admite también la ejecución de conjuntos de comandos almacenados en un archivo con extensión **.m**. Estos archivos ***.m*** pueden ser de dos tipos:

- ***Scripts*** o **Archivo de comandos**: que simplemente ejecutan una serie de órdenes o instrucciones de MATLAB;

  Trabajan sobre variables en el Workspace de la línea de comandos o crean nuevas variables que son añadidas a dicho Workspace, de modo que todas esas variables pueden ser luego manipuladas desde la línea de comandos. Útiles para automatizar una serie de pasos que se repiten muchas veces.
- ***Funciones*** o **Archivo de funciones**: que además aceptan argumentos y producen resultados.

  Por defecto, las variables internas son locales de la función. Son útiles para extender el lenguaje a otras aplicaciones.

Se crea un M-file utilizando un editor de textos o en su propio editor. Una vez escrita el M-file en el editor, lo grabamos. Lo graba en la carpeta C:\MATLAB\work a no ser que le indiquemos otro directorio. Para ejecutarlo, escribimos directamente su nombre en la línea de comandos de MATLAB como si fuera cualquier otra orden de las que ya conocemos. Esto hace que se ejecuten secuencialmente todos los comandos o sentencias incluidas en el archivo.

Este modo de trabajo es útil cuando se requiere procesar conjuntos muy largos de comandos y principalmente favorece la automatización de procedimientos.

Scripts o Archivo de Comandos

Cuando un archivo de comandos es invocado, MATLAB simplemente ejecuta los comandos encontrados en dicho archivo. Las instrucciones en un **scripts** operan globalmente en los datos en el espacio de trabajo. Los comandos son utilizados para hacer análisis, resolver problemas, ó diseñar secuencias largas de comandos que se conviertan en interactivas. A modo de ejemplo, copie el siguiente grupo de comandos y guarde el archivo como **fibonacci.m**.

```
% Un archivo-M para calcular los elementos de la serie de Fibonacci con 12 términos y graficarla
f = [1 1];
for i=1:10
f(i+2) = f(i) + f(i+1);
end
clc
disp('La serie de FIBONACCI con 12 términos es'),
f
pause(5)
plot(f,'r*')
```

En al ventana de comandos escriba `fibonacci` y pulse `enter`. Observe el resultado. Luego que la ejecución del archivo es completada, las variables f y i permanecen en el espacio de trabajo.

- **Ejemplo 2:**

```
% Demostracion de scripts con matrices
A=[1 2 3; 4 5 6; 7 0 9];
clc
disp(' ')
disp('La matriz A='),
disp(A)
pause(3)
disp('La matriz inversa de A es:'),
format rat
Inversa=inv(A);
disp(' ')
disp(inv(A))
```

Grabe los siguientes comandos en un archivo `nombre.m`. Pruebe las siguientes sentencias poniendo en la ventana de comandos su nombre (el nombre con que Ud. grabó el archivo) y pulse `enter`.

- **Ejemplo 3:**

```
% Demostracion de scripts ingreso de datos
clc
A=input('\n Ingrese la matriz cuadrada encerrada por corchetes A =');
clc
disp(' ')
disp('La matriz ingresada por Ud. es'),
disp(A)
pause(3)
if det(A)==0
disp('La matriz no admite inversa')
else
disp('La matriz inversa de A es:'),
format rat
Inversa=inv(A);
disp(' ')
disp(inv(A))
end
disp('Muchas Gracias')
```

Grabe el archivo como **ingreso.m** . Hágalo correr.

Controle que las variables definidas en el programa persisten.

Archivo de Funciones

Una de las aplicaciones más frecuentes se presenta en la definición de funciones, a través del comando ***function*** cuya sintaxis es:

```
function [res1,res2] = nombre_con_ que_ llamaremos_ a_ la_ función(a, b, c)  % definición de la función
  <cuerpo de la función>
```

Una vez que la función ha sido definida, se guarda en un archivo con el nombre que deseamos asignar a la función con extensión **.m**

res1 y res2: es el resultado1 y resultado2, o directamente los parámetros de salida. Cuando las salidas que produce una función son más de una se las encierra entre corchetes. Es lo que me va a mostrar la ejecución de la función.

(a, b, c): son las entradas de la función, pueden ser números o letras. En éste último caso deben ir encerradas por comillas. Son los parámetros de entrada.

Cuerpo de la función: esta parte contiene las sentencias que realizan los cálculos y asignan valores a los parámetros de salida. Es decir, define los valores que asumirán res1 y res2 teniendo en cuenta los valores ingresados. Cada comando o instrucción del cuerpo de la función suele ir en una línea que finaliza con un " **;** " lo cual permite que no se muestre los resultados de las nuevas variables en pasos intermedios.

Cuando la función tiene un único resultado, el esquema puede ser simplemente:

```
function resultado = nombre_con_ que_ llamaremos_ a_ la_ función(a, b, c)
  <Procedimiento>
```

- **Nota:** Al archivo se <u>le debe asignar</u> el nombre con que definimos la función.

Las variables definidas dentro de una función dejan de existir una vez finalizada la ejecución de la función, como ya dijimos, son **variables locales**.

Si se desea, se puede editar el archivo, hacer algunos cambios, guardarlos y de <u>nuevo activar</u> la función mediante F5 si trabaja con el editor de Matlab.

- **EJEMPLO 4:**

```
function [x1, x2]= raices(a, b, c)
% Calcula las raíces de una ecuación de segundo grado ax2 +bx +c =0
x1= (- b - sqrt(b^2 - 4*a*c))/(2*a);
x2= ( - b + sqrt(b^2 - 4*a*c))/(2*a);
```

Líneas de comentarios comienzan con %

Define los valores de salida en términos de los valores de entrada.

a: es el coeficiente del término cuadrático, **b:** del término lineal y **c:** es el término independiente

Grabamos nuestra función con el nombre por ejemplo: **raices.m** en la carpeta Work. Salimos del Editor. Luego de grabado el archivo con las instrucciones, en la ventana de comandos escribimos:

```
>> raices(1, 0,-4)
 ans =
    -2
```

Calcula las raíces de $x^2 - 4 = 0$

Observe que nos muestra una sola raíz. Esto sucede porque no hemos definido dos salidas para nuestra función. Probamos ahora:

```
-->[a,b]=raices(1,0,-4)
 b  =
  - 2.
 a  =
    2.
```

Podríamos haber definido la función de este otro modo:

```
function raices1(a,b,c)
% Calcula las raíces de una ecuación de segundo grado ax^2 +bx +c =0
d=sqrt(b^2-4*a*c);
x1=(-b+d)/2*a;
x2=(-b-d)/2*a;
disp('raíz 1 =  ')
disp(x1)
disp('raíz 2 =  ')
disp(x2)
```

podemos **no** definir acá la salida, pero debe hacerse al final.

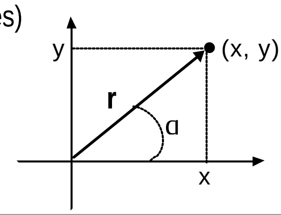
d : es el discriminante.

Define la salida.

```
-->raices1(1,0,-4)
 raíz 2 =
 -2
 raíz 1 =
 2
```

- **EJEMPLO 5:**

Definimos ahora, una función que devuelve las coordenadas rectangulares de un punto a partir del conocimiento del ángulo y del radio (coordenadas polares)



Recuerde que:

$x = r \cos(\theta)$

$y = r \sin(\theta)$

```
function [x,y]= Polar_a_Cart(a,b)
% Devuelve las coordenadas cartesianas de un punto a partir de sus coordenadas polares.
x= a*cos(b);
y= a*sin(b);
```

a: es el radio y **b**: es el ángulo expresado en radianes.

Grabamos el archivo con el nombre **Polar_a_Cart .m**.

```
-->[x,y]=Polar_a_Cart(2,6)
 y  =
  -  .558831
 x  =
    1.9203406
```

```
[x,y]=Polar_a_Cart(2,pi/2)

 y  =
    2.
 x  =
    1.2e-16
```

Observe: 1.2E-16, es un número muy cercano a cero. Podemos cambiar el formato para obtener como coordenada x el 0.

En el mismo archivo podemos definir otra función que considere que la entrada del ángulo es en grado. Para ello:

- **EJEMPLO 6:**

```
function [x,y]= Polar_a_Cartg(a,b)
b=b*pi/180;               <- Convierte el número entrado en radianes
[x,y]= Polar_a_Cart(a,b)  <- Se puede usar una función definida previamente. Debe estar disponible en el momento de ser usada.
```

Veamos:

```
-->[x,y]= Polar_a_Cart(2,90)
 y  =
    1.7879933
 x  =
  -  .8961472
```

```
-->[x,y]= Polar_a_Cartg(2,90)
 y  =
    2.
 x  =
    1.225E-16
```

Note la diferencia entre Polar_a_Cart y Polar_a_Cartg.

Funciones y Control de Flujo

En matemática es frecuente utilizar funciones recursivas, condicionales y definidas por partes. Para definirlas es necesaria la utilización de sentencias propias a este fin y del manejo de bucles. Describimos, a continuación, la sintaxis propias del Scilab para éstas sentencias.

Scilab tiene dos sentencias para bucles, ellas son **FOR** y **WHILE.**

El Bucle FOR

Esta sentencia permite ejecutar de forma repetitiva un comando o grupo de comandos un número determinado de veces. La sintaxis es:

```
for  <variable> = lim1:incr:lim2
comandos
end
```

Siempre comienza con la palabra **for** y termina con la palabra **end**, e incluye una serie de comandos que se separan con comas o punto y comas para evitar repeticiones en las salidas. Estas sentencias pueden estar anidadas, es decir, puede haber un **for** dentro de otro **for** o dentro de un **if**.

La variable <variable> empieza en el límite inferior lim1, después va incrementar el valor en el incremento incr (puede ser negativo) y termina en lim2. Si el incremento es 1, éste se puede suprimir.

- **EJEMPLO 7:**

```
% Define una matriz de orden mxn de elementos a(i,j) = (i*j-1)
function A = matriz(m,n)
for i=1:m,                          nos movemos en la fila
     for j= 1:n,                    fijado un valor para la fila, nos movemos en las columnas
     A(i,j)= (i*j-1);               define el elemento a(i,j)
     end
end
```

Grabamos con el nombre **matriz .m** la carpeta elegida. Luego:

```
-->matriz(2,3)
 ans  =
!  0.   1.   2. !          a(1,1) = 1*1-1 = 0 ,   a(1,2) = 1*2 - 1= 1,   a(1, 3) = 1*3 -1= 2
!  1.   3.   5. !          a(2,1) = 2*1-1 = 1,    a(2,2) = 2*2-1 = 3;    a(2,3) = 2*3-1 = 5
```

El Bucle WHILE

Esta sentencia permite ejecutar un comando o un grupo de comandos un número determinado de veces **mientras se cumpla** una condición lógica especificada. La sintaxis es:

```
while <condición>
comandos
end
```

- **EJEMPLO 8:**

Veamos como utilizar el bucle while, en el siguiente ejemplo:

```
% Calcula el mayor número cuyo suma acumulada no exceda al número m
function n = masum(m)
n=1;
   while cumsum(1:n)<=m   <------   Condición: mientras la suma de 1 hasta n sea menor que m, hace
                                    lo que sigue. [sum(1:m) es una función definida en scilab]
   n=n+1;                 <------   asigna a n el valor n+1 (incrementa en 1 el valor de n).
   end
   n = n – 1              <------   asigna a n el valor n-1. Recuerde: este valor es el resultado que nos
                                    mostrará nuestra función. Observe. La condición no se cumple para n.
```

Grabamos nuestra función con el nombre por ejemplo: **masum.m** en la carpeta elegida. Salimos del Editor. Luego:

```
-->masum(20)
 ans  =
   5.
```

Verifique que la suma de 1 + 2 +3 + 4 + 5 = 20 y que si agregamos el 6, dicha suma supera a 20.

Matlab tiene dos sentencias para procedimientos que incluyen condicionales ellas son: **if-else** y **select-case.**

IF ELSEIF ELSE END

Esta sentencia permite ejecutar un comando o un grupo de comandos un número determinado de veces **si se cumple** una condición lógica especificada. La sintaxis es:

```
if <condición1> then
  comandos1
end
```

La palabra **then** se puede reemplazar por una coma o un cambio de línea. En éste caso sólo se ejecutan los comandos1 si la condición1 es verdadera.

También se puede usar del siguiente modo:

```
if <condición1> then
comandos1
     elseif <condición2>
     comandos2
         else
         comandos3
end
```

En éste caso se ejecutan los comandos1 si la condición1 es cierta, se ejecutan los comandos2 si la condición1 es falsa y la condición 2 es verdadera, se ejecutan los comandos 3 si la condición 1 y la condición 2 son falsas. Puedo poner varios **elseif** anidados.

- **EJEMPLO 9:**

Generemos una función tal que al ingresar un número nos devuelva si ese número ingresado es negativo, par o impar.

```
function numero(n)
if n<0
disp('  es un número negativo');
elseif mod(n,2)==0
disp('es un número par');
else
disp('  es un número impar');
end
```

Veamos:

```
-->numero(4)
             es un número par
-->numero(-3)
              es un número negativo
-->numero(3)
```

es un número impar

SELECT CASE

La sintaxis es:

```
switch  <variable>
  case <valor 1>
      comandos1
  case <valor 2>
        comandos2
      ……
  otherwise
      comandos
end
```

La palabra **then** se puede reemplazar por una coma o un cambio de línea.

- **EJEMPLO 10:**

```
function elija(x)
switch x
case 1
    disp ('Ud eligió el número 1');
case 2
    disp ('Ud eligió el número 2');
case 3
    disp ('Ud eligió el número 3');
otherwise
    disp ('Ud no eligió un número entre 1 y 3');
end;
```

```
-->elija(3)
 Ud eligió el número 3
-->elija(4)
 Ud no eligió un número entre 1 y 3
-->elija(1)
 Ud eligió el número 1
-->elija(2)
 Ud eligió el número 2
```

Otros Ejemplos de funciones

- **EJEMPLO 11:**

Definimos en forma recursiva el factorial, lo podíamos haber definido con la sentencia del Matlab **prod(1:n)**.

El factorial de un número se define como: `0! = 1`, `1! = 1`, `n! = n*(n-1)!`, en general `n!= n(n-1)(n-2)...3.2.1`

```
% Calcula el FACTORIAL de un número positivo
function y=factori(x)
   if x<0
   sprintf('El factorial de %d no existe.',x)
   end
   if x==0
   y=1
   end
   if x==1
   y=1
   end
   if x>1
   y=x*factori(x-1);
   end
```

- **EJEMPLO 12:**

```
% Calcula todos los divisores de un número entero ingresado
function a=fct(n)
a=0
i=0
for j=1:n
      if(mod(n,j)==0)
        i = i + 1;
        a(i)=j
      end
end
```

Veamos otros modos de ingresar datos:

- **EJEMPLO 13:**

```
clc
A= input('\n\n Ingrese dos valores con el formato [a,b]=  ');
a=A(1); b=A(2);
if a==0|b==0
   disp('al menos uno de los valores es cero');
         elseif a <0 & b<0
             disp('Los dos valores son negativos');
         else
             disp('Al menos uno de los valores puede ser positivo')
end
```

Debemos ejecutar el archivo cada vez que lo necesitemos usar o definir alguna condición que vuelva a pedir datos.

```
Ingrese dos valores con el formato [a,b]=-->[0, 2]

 al menos uno de los valores es cero

Ingrese dos valores con el formato [a,b]=-->[-1,-2]

 Los dos valores son negativos

Ingrese dos valores con el formato [a,b]=-->[-1,2]

Al menos uno de los valores puede ser positivo
```

- **EJEMPLO 14:**

```
clc
A= input('\n\n Ingrese una función cualquiera f(x)=  ','s');
x= input('\n\n Ingrese valor de x =   ');
z= eval(A,x)
fprintf('El valor de la función ingresada f(x)= %s en x=%d es f(x)=%d',A,x,z);
```

```
 Ingrese una función cualquiera f(x)=  -->x^2
 Ingrese valor de x-->-1
 El valor de la función ingresada f(x) = x^2 en x=3 es f(x)=9
```

Guía de Actividades Nº 5

- Realizar un programa de tema a elección. El mismo debe reflejar manejo de:
    - Instrucciones de entrada y salida de datos (funciones, intervalos).
    - Validación de datos ingresados.
    - Funciones recursivas, condicionales y definidas por partes.

Se evaluará:

- Presentación
- Capacidad de relacionar e integrar conceptos.
- Uso correcto de comandos específicos de Matlab a fin de optimizar los procedimientos.

INTERFAZ GRÁFICA DE USUARIO EN MATLAB (GUIDE)

MATLAB permite desarrollar de manera simple un conjunto de pantallas con botones, menús, ventanas, etc., que permiten utilizar de manera muy simple programas realizados en el entorno Windows. Este conjunto de herramientas se denomina interface de usuario.

GUIDE (Graphical User Interface Development Enviroment) es un entorno de programación visual disponible en MATLAB para realizar y ejecutar programas. Tiene las características básicas de todos los programas visuales como Visual Basic o Visual C^{++}.

Al usar GUIDE obtendremos dos archivos, un archivo con extensión **fig** que contiene la descripción de la de los componentes de la interfaz y un archivo M (con extensión .m) que contiene las funciones y los controles del GUI, así como las acciones(callback) que llevará a cabo un objeto GUI cuando el usuario lo active.

Iniciamos en la ventana de comandos con la siguiente instruccion:

```
>>guide
```

o bien, desde la barra de menú, siguiendo la secuencia de instrucciones [file] [nuevo][gui]. Esto hace que se abra la ventana GUIDE Quick Start, la cual se presentan las siguientes opciones:

a) Blank GUI (Default)

La opción de interfaz gráfica de usuario en blanco (viene predeterminada), nos presenta una ventana nueva, en el cual podemos diseñar nuestro programa.

b) GUI with Uicontrols

Esta opción presenta un ejemplo en el cual se calcula la masa, dada la densidad y el volumen, en alguno de los dos sistemas de unidades. Podemos ejecutar este ejemplo y obtener resultados.

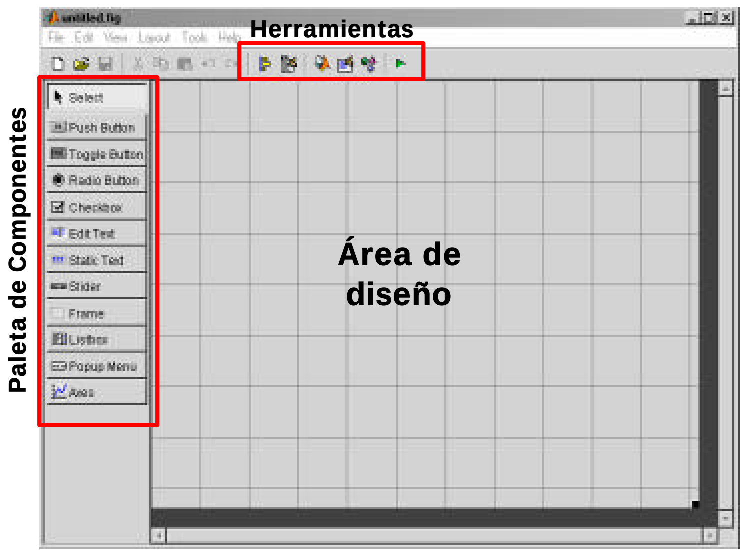
c) GUI with Axes and Menu

Esta opción es otro ejemplo el cual contiene el menú File con las opciones Open, Print y Close. En el formulario tiene un *Popup menu*, un *push button* y un objeto *Axes*, podemos ejecutar el programa eligiendo alguna de las seis opciones que se encuentran en el menú despegable y haciendo click en el botón de comando.

d) Modal Question Dialog

Con esta opción se muestra en la pantalla un cuadro de diálogo común, el cual consta de una pequeña imagen, una etiqueta y dos botones *Yes* y *No*, dependiendo del botón que se presione, el GUI retorna el texto seleccionado (la cadena de caracteres 'Yes' o 'No').

Elegimos la primera opción, *Blank GUI,* la cual no abre la siguiente ventana:



Esta cuenta con las siguientes **herramientas**:

	Alinea Objetos.
	Editor de menú.
	Editor de archivo M
	Propiedades de objetos.
	Navegador de objetos.
	Grabar y ejecutar.

La siguiente tabla muestra una descripción de **La Paleta de Componentes**:

Control	***Valor de estilo***	***Descripción***
Select	'select'	Sirve para seleccionar el objeto.
Push Button	'pushbutton'	Invoca un evento inmediatamente.
Toggle Button	'togglebutton'	Sólo dos estados, "on" o "off".
Radio Button	'radio'	Indica una opción que puede ser seleccionada.
Checkbox	'checkbox'	Indica el estado de una opción o atributo.
Edit Text	'edit'	Caja para editar texto.
Static Text	'text'	Muestra un string de texto en una caja.
Slider	'slider'	Usado para representar un rango de valores.
Frame	'frame'	Crea un marco que puede contener otros controles.
Listbox	'listbox'	Muestra una lista deslizable.
Popup Menu	'popupmenu'	Provee una lista de opciones.
Axes	'axes'	Crea un área de gráfico

Funcionamiento de una aplicación GUIDE

Una aplicación GUIDE consta, como se dijo, de dos archivos: *.m* y *.fig*. El archivo *.m* es el que contiene el código con las correspondencias de los botones de control de la interfaz y el archivo *.fig* contiene los elementos gráficos.

Cada vez que se adicione un nuevo elemento en la interfaz gráfica, se genera automáticamente código en el archivo .m.

Para ejecutarla, en la ventana de comandos simplemente la llamamos con el nombre con que fue guardada. Esto es:

>> *Nombre_asignado*.

O haciendo click derecho en el m-file y seleccionando la opción *[RUN] o F5*.

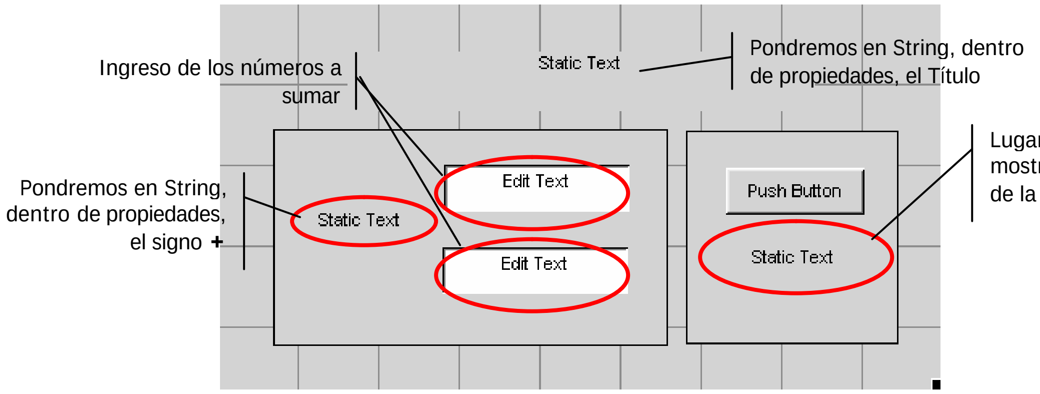
Veamos algunos ejemplos:

PRUEBA1.m **El Objetivo es sumar dos números**

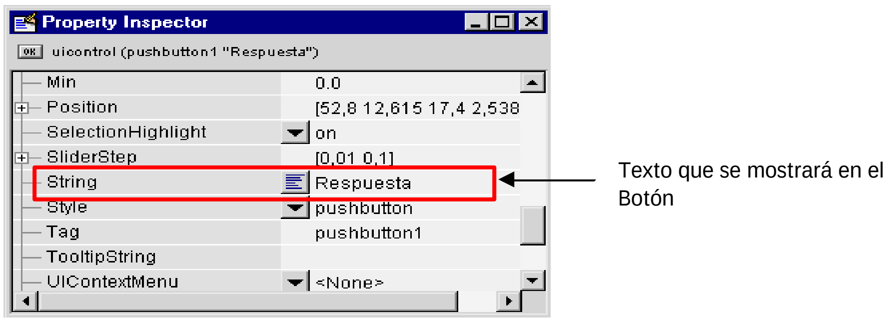
Primero, ejecutamos >> guide en la ventana de comandos, y presionamos *[Enter]*.

Seleccionamos **Blank GUI (default)** y presionamos OK.

Insertamos los componentes que muestra la siguiente figura:



Haciendo doble-click en cada componente, podemos configurar **las propiedades** de cada elemento. Configuramos el botón *pushbutton*, que al presionarlo nos mostrará el valor de la suma de los dos números ingresados:

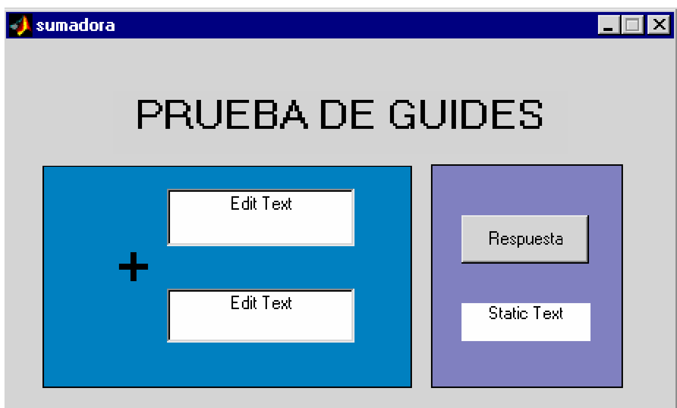


Para cambiar el nombre con el que aparecerá la función del *pushbutton* en el **m-file**, simplemente editando el campo *Tag.* y le asignamos un nuevo nombre.

Para ejecutarla presionamos (o Ctrl+T). En este momento se guarda la figura con el nombre que deseamos en la carpeta work como un archivo .fig y también con el mismo nombre un archivo M.

Llamemos a nuestra figura con el nombre prueba1. Compruebe en la ventana *Current Directory* que aparecen los archivos *prueba1.fig* y *prueba1*.m.

Complete la figura como la siguiente:



MANEJO DE DATOS ENTRE LOS ELEMENTOS DE LA APLICACIÓN Y EL ARCHIVO M

Cada uno de los objetos de MATLAB tiene un ***identificador único (handle)***. Algunos gráficos tienen muchos objetos, en cuyo caso tienen múltiples *handle*s. El ***objeto raíz*** (pantalla) es siempre único.
Tomando el programa anterior, el identificador se asigna en:

handles.output = hObject;

*handle*s, es nuestro identificador a los datos de la aplicación.
Esta definición de identificador es preservada con la siguiente instrucción:

guidata(hObject, handles);

*guidat*a, es la sentencia para salvar los datos de la aplicación y nos garantiza que cualquier cambio o asignación de propiedades o variables quede almacenado.

Cada uno de los elementos añadidos en nuestro diseño como *pushbutton, edit, text, static text* tienen una función asociada en nuestro m-file. Así, al añadir *pushbutton,* tenemos el siguiente código:

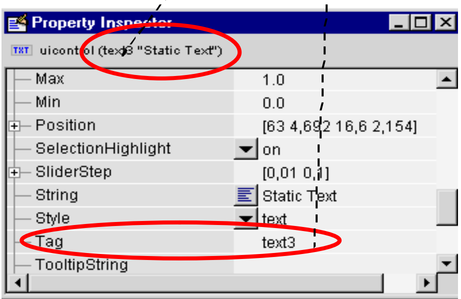
```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
```

Identificador o handles del objeto.

Cada *edit text* tendrá el siguiente código:

```
function edit1_Callback(hObject, eventdata, handles)
% hObject handle to edit1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit1 as text
```

*Static tex*t, no posee función asociada, pero sí una dirección asociada, que la podemos utilizar para escribir los resultados. Para saber cuál es esta dirección, haciendo doble-click en este componente, la ubicamos en la etiqueta *Tag*, como lo muestra la figura.



Ponemos ahora las instrucciones necesarias.....
Debajo de function edit1_Callback(hObject, eventdata, handles),donde ingresaremos **el primero** de los números, escribimos el siguiente código:

```
num1=get(hObject,'String');     %Almacena el valor ingresado en num1 en formato String
a = str2double(num1);           %Transforma a formato double (cambia de palabra a número)
handles.num1=a;                 %Almacena el valor ingresado en identificador (Handles)
guidata(hObject,handles);       %Salva datos de la aplicación
```

La instrucción ***get*** la usamos para obtener datos ingresados por el usuario. Así, la línea num1=get(hObject,'String') almacena en *num1* el valor ingresado en formato *Strin*g.
Repetimos las mismas sentencias justo debajo de function edit2_Callback(hObject, eventdata, handles) , pero esta vez usando el identificador handles.num2=b. Tendremos las siguientes sentencias.

```
num2=get(hObject,'String');     %Almacena el valor ingresado en num1 en formato String
```

```
b = str2double(num2);          %Transforma a formato double (cambia de palabra a número)
handles.num2=b;                %Almacena el valor ingresado en identificador (Handles)
guidata(hObject,handles);      %Salva datos de la aplicación
```

Hasta el momento tenemos los dos sumandos almacenados en los identificadores *handles.num1* y *handles.num2*. Como nuestro resultado se muestra al presionar el botón RESPUESTA, es momento de editar la función correspondiente a *pushbutton*.

Debajo de function pushbutton1_Callback(hObject, eventdata,handles), donde pulsaremos para obtener el resultado, editamos el siguiente código:

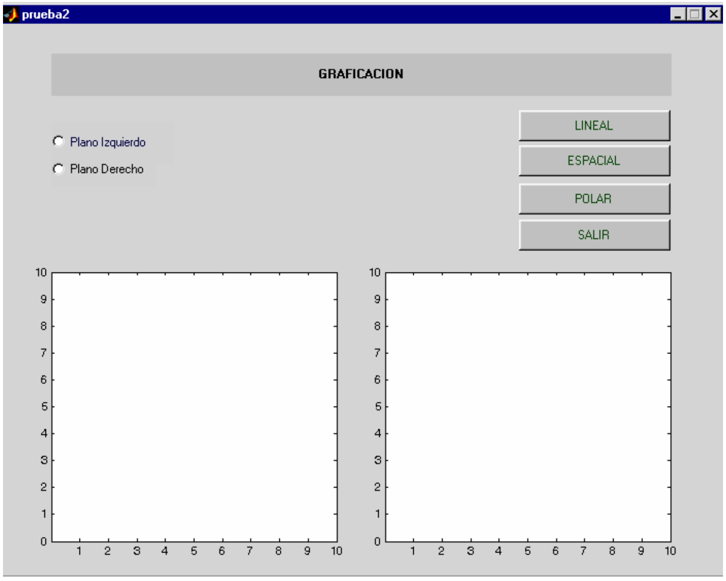
```
Sum=handles.num1+handles.num2;
set(handles.text3,'String',Sum)     % Asigna a la caja de texto donde mostrará el resultado el
                                      valor de la suma Sum como una cadena de caracteres
```

La primera sentencia es por demás obvias. Sin embargo, la segunda línea contiene la instrucción *set*, con la cual muestra un valor (*String*) en *Static text*3 (donde se mostrará el resultado de las suma), con el identificador *handles.text*3.

Bien, hasta aquí ya tenemos nuestro programa que suma dos números. Ejecutamos el programa con F5.

PRUEBA2.m **Objetivo: Graficar en dos ejes distintos. Uso del Botón Radio Botton**

Realizamos una figura con los objetos mostrados:



Grabamos la misma con el nombre *prueba2.*

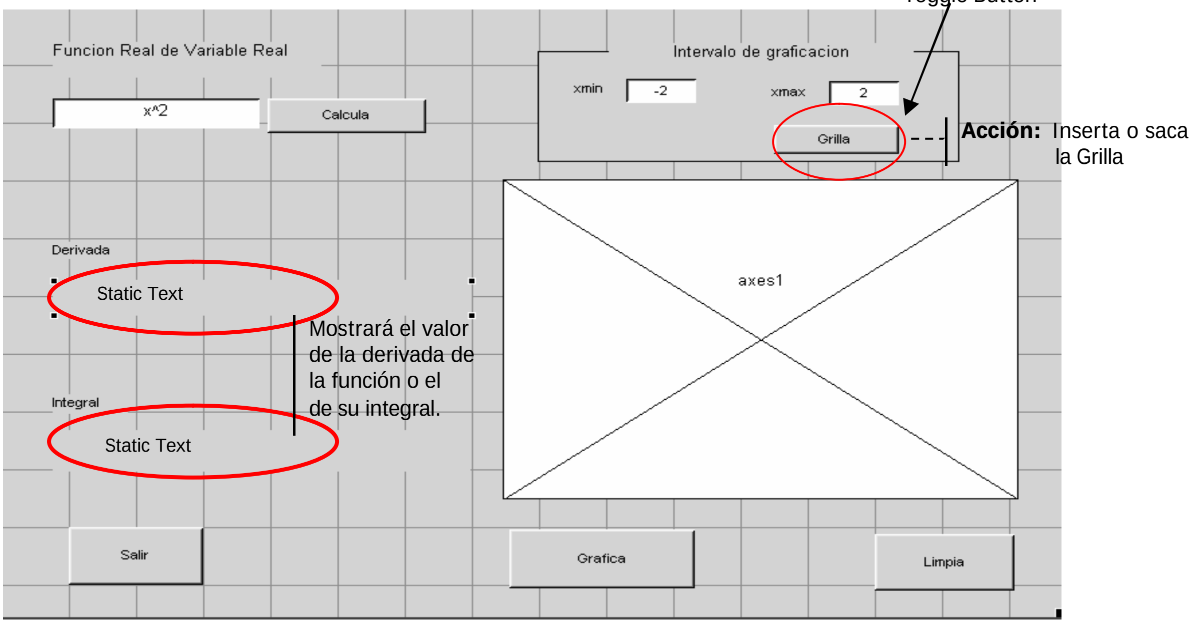
Programamos los botones como siguen:

```
%-----------------------------------------------------------------------------------------------------------
function radiobutton1_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of radiobutton1
axes(handles.axes1)                    % Selecciona los ejes en que se graficará .
set(handles.radiobutton1,'Value',1)    % Asigna el valor 1 al primer botón, es decir lo marca.
set(handles.radiobutton2,'Value',0)    % Asigna el valor 0 al segundo botón, es decir lo desmarca.
```

```
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of radiobutton2
axes(handles.axes2)                 % Selecciona los ejes que usará para graficar, en este caso el derecho.
set(handles.radiobutton2,'value',1)   % Asigna el valor 1 al segundo botón, es decir lo marca.
set(handles.radiobutton1,'value',0)   % Asigna el valor 0 al primer botón, es decir lo desmarca.
%------------------------------------------------------------------------------------------------------------
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
newplot,
x=0:0.01:10;
y=sin(x);
y1=sin(x-0.25);
y2=sin(x-0.5);
plot(x,y,x,y1,x,y2);
legend('sen(x)','sen(x-.25)','sen(x-0.5)')
xlabel('x')
ylabel('y')
title('Graficando funciones Senosoidales', 'FontSize',12)
%------------------------------------------------------------------------------------------------------------
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
newplot;
x=0:0.01:10;
plot3(sin(x),cos(x),x);
axis square; grid on
%------------------------------------------------------------------------------------------------------------
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
newplot;
x=0:0.01:10;
polar(x,sin(x).*cos(2*x),'--r');
axis square;grid on
%------------------------------------------------------------------------------------------------------------
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close;
```

PRUEBA3.m Objetivo: Ingreso de Funciones y su aplicación. Uso del Toggle Botton.

Realice la siguiente figura



Grabamos con el nombre *prueba3*

Programamos como sigue:

```
%------ Boton Calcula ------------------------------------------------------------------------------------------
function calcula_Callback(hObject, eventdata, handles)
% hObject    handle to calcula (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
h1=get(handles.la_ecuacion,'String');   % la_ecuacion es el edit texto que ingresa la función
para_derivada=findobj(gcbf,'Tag','res_der');
derivada1=diff(h1);                      % Deriva la función h1
derivada2=char(derivada1);               % Convierte en Cadena el valor de la derivada.
set(para_derivada,'String',derivada2)    % Asigna  en la variable para_derivada el valor de la cadena.

para_int=findobj(gcbf,'Tag','res_int');
integra1=int(h1);
integra2=char(integra1);
set(para_int,'String',integra2);
% --- Botón Sale ------------------------------------------------------------------------------------------
function sale_Callback(hObject, eventdata, handles) % Su Tag: Sale
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close;                                   % Cierrra la figura actual.
% --- Botón limpia ------------------------------------------------------------------------------------------
function limpia_Callback(hObject, eventdata, handles)
% hObject    handle to limpia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cla;                        % Borra axes
```

```
newplot;
para_derivada=findobj(gcbf,'Tag','res_der');
limpia=' ';                              % Genera una cadena en blanco.
set(para_derivada,'String',limpia);      % Asigna la cadena nula al campo donde muestra la derivada.
para_int=findobj(gcbf,'Tag','res_int');
set(para_int,'String',limpia);
set(handles.la_ecuacion,'string','')     % Otro modo de borrar el contenido del campo.
% --- Boton Grafica -----------------------------------------------------------------------------------------
function grafica_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%Tomar el valor de GRID y mantenerlo en la nueva gráfica.
newplot
axes(handles.axes1);
reset(gca);                          % Borra las propiedades de axes1
xmin=get(handles.xmin,'String'); % Toma el valor ingresado en el campo edit  llamado xmin
xmin=str2num(xmin);
xmax=get(handles.xmax,'String'); % Toma el valor ingresado en el campo edit  llamado xmax
xmax=str2num(xmax);                  % Convierte el string en número
x=xmin:.1:xmax;
h1=get(handles.la_ecuacion,'String');
d1=diff(h1);
int1=int(h1);
h2=char(h1);
y=subs(h2,x); % Valúa la función en x
d2=char(d1);
y1=subs(d2,x); % Valúa la derivada  en x
i2=char(int1);
y2=subs(i2,x);  % Valúa la integral  en x
plot(x,y,x,y1,x,y2);
legend('Funcion', 'Derivada', 'Integral')
handles.g=get(handles.grilla,'Value');  % Toma el valor de Toggle Button llamado Grilla
if handles.g==1
   grid on;
   set(handles.grilla,'String','GRID OFF'); % Cambia el texto del boton
else
   grid off;
   set(handles.grilla,'String','GRID ON');
end
guidata(hObject,handles)
%------------------------------------------------------------------------------------------------------------
```

Probamos, ahora, el programa.