Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Mecánica Eléctrica
Proyectos de Computación Aplicados a la ingenieria
Electronica
Ing. José Anibal Silva De Los Angeles

Tarea 7: practicas del capitulo 1 al final del canal de Youtube
Fecha entrega: 14/12/24

José Manuel Suy Morales 201902185
Pablo Andres Montufar Perez 201902235

CONEXION ENTRE OCTAVE Y POSTGRESQL

La conexion entre la base de datos y octave, muestra el codigo para hacer la conexion y poder añadir datos a la tabla de datos realizada desde pgadmin4, donde haces la consulta del estudiante mediante su id

I. CODIGO PARA LA CONEXION ENTRE OCTAVE Y POSTGRE

```
pkg load database
conn = pq_connect(setdbopts('dbname', 'postgres', 'host', 'localhost', 'port', '5432', 'user', 'postgres', 'password', 'josesuy'))

###Insertar registro
%N = pq_exec_params(conn, "Insert INTO ejemplo (id, nombre, apellido) VALUES (1, 'Alexander', 'Sanan');");

###Eliminar registro que cumpla filtro
##N = pq_exec_params(conn, "DELETE FROM ejemplo where ID = 1;");

###Modificar registro que cumpla filtro
##N = pq_exec_params(conn, "UPDATE ejemplo set ID = 25 where Nombre = 'Alexander';");

###Obtener todos los registros de una tabla
N = pq_exec_params(conn, "SELECT * FROM ejemplo;");

##printf("Toda la estructura de la db: \n");
##N

##printf("Solo la data de la db: \n");
##N.data

printf("Obtencion de informacion de un registro: \n");
ID = cell2mat(N.data(1, 1));
Nombre = cell2mat(N.data(1, 2));
Apellido = num2str(cell2mat(N.data(1, 3)));

output_str = ["El ID del estudiante ", Nombre, " ", Apellido, " es: ", num2str(ID)];
disp(output_str);
```

Figura 1: codigo de conexion y ingreso de datos
Fuente: Elaboración en octave, 2024

II. CONSULTA REALIZADA POR MEDIO DEL ID

```
Ventana de comandos
>> tarea7

conn = <PGconn object>
Obtencion de informacion de un registro:
El ID del estudiante Alexander Sanan es: 1
>>
```

Figura 2: opcion 1
Fuente: Elaboración en octave, 2024

III. CREACION DE LA TABLA EN PGADMIN Y CONSULTA

Query | Query History

```
1  CREATE TABLE ejemplo (
2      id SERIAL PRIMARY KEY,
3      nombre VARCHAR(50) NOT NULL,
4      apellido VARCHAR(50) NOT NULL
5  );
6
7  SELECT * FROM ejemplo
```

Data Output | Messages | Notifications

	id [PK] integer	nombre character varying (50)	apellido character varying (50)
1	1	Alexander	Sanan

Figura 3: tabla crada para la conexion con octave
Fuente: Elaboración en octave, 2024

IV. FUNCION HIPOTENUSA

```
## Author: joses <joses@JOSE01>
## Created: 2024-12-14

function [hipo, a_cuadrada] = hipotenusa (a, b)
    hipo = sqrt(a.^2 + b.^2);
    a_cuadrada = a.^2;
endfunction

%function [hipo, angle] = hipotenusa (a, b)
%   hipo = sqrt(a.^2 + b.^2);
%   angle = atan(a/b);
%endfunction
```

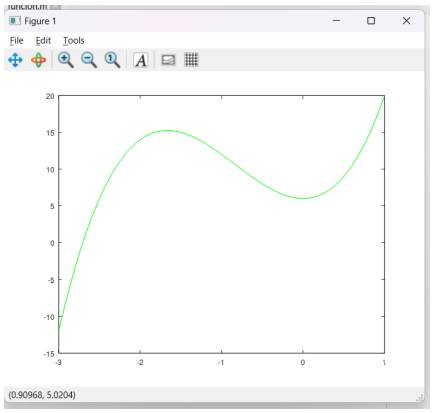
Figura 4: Codigo de la funcion hipotenusa
Fuente: Elaboracion en octave, 2024

```
#FUNCIONES

[x,b]=hipotenusa(1,2)
```

Figura 5: parametros enviados a hipotenusa
Fuente: Elaboración en octave, 2024

```
Ventana de comandos
>> examples

x = 2.2361
>> examples

x = 2.2361
b = 1
>>
```

Figura 6: ejemplo obtenido de la funcion

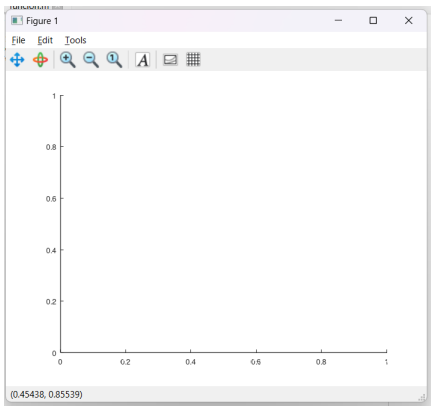


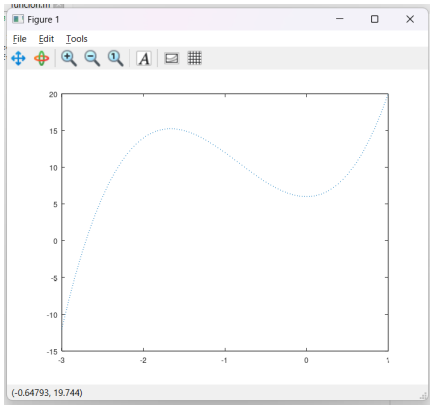
V. COFIGURACIONES DE LAS GRAFICAS

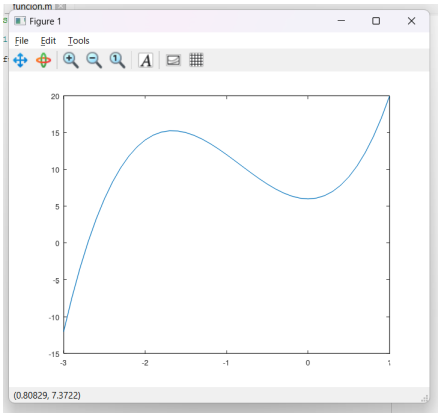
examples.m | funcion.m

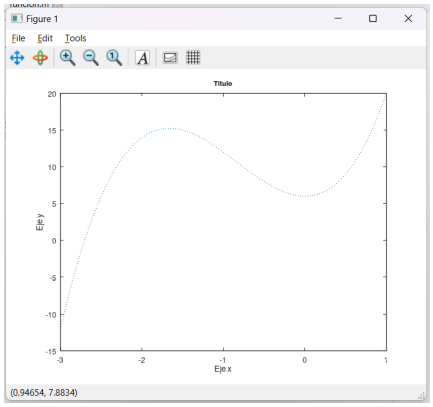
```
1  #GRAFICAS
2
3  x=[-3:0.1:1];
4  x=linspace(-3,1,50)
5  plot(x, funcion(x), 'Color','green')
6  plot(x, funcion(x), 'LineStyle','.')
7  plot(x, funcion(x), 'LineStyle',':')
8
9  stem(x, funcion(x), 'LineStyle',':')
10
11 title('Titulo')
12 ylabel('Eje y')
13 xlabel('Eje x')
14 legend('Funcion')
15
16 x=[0:0.1:4*pi]
17 y1 = sin(x);
18 y2 = cos(x);
19 hold on;
20 p1 = plot(x,y1);
21 p2 = plot(x,y2);
22 set(p1, 'Color' , 'red', 'LineWidth',2)
23 set(p2, 'Color' , 'blue','LineWidth',1)
24 ylabel('EJE Y');
25 xlabel('Eje X');
26 title('Seno y Coseno');
27 legend('Seno' , 'Coseno');
28 hold off;
29
30
```

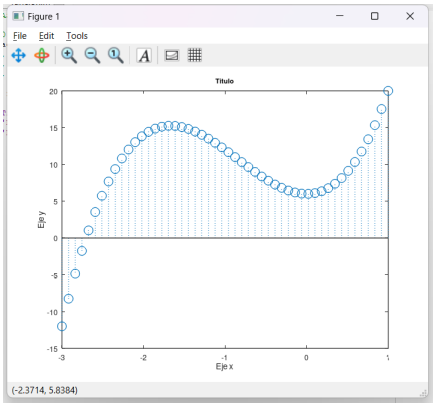
Figura 7: configuraciones de las diferentes graficas

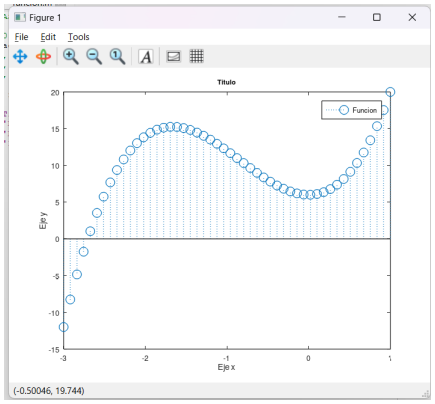


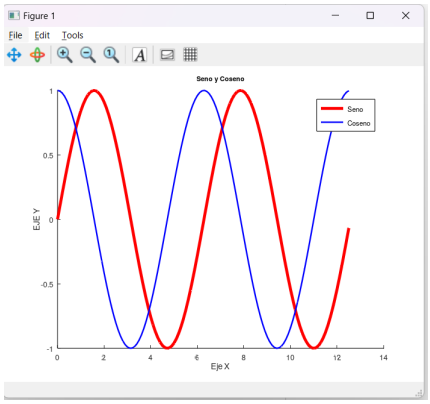












OPERADORES ARITMÉTICOS

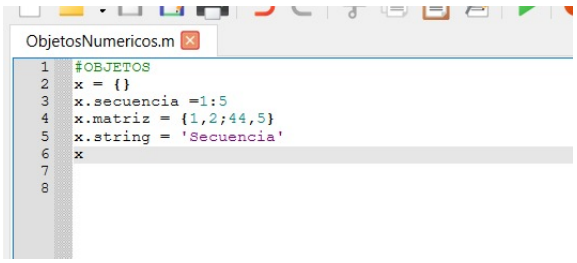
Los operadores aritméticos se utilizan para realizar operaciones matemáticas básicas:

- Suma: $x + y$
- Resta: $x - y$
- Multiplicación: $x \cdot y$
- División: x/y
- Potencia: x^y
- Incremento manual: $x = x + 1$
- Decremento manual: $x = x - 1$

```
scalar structure containing the fields:

  secuencia =

     1   2   3   4   5

  matriz =
  {
    [1,1] = 1
    [2,1] = 44
    [1,2] = 2
    [2,2] = 5
  }

  string = Secuencia

x =

scalar structure containing the fields:

  secuencia =

     1   2   3   4   5

  matriz =
  {
    [1,1] = 1
    [2,1] = 44
    [1,2] = 2
    [2,2] = 5
  }

  string = Secuencia

>>
```

Fuente: Elaboración en octave, 2024

Figura 8: CAP1 primer operador



Fuente: Elaboración en octave, 2024

Figura 9: Codigo primer operador "matriz"

ObjetosNumericos.m

```
#OBJETOS
x = {}
x.secuencia =1:5
x.matriz = {1,2;44,5}
x.string = 'Secuencia'
x.estructura = {}
x.estructura.numero = 0x177
x.estructura.letra = 'A'
x
```

Fuente: Elaboración en octave, 2024

Figura 10: Codigo segundo operador "matriz"

```
scalar structure containing the fields:

  secuencia =

     1   2   3   4   5

  matriz =
  {
    [1,1] = 1
    [2,1] = 44
    [1,2] = 2
    [2,2] = 5
  }

  string = Secuencia
  estructura =

    scalar structure containing the fields:

      numero = 375
      letra = A

>>
```

Ventana de comandos | Documentación | Editor de variables | Editor

Fuente: Elaboración en octave, 2024

Figura 11: segundo operador

```
>> ObjetosNumericos

x = 2
y = 3
ans = 5
ans = -1
ans = 6
ans = 0.6667
ans = 3
ans = 2
>>
```

Fuente: Elaboración en octave, 2024

Figura 12: tercer operador

ObjetosNumericos.m

```
#OPERADORES
x = 2
y = 3
x + y
x - y
x * y
x / y
++x # x = x + 1
--x # x = x - 1
```

Fuente: Elaboración en octave, 2024

Figura 13: Codigo 3er ejemplo

OPERADORES DE COMPARACIÓN

Los operadores de comparación devuelven valores lógicos:

- Menor que: $x < y$
- Menor o igual que: $x \leq y$
- Igualdad: $x == y$
- Mayor que: $x > y$
- Mayor o igual que: $x \geq y$
- Desigualdad: $x \neq y$

```
>> ObjetosNumericos

x = 2
y = 3
ans = 1
ans = 1
ans = 0
ans = 0
ans = 0
ans = 1
>>
```

Fuente: Elaboración en octave, 2024

Figura 14: 4 ejemplo comparacion

```
#OPERADORES de comparacion
x = 2
y = 3

x < y
x <= y
x == y # nota ; 1 "= " igual significa asignacion y 2 iguales "==" es comparacion

x > y
x >= y
x != y
```

Fuente: Elaboración en octave, 2024

Figura 15: Codigo 4 ejemplo

VI. ESTRUCTURAS DE CONTROL DE FLUJO

Las estructuras de control de flujo en programación permiten tomar decisiones y realizar acciones de forma condicional o iterativa. Son esenciales para dirigir la ejecución del código en función de ciertas condiciones o para repetir bloques de instrucciones.

A. Condicional `if-elseif-else`

Esta estructura evalúa una condición lógica y ejecuta diferentes bloques de código en función del resultado:

```
x = 5;
y = 3;

if (x > y)
  'X es mayor a Y'
elseif (x == y)
  'X y Y son iguales'
else
  'Y es mayor a X'
endif
```

En este ejemplo, dependiendo del valor de `x` y `y`, se imprimirá el mensaje correspondiente.

B. Aplicaciones Comunes

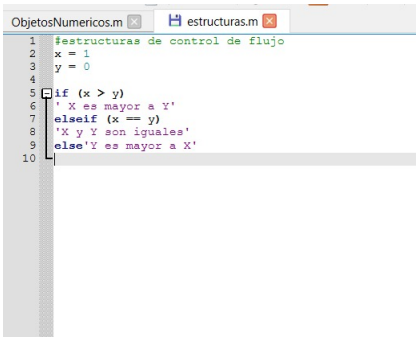
Las estructuras condicionales son útiles para:

- Comparar valores para mostrar mensajes específicos.
- Tomar decisiones basadas en datos dinámicos (como entradas del usuario).
- Controlar flujos de algoritmos, como en bucles o verificaciones.

```
>> #estructuras de control de flujo
x = 1;
y = 0;
if (x > y)
    disp('X es mayor a Y');
elseif (x == y)
    disp('X y Y son iguales');
else
    disp('Y es mayor a X');
end
X es mayor a Y
>>
```

Fuente: Elaboración en octave, 2024

Figura 16: 5 ejemplo estructuras



Fuente: Elaboración en octave, 2024

Figura 17: Codigo 5 esctructuras

VII. OPERACIONES CON MATRICES

Las matrices son fundamentales en MATLAB y Octave para representar datos numéricos en múltiples dimensiones. Pueden usarse en cálculos aritméticos, álgebra lineal y visualización de datos.

A. Definición de Matrices

Las matrices se crean usando corchetes cuadrados:

```
M = [1, 2, 3;
     7, 4, 5;
     11, 1, 2];
```

Aquí, `M` es una matriz de 3×3 donde cada fila está separada por un punto y coma (`;`) y las columnas por comas (`,`).

B. Operaciones Básicas

Las operaciones entre matrices incluyen suma, resta, multiplicación cruzada y producto punto:

```
A = M + N; % Suma de matrices
B = M - N; % Resta de matrices
C = cross(M, N); % Producto cruzado
D = dot(M(1,:), N(1,:)); % Producto punto
M_transpuesta = M'; % Traspuesta
```

C. Aplicaciones Comunes

Las operaciones con matrices son útiles para:

- Resolver sistemas de ecuaciones lineales.
- Procesamiento de imágenes y señales.
- Simulaciones científicas y cálculos avanzados.

```
N = [0, 1, 2;
     8, 10, 12;
     177, 176, 125];
% Suma y resta
A = M + N;
B = M - N;
% Producto cruz fila a fila
C = cross(M, N);
% Producto punto entre la primera fila
D = dot(M(1, :), N(1, :));
% Traspuesta de M
M_transpuesta = M';
% Mostrar resultados
disp('M + N:');
disp(A);
disp('M - N:');
disp(B);
disp('Producto cruz M x N:');
disp(C);
disp('Producto punto entre la primera fila de M y N:');
disp(D);
disp('Traspuesta de M:');
disp(M_transpuesta);
M + N:
     1     3     5
    15    14    17
   188   177   127
M - N:
     1     1     1
    -1    -6    -7
  -166  -175  -123
Producto cruz M x N:
   1151    694    601
   -177   -351   -371
      8     16     26
Producto punto entre la primera fila de M y N:
```

Fuente: Elaboración en octave, 2024

Figura 18: 5 ejemplo matrices

```
#Matriz
M = [1,2;7; 4,5,11; 1,2,3 ]
N = {0,1,2; 8,10,12; 177,176,125}
M + N
M - N
M + N
cross(M,N) #MXN
dot(M,N)
M'
```

Fuente: Elaboración en octave, 2024

Figura 19: Codigo matrices

EJEMPLO DE CÓDIGO EN OCTAVE/MATLAB

```
x = 2;
y = 3;

% Operadores aritméticos
suma = x + y;
resta = x - y;
producto = x * y;
division = x / y;
potencia = x ^ y;

% Operadores de comparación
menor = x < y;
igualdad = x == y;
desigualdad = x ~= y;
```

VIII. LINK DEL REPOSITORIO EN GITHUB

GitHub José Suy.