



Universidad de San Carlos de Guatemala

FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA MECÁNICA ELÉCTRICA
PROYECTOS DE COMPUTACIÓN APLICADA A I.E.
SECCIÓN 'N'
2DO SEMESTRE 2023
ING. ARMANDO ALONSO RIVERA CARILLO

PRIMER EXAMEN PARCIAL

Estudiante:

- Henry Isaac Pineda García, 202001466.

31 de Agosto del 2023

1er Parcial

Serie I

No. Pregunta	Respuesta
2	a)
4	a)
6	b)
8	b)
10	a)
12	a)
14	c)
16	a)
18	b)
20	a)
22	a)
24	c)

No. Pregunta	Respuesta
26	a)
28	c)
30	a)
32	a)
34	a)
36	b)
38	b)
40	a)
42	a)
44	d)
46	b)
48	a)
50	d)

Serie II

1. Es un lenguaje para el control de base de datos
2. Se usa para no tener datos repetidos en una tabla
3. `Insert into nombre de la tabla (nombre de columnas,...) values (valor 1,...)`
4. `Delet from descripción where columna = valor;`
5. `Update nombre de la tabla set columna = valor... where condición`
6. Es un filtro para seleccionar datos, es decir que solo los valores que cumplan con la función serán afectados
8. Es una forma de visualizar los datos almacenados de una tabla
10. Es un bloque de código que se puede ejecutar cada que es llamada, sus valores de entrada pueden variar y esta devuelve un resultado
12. $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$
 $B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$
 $C = A \times B$
 $C = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$
14. Se usa el comando `mean`
`B = mean(A)`
16. `b = sqrt(a)` → 'b' es la raíz cuadrada de 'a'
18. Se usa la función `max`
`m = max(A)`
20. Se usa la función `randi`, que puede generar un número aleatorio en un rango de valores de 0 hasta un valor máximo
`b = randi(valor max)`

Serie III.A

I. PROBLEMA 1

```
Programa1.m x
1  %{
2  Programa de registro de estudiantes: Este programa utiliza Octave y una base de
3  datos para almacenar información de estudiantes, como su nombre, edad, género y
4  dirección. Ofrece opciones para agregar nuevos estudiantes, editar la información
5  de estudiantes existentes y eliminar estudiantes de la base de datos.
6  %}
7
8
9  pkg load database
10
11  conn = pq_connect(setdbopts('dbname','0980 Proyectos','host','localhost',
12  'port','5432','user','postgres','password','202001466'));
13
14  consulta=1;
15  while consulta
16
17      fprintf('Bienvenido, que operación quiere realizar: \n 1.Agregar a un estudiante,
18      \n 2.Editar la información de estudiantes existentes.
19      \n 3.Revisar los datos de un estudiante.
20      \n 4.Revisar Toda la base de datos.
21      \n 5.Eliminar un estudiantes de la base de datos.
22      \n 6.Eliminar Toda la Base de Datos. \n \n');
23
24
25      opciones=input("Seleccione el numero de su operación: ");
26      fprintf(' \n')
27
28      if(opciones==1)%1.Agregar a un estudiante.
29
30          nombre=input("Ingrese Nombre del Estudiante: ", 's');
31          fprintf(' \n');
32
33          edad=input("Ingrese la Edad del Estudiante: ");
34          fprintf(' \n');
35
36          g=1;
37          while g
38              genero=input("¿El Genero del Estudiante es 'M' o 'F'? ", 's');
39              if(genero=='M') || (genero=='m')
40                  genero="Masculino";
```

Figura 1: Programa 1, 1.

```

40     genero="Masculino";
41     g=0;
42     elseif(genero=='F') || (genero=='f')
43         genero="Femenino";
44         g=0;
45     else
46         fprintf('El genero no es reconocido: \n')
47     endif
48     fprintf(' \n');
49 endwhile
50
51     direccion=input("Ingrese su Direccion: ", 's');
52     fprintf(' \n');
53
54     fprintf(" Su nombre es: %s \n Su edad es: %d \n Su genero es: %s \n Su direccion es: %s",...
55         nombre, edad, genero,direccion)
56
57     query = sprintf("insert into El_Programal (Nombre, Edad, Genero, Direccion) values ('%s',
58         '%d', '%s', '%s')", ...
59         nombre, edad, genero,direccion);
60     pq_exec_params(conn, query);
61     fprintf(' \n')
62
63
64 elseif (opciones==2)%2.Editar la información de estudiantes existentes.
65
66     editar=input("Ingrese Nombre del Estudiante a editar: ", 's');
67     fprintf(' \n');
68
69     nombre=input("Ingrese Nuevo Nombre del Estudiante: ", 's');
70     fprintf(' \n');
71
72     edad=input("Ingrese la Edad del Estudiante: ");
73     fprintf(' \n');
74
75     g=1;

```

Figura 2: Programa 1, 2.

```

75 g=1;
76 while g
77     genero=input("¿El Genero del Estudiante es 'M' o 'F'? ", 's');
78     if(genero=='M') || (genero=='m')
79         genero="Masculino";
80         g=0;
81     elseif(genero=='F') || (genero=='f')
82         genero="Femenino";
83         g=0;
84     else
85         fprintf('El genero no es reconocido: \n')
86     endif
87     fprintf(' \n');
88 endwhile
89
90 direccion=input("Ingrese su Direccion: ", 's');
91 fprintf(' \n');
92
93 fprintf(" Su nombre es: %s \n Su edad es: %d \n Su genero es: %s \n Su direccion es: %s",...
94     nombre, edad, genero,direccion)
95
96 query = sprintf("UPDATE E1_Programal SET Nombre='%s', Edad='%d', Genero='%s', Direccion='%s'
97 WHERE Nombre = ('%s');", ...
98     nombre, edad, genero,direccion, editar);
99 pq_exec_params(conn, query);
100 fprintf(' \n')
101
102 elseif (opciones==3)%3.Revisar los datos de un estudiante.
103     revisar=input("Nombre del estudiante a Revisar: ", 's');
104     query = sprintf("select * from E1_Programal WHERE Nombre =('%s')", ...
105         revisar)
106     Historial_Estudiante=pq_exec_params(conn, query)
107
108
109 elseif (opciones==4)%4.Revisar Toda la base de datos.
110     Historial_Postgresql=pq_exec_params(conn, 'select * from E1_Programal;')

```

Figura 3: Programa 1, 3.

```

110 Historial_Postgresql=pq_exec_params(conn, 'select * from El_Programa1;')
111
112 elseif (opciones==5)%5.Eliminar un estudiantes de la base de datos
113     estudiante=input("Nombre del estudiante a Eliminar: ", 's');
114     query = sprintf("DELETE FROM El_Programa1 WHERE Nombre =('%s')", ...
115         estudiante);
116     pq_exec_params(conn, query);
117
118
119 elseif (opciones==6)% 6.Eliminar Toda la Base de Datos.
120     Borrar_Tabla=pq_exec_params(conn, "DELETE FROM El_Programa1;");
121
122 else
123     fprintf("No selecciono ninguna Opcion valida.")
124 endif
125
126 fprintf(' \n \n')
127
128 consulta=yes_or_no("¿Quieres realizar otra operacion: ");
129
130
131 endwhile
132
133 fprintf(' \n \n')
134

```

Figura 4: Programa 1, 4.

II. PROBLEMA 2

```
Programa2.m Serie_4A.m
1  %{
2  Programa de seguimiento de presupuesto personal: Este programa utiliza Octave y
3  una base de datos para ayudar a los usuarios a realizar un seguimiento de sus
4  gastos y presupuestos personales. Ofrece opciones para ingresar nuevos gastos,
5  ver un resumen de los gastos acumulados y ajustar los presupuestos según sea necesario.
6  %}
7
8
9  pkg load database
10
11  conn = pq_connect(setdbopts('dbname','0980 Proyectos','host','localhost',
12  'port','5432','user','postgres','password','202001466'));
13
14  consulta=1;
15  while consulta
16
17      fprintf('Bienvenido, que operación quiere realizar: \n 1.Ingresar nuevos gastos.
18      \n 2.Ajustar los presupuestos.
19      \n 3.Revisar un gasto.
20      \n 4.Resumen de los gastos acumulados.
21      \n 5.Eliminar un gasto de la base de datos.
22      \n 6.Eliminar Toda la Base de Datos. \n \n');
23
24
25      opciones=input("Seleccione el numero de su operación: ");
26      fprintf(' \n')
27
28      if(opciones==1)%1.Ingresar nuevos gastos.
29          gasto=input("Ingrese el nombre del Gasto: ", 's');
30          fprintf(' \n');
31          precio=input("Ingrese el total del Gasto: ");
32          fprintf(' \n');
33
34          fprintf(" Se gasto en: %s un total de: %d \n \n ",...
35              gasto, precio)
36
37          query = sprintf("insert into El_Programa2 (Gasto, Precio) values ('%s', '%d')", ...
38              gasto,precio);
39          pq_exec_params(conn, query);
40          fprintf(' \n')
```

Figura 5: Programa 2, 1.


```

40     fprintf(' \n')
41
42
43 elseif (opciones==2)%2.Ajustar los presupuestos.
44     editar=input("Ingrese Nombre del Gasto a editar: ", 's');
45     fprintf(' \n');
46
47     gasto=input("Ingrese Nombre del Nuevo Gasto: ", 's');
48     fprintf(' \n');
49
50     precio=input("Ingrese Nombre del Nuevo Precio: ");
51     fprintf(' \n');
52
53     fprintf(" Se gasto en: %s un total de: %d \n \n ",...
54         gasto, precio)
55
56     query = sprintf("UPDATE El_Programa2 SET Gasto='%s', Precio='%d' WHERE Gasto = ('%s');", .
57         gasto,precio, editar);
58     pq_exec_params(conn, query);
59     fprintf(' \n')
60
61
62 elseif (opciones==3)%3.Revisar un gasto.
63     revisar=input("Nombre del Gasto a Revisar: ", 's');
64     query = sprintf("select * from El_Programa2 WHERE Gasto =('%s')", ...
65         revisar)
66     Historial_Estudiente=pq_exec_params(conn, query)
67
68
69 elseif (opciones==4)%4.Resumen de los gastos acumulados.
70     Historial_Postgresql=pq_exec_params(conn, 'select * from El_Programa2;')
71
72
73 elseif (opciones==5)%5.Eliminar un gasto de la base de datos.
74     estudiante=input("Nombre del Gasto a Eliminar: ", 's');
75     query = sprintf("DELETE FROM El_Programa2 WHERE Gasto =('%s')", ...

```

Figura 6: Programa 2, 2.

```

75 query = sprintf("DELETE FROM E1_Programa2 WHERE Gasto =('%s')", ...
76 estudiante);
77 pq_exec_params(conn, query);
78
79
80 elseif (opciones==6)%6.Eliminar Toda la Base de Datos.
81   Borrar_Tabla=pq_exec_params(conn, "DELETE FROM E1_Programa2;");
82
83 else
84   fprintf("No selecciono ninguna Opcion valida.")
85 endif
86
87 fprintf(' \n \n')
88
89 consulta=yes_or_no("¿Quieres realizar otra operacion: ");
90
91
92 endwhile
93
94 fprintf(' \n \n')
95

```

Figura 7: Programa 2, 2.

III. PROBLEMA 3

IV. PROBLEMA 4

V. PROBLEMA 5

Serie III.B

VI. REPOSITORIO

<https://github.com/HipG-3007/Proyectos.git>

Serie IV

1st Henry Isaac Pineda García,
202001466.

Escuela de Mecánica Eléctrica
Facultad de Ingeniería.
Universidad de San Carlos de Guatemala.
Ciudad de Guatemala, Guatemala.

I. CÓDIGO

I-A. A) Grabador, Reproductor y Graficador de Audio

```
1 % Comprueba si estamos ejecutando en MATLAB o en Octave
2 if (exist('OCTAVE_VERSION', 'builtin') ~= 0)
3     % Estamos en Octave
4     pkg load signal;
5 end
6
7 % Menú principal
8 opcion = 0;
9 while opcion ~= 5
10     disp('Seleccione una opción:');
11     disp('1. Grabar');
12     disp('2. Reproducir');
13     disp('3. Graficar');
14     disp('4. Graficar densidad');
15     disp('5. Salir');
16     opcion = input('Ingrese su elección: ');
17
18     switch opcion
19     case 1
20         % Grabación de audio
21         try
22             duracion = input('Ingrese la duración de la grabación en segundos: ');
23             disp('Comenzando la grabación...');
24             recObj = audiorecorder;
25             recordblocking(recObj, duracion);
26             disp('Grabación finalizada. ');
27             data = getaudiodata(recObj);
28             audiowrite('audio.wav', data, recObj.SampleRate);
29             disp('Archivo de audio grabado correctamente. ');
30         catch
31             disp('Error al grabar el audio. ');
32         end
33     case 2
34         % Reproducción de audio
35         try
36             [data, fs] = audioread('audio.wav');
37             sound(data, fs);
38         catch
39             disp('Error al reproducir el audio. ');
40         end
41     end
42 end
```

Figura 1: Grabador, Reproductor y Graficador de Audio, 1

```
40 end
41 case 3
42     % Gráfico de audio
43     try
44         [data, fs] = audioread('audio.wav');
45         tiempo = linspace(0, length(data)/fs, length(data));
46         plot(tiempo, data);
47         xlabel('Tiempo (s)');
48         ylabel('Amplitud');
49         title('Audio');
50     catch
51         disp('Error al graficar el audio. ');
52     end
53 case 4
54     % Graficando espectro de frecuencia
55     try
56         disp('Graficando espectro de frecuencia...');
57         [audio, Fs] = audioread('audio.wav');
58         N = length(audio);
59         f = linspace(0, Fs/2, N/2+1);
60         ventana = hann(N);
61         Sxx = pwelch(audio, ventana, 0, N, Fs);
62         plot(f, 10*log10(Sxx(1:N/2+1)));
63         xlabel('Frecuencia (Hz)');
64         ylabel('Densidad espectral de potencia (dB/Hz)');
65         title('Espectro de frecuencia de la señal grabada. ');
66     catch
67         disp('Error al graficar el espectro de frecuencia. ');
68     end
69 case 5
70     % Salir
71     disp('Saliendo del programa...');
72 otherwise
73     disp('Opción no válida. ');
74 end
75 end
```

Figura 2: Grabador, Reproductor y Graficador de Audio, 2

I-A1. Comentario y Observaciones: El programa despliega de 5 opciones, que con las cuales podemos grabar, reproducir, realizar gráficas o cerrar el programa.

Al seleccionar la opción de grabar, debemos indicar por cuantos segundos vamos a grabar, para poder tener un control del tiempo limite, el programa crea un archivo de audio en el cual se almacenara la grabación.

La calidad del sonido es baja y al tener un micrófono inalámbrico se puede entender que allá interferencias o ruido que monten al audio.

Su reproducción se da de el ultimo archivo de audio creado, en la primera gráfica se puede observar la modulación que tiene la voz, esta modulación podría ser PCM y la otra gráfica que nos muestra la densidad que llega a tener la grabación mas el ruido.

Un programa muy interesante tanto del lado de la programación como del lado de las comunicaciones digitales.

I-B. B) Entrenador y Reconocimiento Facial

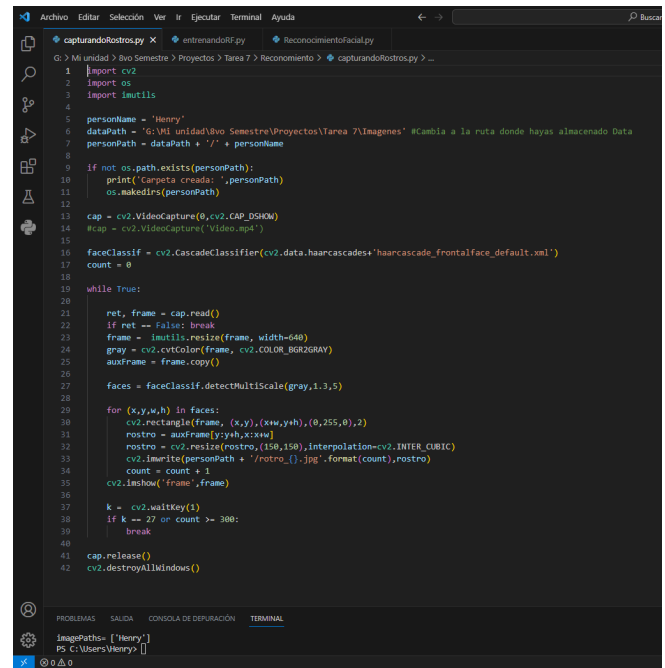


Figura 3: Captura de Rostros

```

Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda
+ capturandoRostros.py  + entrenandoRF.py  + ReconocimientoFacial.py
G: > Mi unidad > 8vo Semestre > Proyectos > Tarea 7 > VideoTutorialesAuxiliatura090-main > Reconocimiento > entrenandoRF.py
6  peopleList = cv2.listdir(dataPath)
7  print('Lista de personas: ', peopleList)
8
9  labels = []
10 facesData = []
11 label = 0
12
13 for nameDir in peopleList:
14     personPath = dataPath + '/' + nameDir
15     print('Leyendo las imágenes')
16
17     for fileName in os.listdir(personPath):
18         print('Rostros: ', nameDir + '/' + fileName)
19         labels.append(label)
20         facesData.append(cv2.imread(personPath + '/' + fileName, 0))
21         #image = cv2.imread(personPath + '/' + fileName, 0)
22         #cv2.imshow('image', image)
23         #cv2.waitKey(10)
24         label = label + 1
25
26 #print('Labels= ', labels)
27 #print('Número de etiquetas 0: ', np.count_nonzero(np.array(labels)==0))
28 #print('Número de etiquetas 1: ', np.count_nonzero(np.array(labels)==1))
29
30 # Métodos para entrenar el reconocedor
31 #face_recognizer = cv2.face.EigenFaceRecognizer_create()
32 #face_recognizer = cv2.face.FisherFaceRecognizer_create()
33 face_recognizer = cv2.face.LBPHFaceRecognizer_create()
34
35 # Entrenando el reconocedor de rostros
36 print('Entrenando...')
37 face_recognizer.train(facesData, np.array(labels))
38
39 # Almacenando el modelo obtenido
40 #face_recognizer.write('modeloEigenFace.xml')
41 #face_recognizer.write('modeloFisherFace.xml')
42 face_recognizer.write('modeloLBPHFace.xml')
43 print('Modelo almacenado...')
44
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL
imagePaths= ['Henry']
PS C:\Users\Henry>

```

Figura 4: Entrenador

```

Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda
+ capturandoRostros.py  + entrenandoRF.py  + ReconocimientoFacial.py
G: > Mi unidad > 8vo Semestre > Proyectos > Tarea 7 > VideoTutorialesAuxiliatura090-main > Reconocimiento > ReconocimientoFacial.py
1  import cv2
2  import os
3
4  dataPath = 'G:\Mi unidad\8vo Semestre\Proyectos\Tarea 7\Imagenes' #Cambia a la ruta donde hayas almacenado Data
5  imagePaths = os.listdir(dataPath)
6  print('imagePaths= ', imagePaths)
7
8  #face_recognizer = cv2.face.EigenFaceRecognizer_create()
9  #face_recognizer = cv2.face.FisherFaceRecognizer_create()
10 face_recognizer = cv2.face.LBPHFaceRecognizer_create()
11
12 # Leyendo el modelo
13 #face_recognizer.read('modeloEigenFace.xml')
14 #face_recognizer.read('modeloFisherFace.xml')
15 face_recognizer.read('modeloLBPHFace.xml')
16
17 cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
18 #cap = cv2.VideoCapture('video.mp4')
19
20 faceClassif = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
21
22 while True:
23     ret, frame = cap.read()
24     if ret == False: break
25     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
26     auxFrame = gray.copy()
27
28     faces = faceClassif.detectMultiScale(gray, 1.3, 5)
29
30     for (x,y,w,h) in faces:
31         rostro = auxFrame[y:y+h, x:x+w]
32         rostro = cv2.resize(rostro, (150, 150), interpolation=cv2.INTER_CUBIC)
33         result = face_recognizer.predict(rostro)
34
35         cv2.putText(frame, '{}'.format(result), (x,y-5), 1, 1.3, (255, 255, 0), 1, cv2.LINE_AA)
36         ...
37
38 # EigenFaces
39 if result[1] < 5700:
40     cv2.putText(frame, '{}'.format(imagePaths[result[0]]), (x,y-25), 2, 1.1, (0, 255, 0), 1, cv2.LINE_AA)
41     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 255, 0), 2)
42
43 else:
44     cv2.putText(frame, 'Desconocido', (x,y-20), 2, 0.8, (0, 0, 255), 1, cv2.LINE_AA)
45     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 0, 255), 2)
46
47 # FisherFace
48
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL
imagePaths= ['Henry']
PS C:\Users\Henry>

```

Figura 5: Reconocimiento Facial 1

```

37
38 # EigenFaces
39 if result[1] < 5700:
40     cv2.putText(frame, '{}'.format(imagePaths[result[0]]), (x,y-25), 2, 1.1, (0, 255, 0), 1, cv2.LINE_AA)
41     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 255, 0), 2)
42
43 else:
44     cv2.putText(frame, 'Desconocido', (x,y-20), 2, 0.8, (0, 0, 255), 1, cv2.LINE_AA)
45     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 0, 255), 2)
46
47 # FisherFace
48 if result[1] < 500:
49     cv2.putText(frame, '{}'.format(imagePaths[result[0]]), (x,y-25), 2, 1.1, (0, 255, 0), 1, cv2.LINE_AA)
50     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 255, 0), 2)
51
52 else:
53     cv2.putText(frame, 'Desconocido', (x,y-20), 2, 0.8, (0, 0, 255), 1, cv2.LINE_AA)
54     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 0, 255), 2)
55
56 # LBPHFace
57 if result[1] < 70:
58     cv2.putText(frame, '{}'.format(imagePaths[result[0]]), (x,y-25), 2, 1.1, (0, 255, 0), 1, cv2.LINE_AA)
59     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 255, 0), 2)
60
61 else:
62     cv2.putText(frame, 'Desconocido', (x,y-20), 2, 0.8, (0, 0, 255), 1, cv2.LINE_AA)
63     cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 0, 255), 2)
64
65 cv2.imshow('frame', frame)
66 k = cv2.waitKey(1)
67 if k == 27:
68     break
69
70 cap.release()
71 cv2.destroyAllWindows()

```

Figura 6: Reconocimiento Facial 2

II. RESULTADOS

```

--:-----
Ingrese su elección: 1
Ingrese la duración de la grabación en segundos: 10
Comenzando la grabación...
Grabación finalizada.
Archivo de audio grabado correctamente.
Seleccione una opción:
1. Grabar
2. Reproducir
3. Graficar
4. Graficar densidad
5. Salir
Ingrese su elección: 2
Seleccione una opción:

```

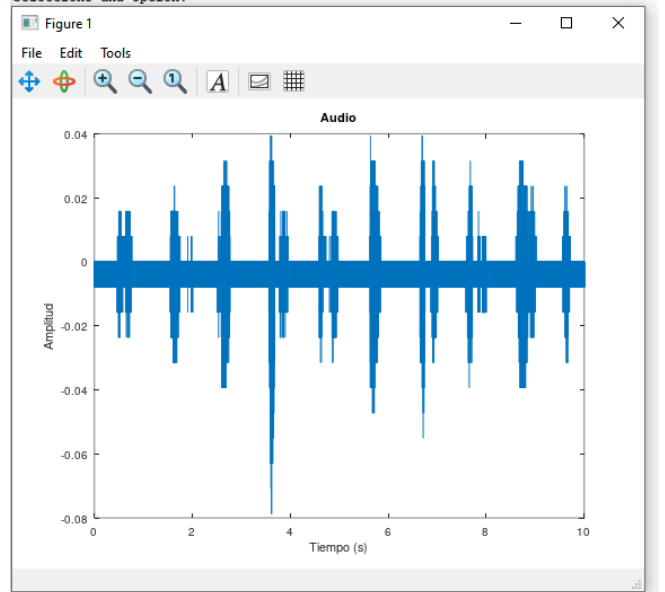


Figura 7: Gráfica de la modulación de la voz.

```
Ingrese su elecci3n: 3
Seleccione una opci3n:
1. Grabar
2. Reproducir
3. Graficar
4. Graficar densidad
5. Salir
Ingrese su elecci3n: 4
Graficando espectro de frecuencia...
Seleccione una opci3n:
```

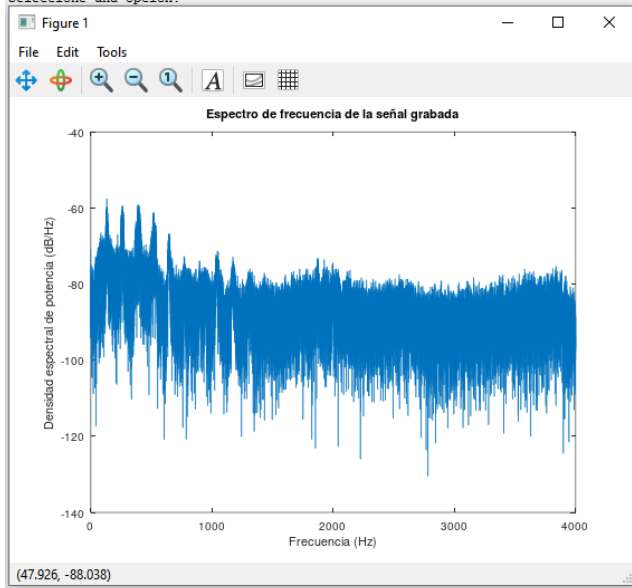


Figura 8: Gr1fica de la densidad de la voz.

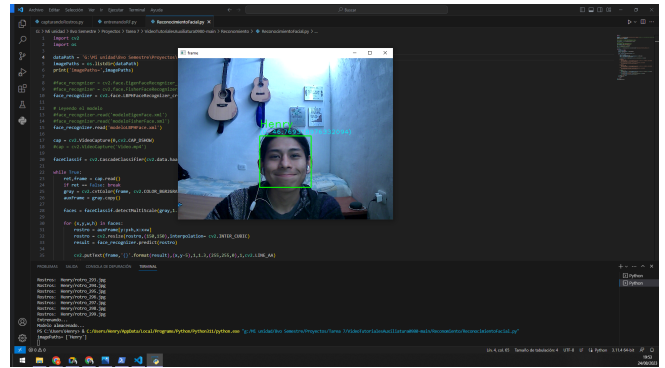


Figura 9: Reconocimiento Facial

III. REPOSITORIO

<https://github.com/HipG-3007/Proyectos.git>