

**UNIVERSIDAD NACIONAL DE TRUJILLO**

Facultad de Ingeniería

Programa de Ingeniería Mecatrónica

LABORATORIO N° 6

“ARCHIVOS - MÓDULOS - LIBRERÍAS DIVERSAS”

**DESARROLLO DE GUIA DE LABORATORIO**

PROGRAMACIÓN I

**ESTUDIANTE(S) :**

**SÁNCHEZ ROJAS JHONATAN ARTEMIO**

**VALDIVIEZO JIMÉNEZ VÍCTOR JAVIER**

**VIGO VILLAR CRISTHIAN AARON**

**DOCENTE :**

**ASTO RODRIGUEZ EMERSON MAXIMO**

**CICLO :**

**2022 I**

Trujillo, Perú

2022

**INDICE**

[RESUMEN 3](#_Toc109988806)

[DESARROLLO DEL LABORATORIO 4](#_Toc109988807)

[1.1. Desarrollo de la experiencia 4](#_Toc109988808)

[a) Ejercicio 1 4](#_Toc109988809)

[b) Ejercicio 2 5](#_Toc109988810)

[c) Ejercicio 3 7](#_Toc109988811)

[d) Ejercicio 4 9](#_Toc109988812)

[1.2. Resultados de la experiencia 10](#_Toc109988813)

[a) Ejercicio 1 10](#_Toc109988814)

[b) Ejercicio 2 11](#_Toc109988815)

[c) Ejercicio 3 12](#_Toc109988816)

[d) Ejercicio 4 13](#_Toc109988817)

[1.3. Desarrollo de test de comprobación 14](#_Toc109988818)

[1.4. Recomendaciones 15](#_Toc109988819)

[1.5. Conclusiones 15](#_Toc109988820)

[REFERENCIAS BIBLIOGRÁFICAS 16](#_Toc109988821)

[ANEXOS 17](#_Toc109988822)

[1.1 Código completo del ejercicio 1: 17](#_Toc109988823)

[1.2 Código completo del ejercicio 2: 18](#_Toc109988824)

[1.3 Código completo del ejercicio 3: 19](#_Toc109988825)

[1.4 Código completo del ejercicio 4: 20](#_Toc109988826)

# RESUMEN

El presente informe de laboratorio supone un análisis especifico del desarrollo de 4 actividades propuestas sobre programación. Las cuales piden la elaboración de códigos en Python para el desarrollo de cierto programa que se ejecute de la forma especificada en la actividad. La realización de estas actividades surge en torno a la práctica de un curso de Python, por lo que, en este caso, supone el uso de herramientas como los archivos y módulos. Y, de las ya conocidas formas básicas para la iteración, condicionales, excepciones, entre otros.

Finalmente, se pudo llegar a los correctos resultados pedidos en cada ejercicio, concluyendo en lo fácil que es la instalación de nuevos módulos/librerías para facilitar la creación de muchos más códigos, orientados para una infinidad de temas.

# DESARROLLO DEL LABORATORIO

## Desarrollo de la experiencia

### Ejercicio 1

Explicación en el video:

<https://drive.google.com/file/d/1VSRB81_g_gl4hI_eGF7di7EpgO7W7Y0H/view?usp=sharing>

### Ejercicio 2

Valdiviezo Jimenez, Victor Javier

* **Escribir un programa que consulte si uno quiere saber sobre algún tema en específico, y luego responda sobre lo consultado usando información de Wikipedia. El programa debe responder usando voz.**
* Empezamos importando los módulos que se van a necesitar para la realización del programa, dándonos cuenta, en el ítem, que serán necesarios los módulos de síntesis de voz, reconocimiento de audio y Wikipedia. Además, los inicializamos.

import pyttsx3

import speech\_recognition

import wikipedia

engine = pyttsx3.init()

recognizer = speech\_recognition.Recognizer()

* Definimos la función “sintesis(text)” que hará decir al programa el valor de “text” que se evalué en la función.

def sintesis(text):

 engine.say(text)

 engine.runAndWait()

* Definimos la función “reconocimiento ( )” que guardara la búsqueda que el usuario hará por voz en el valor “mensaje“ y finalmente retornara este mensaje.

def reconocimiento():

    try:

        with speech\_recognition.Microphone() as source:

            print("Escuchando... ")

            audio = recognizer.listen(source)

            mensaje = recognizer.recognize\_google(audio,language="es")

            print(mensaje)

    except:

       pass

    return mensaje

* Finalmente, definimos la función “buscar\_Wikipedia ( )” que pasara el valor de “mensaje” hacia el buscador de Wikipedia y arrojara el resultado mediante voz, usando la función “sintesis ( )”. Adicionalmente, agregamos un try except dentro de la función, para evitar la paralización del programa en caso no se reconozca la búsqueda.

def buscar\_wikipedia():

    try:

        sintesis("Hola, soy tu asistente de wikipedia, di lo que estas buscando")

        mensaje = reconocimiento()

        busqueda = mensaje

        wikipedia.set\_lang("es")

        respuesta = wikipedia.summary(busqueda, sentences = 1)

        sintesis(respuesta)

    except:

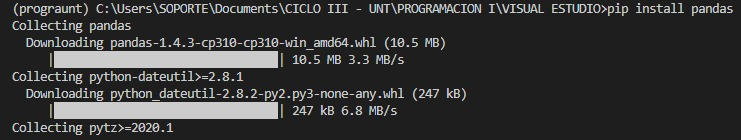
        sintesis("¡No se encontraron resultados para tu busqueda!")

buscar\_wikipedia()

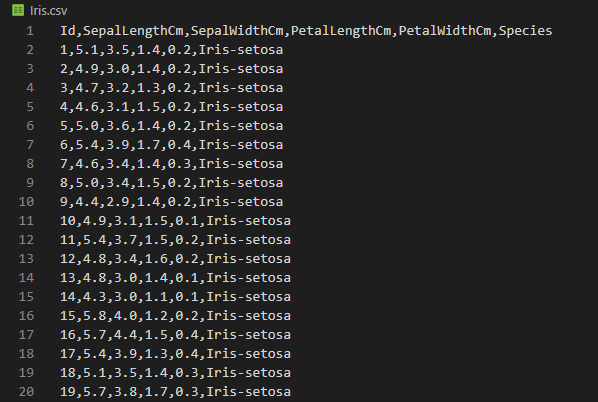
### Ejercicio 3

Valdiviezo Jimenez, Victor Javier

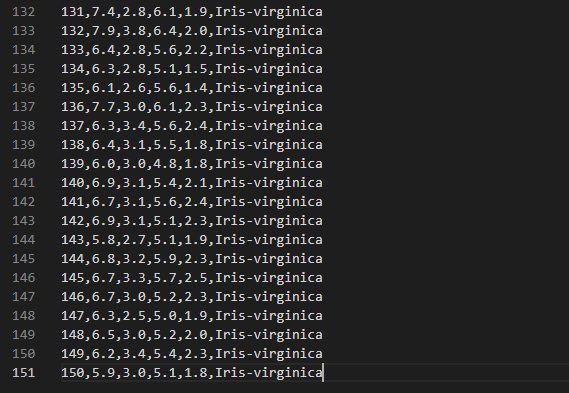
* **Escribir un programa que lea un archivo csv e imprima la información de forma agradable en la consola.**
* Para esto descargamos el módulo de pandas, que nos permite leer el archivo en forma de una tabla con encabezados.



* Seleccionamos el archivo csv con el que se va a trabajar. En este caso, el archivo elegido es “Iris.csv”, compuesto por 151 filas y 6 columnas. Significando la primera fila los encabezados de las columnas.



**…**



* Empezamos con la escritura del codigo, importando el archivo y el modulo pandas.

import csv

import pandas as pd

* Inicializamos el modulo panda con la funcion “read\_csv” y agregamos el comando “index\_col” para cambiar el idenficador propio de la dataframe a la columna “Id”.

df = pd.read\_csv("Iris.csv", index\_col = "Id")

* Finalmente imprimimos la funcion.

print(df)

* Cabe resaltar que en casos donde las filas son bastantes, la dataframe se imprimira con las 5 primeras filas y las 5 ultimas. Esto es suficiente para el objetivo del item, que nos pide que la lectura sea agradable. Sin embargo, existen funciones para poder ver mas filas como es el “head(n)” o el “tail(n)”, la primera hace un llamado a las n primeras filas, mientras que la segunda a las n ultimas filas.

print(df.head(10))

print(df.tail(10))

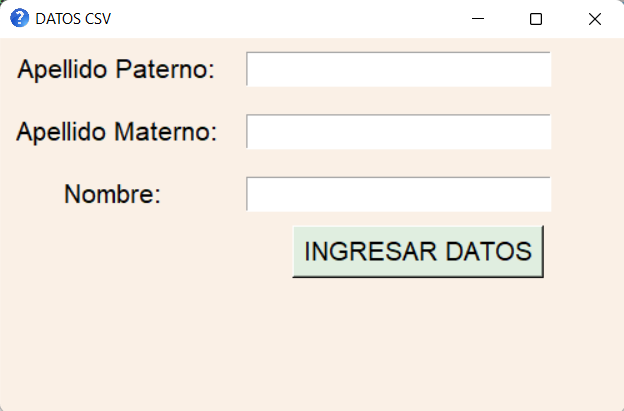
### Ejercicio 4

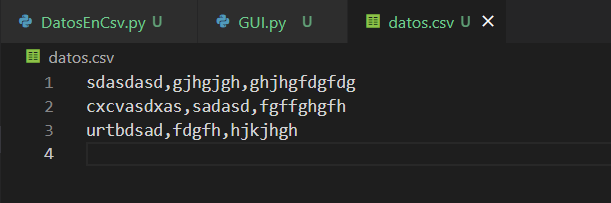
## Resultados de la experiencia

Link del repositorio: <https://github.com/VictorValdiviezo/Laboratorio-8-Grupo1-PI-UNT_2022.git>

### Ejercicio 1







### Ejercicio 2

Como el programa usa el micrófono para poder ejecutarse, ocurrieron muchos problemas para poder grabar la ejecución de este mismo; puesto que, las grabadoras con las que se intentó usaban también el micrófono para poder grabar lo que se diga y al momento de ejecutar el código, el micrófono de desconectaba. Por consiguiente, para comprobar la correcta ejecución del programa se recomienda copiar el código completo en su lector de Python de preferencia y ejecutarlo. Además, cabe recordar que para la correcta ejecución es necesario la instalación de los módulos: síntesis de voz, reconocimiento de audio y Wikipedia.

import pyttsx3

import speech\_recognition

import wikipedia

engine = pyttsx3.init()

recognizer = speech\_recognition.Recognizer()

def sintesis(text):

 engine.say(text)

 engine.runAndWait()

def reconocimiento():

    try:

        with speech\_recognition.Microphone() as source:

            print("Escuchando... ")

            audio = recognizer.listen(source)

            mensaje = recognizer.recognize\_google(audio,language="es")

            print(mensaje)

    except:

       pass

    return mensaje

def buscar\_wikipedia():

    try:

        sintesis("Hola, soy tu asistente de wikipedia, di lo que estas buscando")

        mensaje = reconocimiento()

        busqueda = mensaje

        wikipedia.set\_lang("es")

        respuesta = wikipedia.summary(busqueda, sentences = 1)

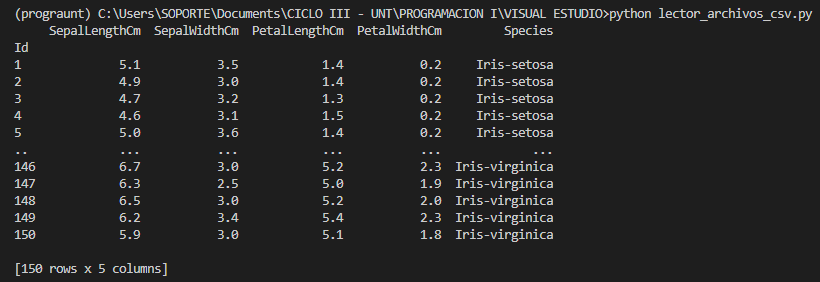
        sintesis(respuesta)

    except:

        sintesis("¡No se encontraron resultados para tu busqueda!")

buscar\_wikipedia()

### Ejercicio 3



### Ejercicio 4

## Desarrollo de test de comprobación

* 1. **Explique que es el archivo \_\_init\_\_.py**

\_\_init\_\_ es una convención en python, la cual siempre se debe crear en cada carpeta de donde queramos importar funciones, datos, o variables que se encuentran en un script diferente al que tenemos, además este se ejecuta implícitamente. \_\_init\_\_.py puede llegar a contener el mismo código Python que contiene cualquier otro módulo

* 1. **Explique qué significa la w, r, a, t, b, en el manejo de archivos.**

w: Abre el archivo para escribir nuevo contenido en reemplazo al anterior.

r: Abre el archivo como modo de lectura.

a: Abre el archivo para escribir nuevo contenido, pero a diferencia del “w”, este contenido es adicional al contenido anterior.

t: Guarda un archivo en formato texto con valor predeterminado

b: Guarda archivo en forma de binario.

* 1. **Investigue y explique que son los decoradores en python.**

## Recomendaciones

* Utilizar “a” en vez de “w” debido a que “w” sobrescribe los datos en el mismo dato y no genera más en el archivo csv.
* Para trabajar con el micrófono a la hora de la ejecución del asistente virtual, es muy importante que solo exista un micrófono conectado a la computadora, puesto que, al haber más, Python no reconocerá ninguno. En este caso, se tenía conectado el micrófono de audífonos y también el de la cámara, y al ejecutar el programa, ocurría que el programa se mantenía escuchando sin fin la búsqueda ya que no encontraba respuesta nuestra porque no se aceptaba ningún micrófono. Para solucionar esto, desconectamos la cámara (contenía micrófono incluido) y el programa ya reconocía nuestra búsqueda dicha por voz y respondía.

## Conclusiones

* La facilidad del lenguaje Python para la instalación de módulos o librerías, supone una gran multiplicación en la variedad de códigos que se pueden crear usando este lenguaje. Por consiguiente, aborda más campos y universaliza el desarrollo de la programación.
* E.

# REFERENCIAS BIBLIOGRÁFICAS

ACERVO LIMA. (15 de Abril de 2021). Diferencia entre lista y matriz en Python. Obtenido de ACERVO LIMA: https://es.acervolima.com/diferencia-entre-lista-y-matriz-en-python/

Disco Duro de Roer. (1 de Diciembre de 2013). Arreglos O Arrays En Pseudocódigo. Obtenido de Disco Duro de Roer: https://www.discoduroderoer.es/arreglos-o-arrays-en-pseudocodigo/

Foundation, P. S. (12 de Julio de 2022). Documentación de python. Obtenido de https://docs.python.org/es/3.10/contents.html

Line!, C. T. (2020, septiembre 30). Listas, Tuples, Sets, Strings y Diccionarios en PYTHON. https://www.youtube.com/watch?v=CCUNuqqn7PQ&t=392s

Samaniego, T. (2017, marzo 12). Tuplas & conjuntos en python. Mi Diario Python; dcaraballo. https://pythondiario.com/2017/03/tuplas-conjuntos-en-python.html

# ANEXOS

## Código completo del ejercicio 1:

DatosEnCsv.py:

from GUI import GUI

from tkinter import messagebox

try:

    GUI()

except:

    titulo="ERROR"

    mensaje="No se pudo ejecutar"

    messagebox.showerror(titulo,mensaje)

GUI.py

import tkinter as tk

import csv

import pyttsx3

def GUI():

    engine1 = pyttsx3.init()

    engine1.say("¿Que deseas realizar? ")

    engine1.runAndWait()

    preguntar = tk.Tk()

    preguntar.title("CSV")

    preguntar.geometry("240x250+700+280")

    preguntar.resizable(width=False, height=False)

    preguntar.iconbitmap("pregunta.ico")

    preguntar.config(background="khaki1")

    Label=tk.Label(preguntar,

    text="Elija una opción",

    background="khaki1",

    font=("Arial", 18, "bold"),

    foreground="black"

    )

    Label.pack(pady=25)

    def entrys\_llenar():

        ventana2=tk.Tk()

        ventana2.title("DATOS CSV")

        ventana2.geometry("500x300+500+250")

        ventana2.resizable(width=True, height=True)

        ventana2.iconbitmap("pregunta.ico")

        ventana2.config(background="linen")

        Label2=tk.Label(ventana2,

            text="Apellido Paterno:",

            font=("Arial", 16),

            background="linen",

            foreground="black"

        )

        Label2.grid(column=0, row=0, padx=10, pady=10)

        Label3=tk.Label(ventana2,

            text="Apellido Materno:",

            font=("Arial", 16),

            background="linen",

            foreground="black"

        )

        Label3.grid(column=0, row=1, padx=10, pady=10)

        Label4=tk.Label(ventana2,

            text="Nombre: ",

            font=("Arial", 16),

            background="linen",

            foreground="black"

        )

        Label4.grid(column=0, row=2, padx=10, pady=10)

        Entry2=tk.Entry(ventana2,

            font=("Arial", 16),

            foreground="black"

        )

        Entry2.grid(column=1, row=0, padx=10, pady=10,columnspan=2)

        Entry3=tk.Entry(ventana2,

            font=("Arial", 16),

            foreground="black"

        )

        Entry3.grid(column=1, row=1, padx=10, pady=10,columnspan=2)

        Entry4=tk.Entry(ventana2,

            font=("Arial", 16),

            foreground="black"

        )

        Entry4.grid(column=1, row=2, padx=10, pady=10,columnspan=2)

        def llenar():

            lista=[]

            a=Entry2.get()

            b=Entry3.get()

            c=Entry4.get()

            lista.append(a)

            lista.append(b)

            lista.append(c)

            writer=csv.writer(open("datos.csv", "a", newline=""))

            writer.writerow(lista)

            lista.clear()

            engine2 = pyttsx3.init()

            engine2.say("¿Desea agregar a otra persona? sino, solo cierre la ventana y visualice los datos ")

            engine2.runAndWait()

        botoningresar=tk.Button(ventana2,

            text="INGRESAR DATOS",

            font=("Arial", 16),

            background="honeydew2",

            activebackground="honeydew3",

            command=llenar

        )

        botoningresar.grid(column=2, row=4)

        ventana2.mainloop()

    boton1 = tk.Button(preguntar,

    text = "Registrar mis datos",

    font = ("Times", 14),

    background="orangered2",

    activebackground="orangered4",

    command=entrys\_llenar

    )

    boton1.pack(pady=10)

    def consultar():

        reader=csv.reader(open("datos.csv", "r"))

        for row in reader:

            print("Apellido Paterno: {0}, Apellido Materno: {1}, Nombre: {2}".format(row[0], row[1], row[2]))

    boton2 = tk.Button(preguntar,

    text = "Ver todos los datos",

    font = ("Times", 14),

    background="green2",

    activebackground="green4",

    command=consultar

    )

    boton2.pack(pady=10)

    preguntar.mainloop()

## Código completo del ejercicio 2:

import pyttsx3

import speech\_recognition

import wikipedia

engine = pyttsx3.init()

recognizer = speech\_recognition.Recognizer()

def sintesis(text):

 engine.say(text)

 engine.runAndWait()

def reconocimiento():

    try:

        with speech\_recognition.Microphone() as source:

            print("Escuchando... ")

            audio = recognizer.listen(source)

            mensaje = recognizer.recognize\_google(audio,language="es")

            print(mensaje)

    except:

       pass

    return mensaje

def buscar\_wikipedia():

    try:

        sintesis("Hola, soy tu asistente de wikipedia, di lo que estas buscando")

        mensaje = reconocimiento()

        busqueda = mensaje

        wikipedia.set\_lang("es")

        respuesta = wikipedia.summary(busqueda, sentences = 1)

        sintesis(respuesta)

    except:

        sintesis("¡No se encontraron resultados para tu busqueda!")

buscar\_wikipedia()

## Código completo del ejercicio 3:

import csv

import pandas as pd

df = pd.read\_csv("Iris.csv", index\_col = "Id")

print(df)

## Código completo del ejercicio 4: