UNIVERSIDAD SANTO TOMAS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
EDUCACIÓN ABIERTA Y A DISTANCIA
VIGILADA MINEDUCACIÓN

ACREDITACIÓN INSTITUCIONAL DE ALTA CALIDAD MULTICAMPUS
Res. MEN No. 01456 del 29 de enero de 2016
Vigencia por seis años

# Programación Avanzada

## Momento 4 - Evaluación Final

**2022-2**

**Danielmer Solis Arrieta**

**Código: 2251635**

**Universidad Santo Tomás**

**Vicerrectoría de Universidad Abierta y a Distancia**

**Ingeniería en Informática**

**Centro de Atención Universitario Barranquilla**

**2022**

**Contenido**

## 1. Introducción

Con el estudio de este espacio académico, el estudiante estará en la capacidad de comprender los conceptos del paradigma de programación orientada a objetos, manejo de archivo y ficheros, excepciones, documentación, módulos y paquetes. De igual manera, el estudiante estará en la capacidad desarrollar aplicaciones de inteligencia artificial para la automatización de tareas.

## 2. Objetivos

### 2.1 Objetivos generales.

Resolver lo planteado en el aula virtual de programación avanzada para la evaluación en línea, en el cuarto momento evaluativo.

### 2.2 Objetivos específicos.

2.2.1 Fortalecer la capacidad del estudiante para adaptarse a nuevos conceptos de programación.

2.2.2 Fundamentar en el estudiante la cualidad de soportar de manera teórica y práctica los programas de software que se desarrollan.

2.2.3 Documentar los programas y detallar teóricamente los conceptos desarrollados durante la práctica.

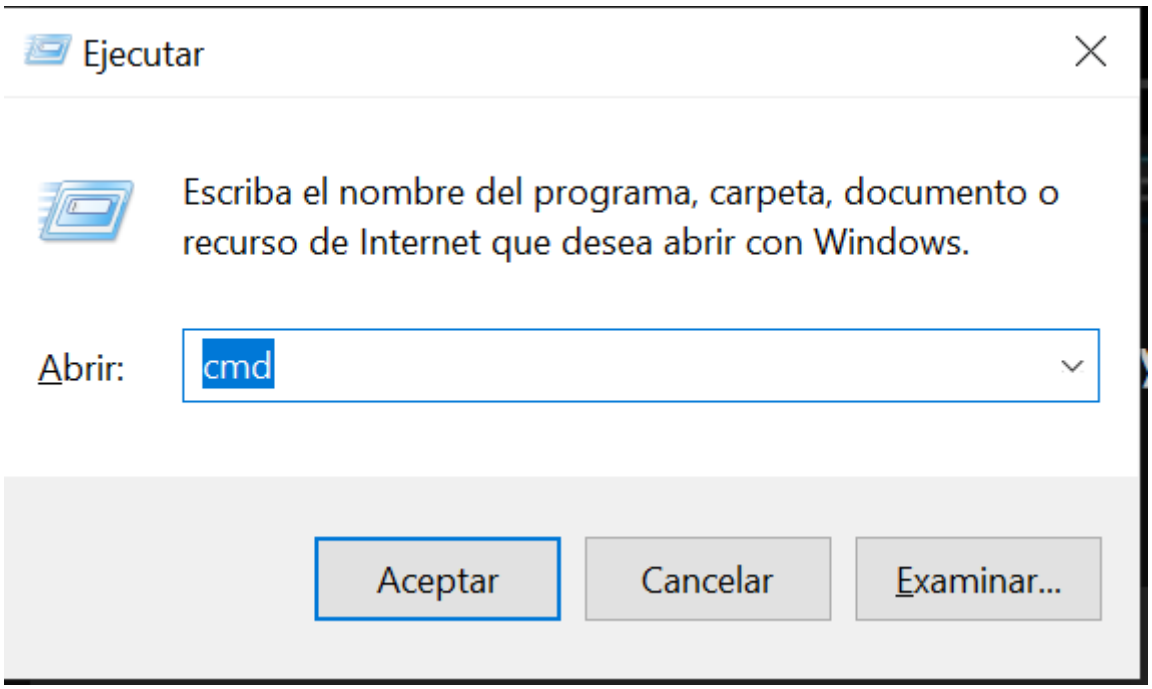
2.2.4 Realizar y documentar las diferentes pruebas de software usando notaciones específicas.

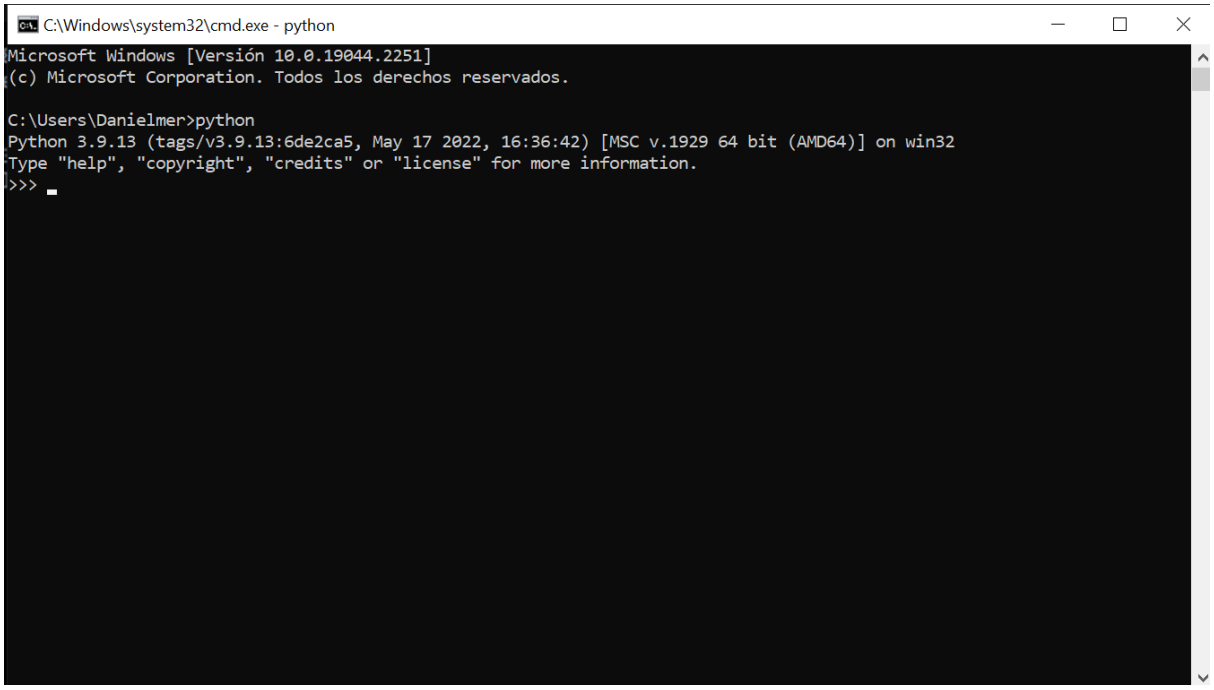
## 3. Actividades a desarrollar.

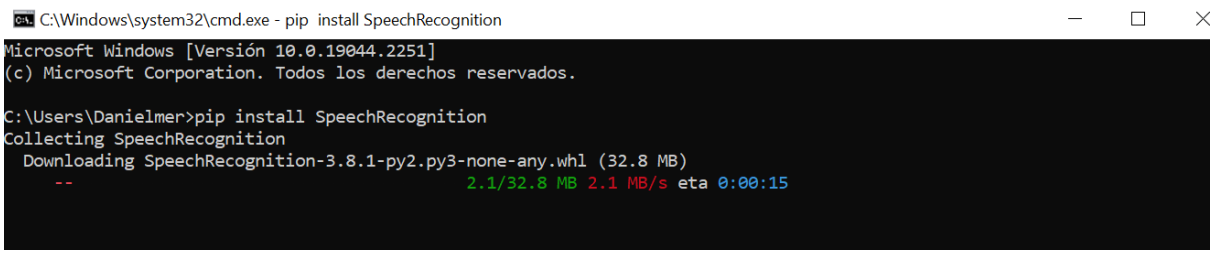
Utilizando el lenguaje de programación Python y un ambiente virtual, instale las librerías que considere necesarias.

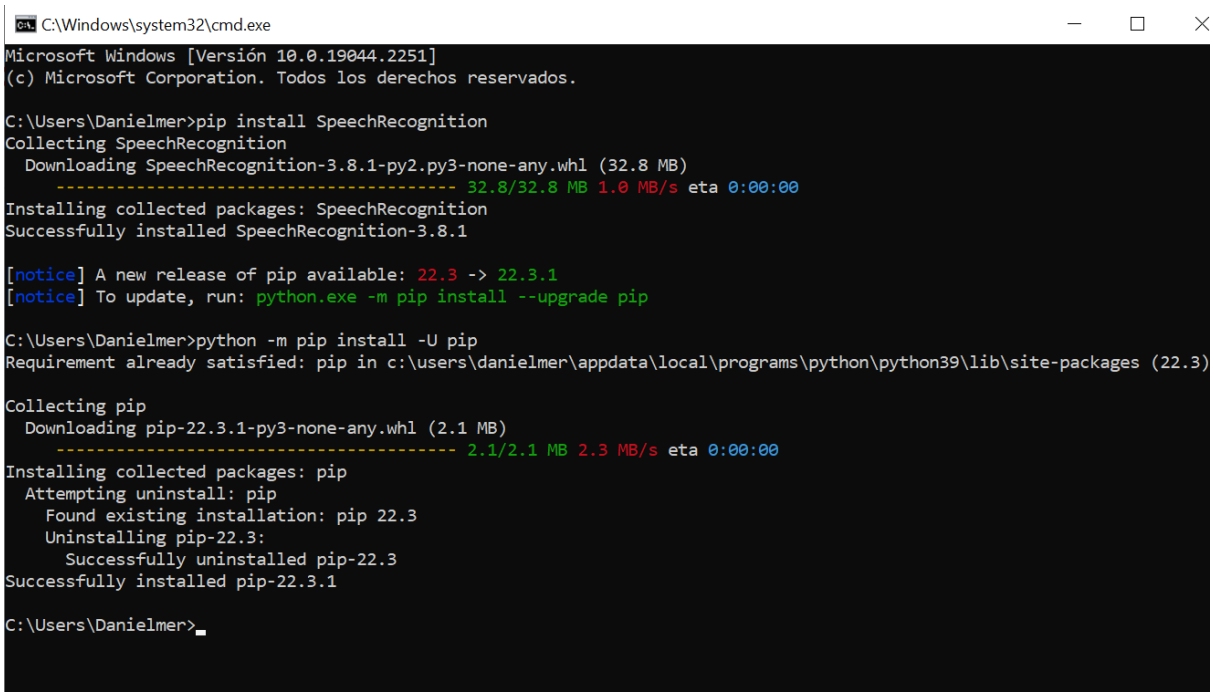
**3.1 Escriba un programa con la funcionalidad de un asistente virtual y cumpla con los siguientes requerimientos:**

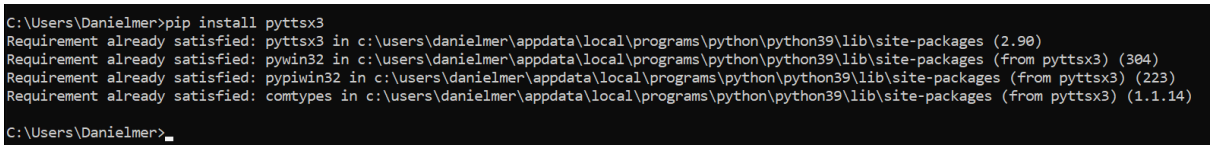
- El asistente virtual debe tener un nombre a través del cual se llama y recibe órdenes.
- El asistente virtual debe reconocer comandos por voz y convertirlos a texto para su posterior procesamiento.
- El asistente virtual debe convertir texto a voz.
- El asistente virtual debe reproducir un video en YouTube.
- El asistente virtual debe responder cuando se le pregunte por la hora actual.
- El asistente virtual debe buscar cualquier información en Wikipedia.
- El asistente virtual debe abrir la página de Google.
- El asistente virtual debe enviar un mensaje de correo electrónico.
- El asistente virtual debe tomar una foto.




C:\Windows\system32\cmd.exe - python
Microsoft Windows [Versión 10.0.19044.2251]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Users\Danielmer>python
Python 3.9.13 (tags/v3.9.13:6de2ca5, May 17 2022, 16:36:42) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>


C:\Windows\system32\cmd.exe - pip install SpeechRecognition
Microsoft Windows [Versión 10.0.19044.2251]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Users\Danielmer>pip install SpeechRecognition
Collecting SpeechRecognition
Downloading SpeechRecognition-3.8.1-py2.py3-none-any.whl (32.8 MB)
2.1/32.8 MB 2.1 MB/s eta 0:00:15


C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 10.0.19044.2251]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Users\Danielmer>pip install SpeechRecognition
Collecting SpeechRecognition
Downloading SpeechRecognition-3.8.1-py2.py3-none-any.whl (32.8 MB)
32.8/32.8 MB 1.0 MB/s eta 0:00:00
Installing collected packages: SpeechRecognition
Successfully installed SpeechRecognition-3.8.1
[notice] A new release of pip available: 22.3 -> 22.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
C:\Users\Danielmer>python -m pip install -U pip
Requirement already satisfied: pip in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (22.3)
Collecting pip
Downloading pip-22.3.1-py3-none-any.whl (2.1 MB)
2.1/2.1 MB 2.3 MB/s eta 0:00:00
Installing collected packages: pip
Attempting uninstall: pip
Found existing installation: pip 22.3
Uninstalling pip-22.3:
Successfully uninstalled pip-22.3
Successfully installed pip-22.3.1
C:\Users\Danielmer>


C:\Users\Danielmer>pip install pyttsx3
Requirement already satisfied: pyttsx3 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (2.90)
Requirement already satisfied: pywin32 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from pyttsx3) (304)
Requirement already satisfied: pypiwin32 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from pyttsx3) (223)
Requirement already satisfied: comtypes in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from pyttsx3) (1.1.14)
C:\Users\Danielmer>

```
C:\Users\Danielmer>pip install PyAudio
Collecting PyAudio
  Downloading PyAudio-0.2.12-cp39-cp39-win_amd64.whl (163 kB)
     ---------------------------------------- 164.0/164.0 kB 3.3 MB/s eta 0:00:00
Installing collected packages: PyAudio
Successfully installed PyAudio-0.2.12

C:\Users\Danielmer>
```

```
C:\Users\Danielmer>pip install pywhatkit
Collecting pywhatkit
  Downloading pywhatkit-5.4-py3-none-any.whl (15 kB)
Requirement already satisfied: Pillow in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from pywhatkit) (9.2.0)
Collecting Flask
  Downloading Flask-2.2.2-py3-none-any.whl (101 kB)
     ---------------------------------------- 101.5/101.5 kB 1.9 MB/s eta 0:00:00
Collecting wikipedia
  Downloading wikipedia-1.4.0.tar.gz (27 kB)
  Preparing metadata (setup.py) ... done
Collecting pyautogui
  Downloading PyAutoGUI-0.9.53.tar.gz (59 kB)
     ---------------------------------------- 59.0/59.0 kB 3.3 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: requests in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from pywhatkit) (2.28.1)
Collecting Werkzeug>=2.2.2
  Using cached Werkzeug-2.2.2-py3-none-any.whl (232 kB)
Collecting importlib-metadata>=3.6.0
  Downloading importlib_metadata-5.0.0-py3-none-any.whl (21 kB)
Collecting Jinja2>=3.0
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
     ---------------------------------------- 133.1/133.1 kB 4.0 MB/s eta 0:00:00
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Requirement already satisfied: click>=8.0 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from Flask->pywhatkit) (8.1.3)
Collecting pymsgbox
  Downloading PyMsgBox-1.0.9.tar.gz (18 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting PyTweening>=1.0.1
  Downloading pytweening-1.0.4.tar.gz (14 kB)
  Preparing metadata (setup.py) ... done
Collecting pyscreeze>=0.1.21
```

```
Collecting PyTweening>=1.0.1
  Downloading pytweening-1.0.4.tar.gz (14 kB)
  Preparing metadata (setup.py) ... done
Collecting pyscreeze>=0.1.21
  Downloading PyScreeze-0.1.28.tar.gz (25 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting pygetwindow>=0.0.5
  Downloading PyGetWindow-0.0.9.tar.gz (9.7 kB)
  Preparing metadata (setup.py) ... done
Collecting mouseinfo
  Downloading MouseInfo-0.1.3.tar.gz (10 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from requests->pywhatkit) (1.26.12)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from requests->pywhatkit) (2022.9.24)
Requirement already satisfied: idna<4,>=2.5 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from requests->pywhatkit) (3.4)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from requests->pywhatkit) (2.1.1)
Collecting beautifulsoup4
  Downloading beautifulsoup4-4.11.1-py3-none-any.whl (128 kB)
     ---------------------------------------- 128.2/128.2 kB ? eta 0:00:00
Requirement already satisfied: colorama in c:\users\danielmer\appdata\local\programs\python\python39\lib\site-packages (from click>=8.0->Flask->pywhatkit) (0.4.5)
Collecting zipp>=0.5
  Downloading zipp-3.10.0-py3-none-any.whl (6.2 kB)
Collecting MarkupSafe>=2.0
  Using cached MarkupSafe-2.1.1-cp39-cp39-win_amd64.whl (17 kB)
Collecting pyrect
  Downloading PyRect-0.2.0.tar.gz (17 kB)
  Preparing metadata (setup.py) ... done
Collecting soupsieve>1.2
  Downloading soupsieve-2.3.2.post1-py3-none-any.whl (37 kB)
Collecting pyperclip
  Downloading pyperclip-1.8.2.tar.gz (20 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: pyscreeze, pymsgbox
  Building wheel for pyscreeze (pyproject.toml) ... done
  Created wheel for pyscreeze: filename=PyScreeze-0.1.28-py3-none-any.whl size=13027 sha256=7b332c7e2688348dc9f484dd0983b9c08c1ca479285ef8cee03e1f37c996aea3
  Stored in directory: c:\users\danielmer\appdata\local\pip\cache\wheels\b2\cb\35\3998fb8c95724397e2967b697522ba660bbdf62b12d1da2d9a
  Building wheel for pymsgbox (pyproject.toml) ... done
  Created wheel for pymsgbox: filename=PyMsgBox-1.0.9-py3-none-any.whl size=7416 sha256=6f4194e53339c394d861b7b3b11c6fcc14b3fcff8a7f6d46024f69ba82b914d6
  Stored in directory: c:\users\danielmer\appdata\local\pip\cache\wheels\ec\83\a6\df8264d6d5c838598e8f94b73854c7abb718bf1987c18605dd
Successfully built pyscreeze pymsgbox
Installing collected packages: PyTweening, pyscreeze, pyrect, pyperclip, pymsgbox, zipp, soupsieve, pygetwindow, mouseinfo, MarkupSafe, itsdangerous, Werkzeug, pyautogui, Jinja2, importlib-metadata, beautifulsoup4, wikipedia, Flas
k, pywhatkit
```

```
Successfully built pyscreeze pymsgbox
Installing collected packages: PyTweening, pyscreeze, pyrect, pyperclip, pymsgbox, zipp, soupsieve, pygetwindow, mouseinfo, MarkupSafe, itsdangerous, Werkzeug, pyautogui, Jinja2, importlib-metadata, beautifulsoup4, wikipedia, Flas
k, pywhatkit
  DEPRECATION: PyTweening is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible
 replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for PyTweening ... done
  DEPRECATION: pyrect is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible rep
lacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for pyrect ... done
  DEPRECATION: pyperclip is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible
replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for pyperclip ... done
  DEPRECATION: pygetwindow is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possibl
e replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for pygetwindow ... done
  DEPRECATION: mouseinfo is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible
replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for mouseinfo ... done
  DEPRECATION: pyautogui is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible
replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for pyautogui ... done
  DEPRECATION: wikipedia is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible
replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for wikipedia ... done
Successfully installed Flask-2.2.2 Jinja2-3.1.2 MarkupSafe-2.1.1 PyTweening-1.0.4 Werkzeug-2.2.2 beautifulsoup4-4.11.1 importlib-metadata-5.0.0 itsdangerous-2.1.2 mouseinfo-0.1.3 pyautogui-0.9.53 pygetwindow-0.0.9 pymsgbox-1.0.9 p
yperclip-1.8.2 pyrect-0.2.0 pyscreeze-0.1.28 pywhatkit-5.4 soupsieve-2.3.2.post1 wikipedia-1.4.0 zipp-3.10.0

C:\Users\Danielmer>
```


Código:

```
import speech_recognition as sr
import pyttsx3
import pywhatkit
import pyjokes
import datetime
import webbrowser
import os
import wikipedia
import cv2
import uuid
import pickle
import os
from google_auth_oauthlib.flow import flow, InstalledAppFlow
from googleapiclient.discovery import build
from googleapiclient.http import MediaFileUpload, MediaToBaseDownLoad
from google.auth.transport.requests import Request

horas_invertidas = 20;

name = 'DaddyBeto'

listener = sr.Recognizer()

engine = pyttsx3.init()

voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)
wikipedia.set_lang("es")

def talk(text):
    engine.say(text)
    engine.runAndWait()

def listen(texto):
    try:
        with sr.Microphone() as source:
            print(texto)
            voice = listener.listen(source)
```

```
            rec = listener.recognize_google(voice, language='es-ES')
            rec = rec.lower()
            if name in rec:
                rec = rec.replace(name,'')
                print("Usted dijo: " +
rec)
    except:
        pass

    return rec


def run():
    #Música y Videos en YT
    rec = listen('Esperando ordenes...')
    if 'reproduce' in rec:
        music = rec.replace('reproduce', '')
        talk('Reproduciendo '+ music)
        pywhatkit.playonyt(music)

    #Hora
    elif 'dime la hora actual' in rec:
        hora = datetime.datetime.now().strftime('%I:%M %p')
        talk("Son las " + hora)

    # BUSCA EN WIKIPEDIA
    elif 'busca en wikipedia' in recognizer:
        consulta = recognizer.replace('busca en wikipedia', '')
        talk('buscando en wikipedia' + consulta)
        resultado = wikipedia.summary(consulta, sentences=3)
        talk(resultado)

    #Buscador
    elif 'busca' in rec:
        order = rec.replace('busca', '')
        talk('Buscando '+ order)
        pywhatkit.search(order)

 # BUSCA EN GOOGLE
    elif 'busca en google' in recognizer:
        consulta = recognizer.replace('busca en google', '')
        talk('Buscando en google' + consulta)
        pywhatkit.search(consulta)

    #Chistes
    elif 'dime un chiste' in rec:
        talk(pyjokes.get_joke('es'))
```

```
    elif 'créditos' in rec:
        webbrowser.open('https://www.youtube.com/watch?v=AOamtC2_r7k')


    #Ejecución de aplicaciones.exe
    elif 'ejecuta' in rec:
        order = rec.replace('ejecuta','')
        talk('Abriendo '+ order)
        app = order+'.exe'
        os.system(app)

    #Creación de archivos de texto
    elif 'crea el archivo' in rec:
        order = rec.replace('crea el archivo','')
        order = order+'.txt'
        if os.path.exists(order):
            talk("El archivo ya existe")

        else:
            archivo = open(order,"w")
            archivo.close()
            talk("Se creo el archivo correctamente")

    #Eliminación de archivos de texto
    elif 'borra el archivo' in rec:
        order = rec.replace('borra el archivo','')
        order = order+'.txt'
        if os.path.exists(order):
            os.remove(order)
            talk("Se elimino el archivo correctamente")
        else:
            talk("El archivo no existe")

    else:
        talk("No te entendi muy bien, vuelve a intentarlo")

cap = cv2.VideoCapture(0)

leido, frame = cap.read()

if leido == True:
    nombre_foto = str(uuid.uuid4()) + ".png" # uuid4 regresa un objeto, no una
cadena. Por eso lo convertimos
    cv2.imwrite(nombre_foto, frame)
    print("Foto tomada correctamente con el nombre {}".format(nombre_foto))
else:
    print("Error al acceder a la cámara")

"""
```

```
    Finalmente liberamos o soltamos la cámara
"""
cap.release()

from google import Create_Services
import base64
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

cliente = "trchatbot.json"
API_NAME = "gmail"
API_VERSION = "v1"
SCOPES = ["https://mail.google.com"]

service = Create_Services(cliente, API_NAME, API_VERSION, SCOPES)

mimeMessage["subject"] = "Evaluacion final programacion avanzada"
emailMsg = "Buen dia, este es mi trabajo"
mimeMessage["to"] = "danielmersolis@ustadistancia.edu.co"
mimeMessage = MIMEMultipart()

mimeMessage.attach(MIMEText(emailMsg, "plain"))

raw_string = base64.urlsafe_b64decode(mimeMessage.as_bytes().decode)

message = service.users().messages().send(userId = "Me", body =
{"raw":raw_string}).execute()
print(message)

#Iniciador
if __name__ == "__main__":
    run()
```

### 3.2 Manual de usuario del asistente virtual.

Esta aplicación, que en esta ocasión es un asistente virtual el cual implementa inteligencia artificial, dicho asistente virtual llamado como DaddyBeto podrá reconocer la voz humana, y en el caso presente de esta aplicación en base a las instrucciones que se le otorgaron podrá realizar o cumplir con ciertas funciones, en este caso en particular explicare brevemente el como utilizar este asistente virtual DaddyBeto.

En el funcionamiento encontraremos lo siguiente:

- El sistema detecta las palabras que la persona emita.
- Convierte estas palabras en un formato que sea legible por la máquina.
- Dependiendo del mensaje emitido, el asistente procederá a ejecutarse, dicha reacción de la maquina puede ser la ejecución de una orden, ofrecer una respuesta, enviar mensajes, tomar fotos, reproducir un video en YouTube, ejecutar una aplicación, buscar una información en Wikipedia e indicarnos la hora.

Podremos hacer uso de este asistente virtual mediante la ejecución del mismo en el ordenador.

### 3.3 Manual técnico del asistente virtual.

Implementación en código y explicación de las funcionalidades desarrolladas dentro de este asistente virtual.

Las librerías empleadas dentro del código del asistente virtual. Encontraremos la librería "**speech_recognition**" que es el reconocedor de voz de nuestro asistente virtual. También esta el "pyttsx3" el cual nos va a permitir enviar las respuestas por sonido, las cuales el asistente virtual generara, que en este caso será como el modulo del habla. El "**pywhatkit**" que en este asistente virtual ayuda para lograr la reproducción de videos de YouTube. El "**datetime**" que servirá para indicarnos la hora actual. El "**webbrowser**" que servirá para abrir una pagina web en el navegador. La "**os**" permitirá acceder a funcionalidades dependientes del sistema operativo y finalmente "**wikipedia**" que nos servirá para buscar cualquier información en Wikipedia.

Generamos el motor "**engine = pyttsx3.init()**" para implementar el speech del asistente virtual, darle el nombre "**name = 'DaddyBeto'**" , "**def talk(text):**" el cual permite que el asistente hable, "**def run():**

**#Música y Videos en YT**

**rec = listen('Esperando ordenes...')**

**if 'reproduce' in rec:**

**music = rec.replace('reproduce', '')**

**talk('Reproduciendo '+ music)**

**pywhatkit.playonyt(music)**" para la música y videos de YouTube,

"**elif 'dime la hora actual' in rec:**

**hora = datetime.datetime.now().strftime('%I:%M %p')**

**talk("Son las " + hora)**" para la hora, "**elif 'busca en wikipedia' in recognizer:**

**consulta = recognizer.replace('busca en wikipedia', '')**

**talk('buscando en wikipedia' + consulta)**

**resultado = wikipedia.summary(consulta, sentences=3)**

**talk(resultado)**” para la búsqueda en Wikipedia, “**elif 'busca en google' in recognizer:**

**consulta = recognizer.replace('busca en google', '')**

**talk('Buscando en google' + consulta)**

**pywhatkit.search(consulta)**” para la búsqueda en Google, “**elif 'crea el archivo' in rec:**

**order = rec.replace('crea el archivo','')**

**order = order+'.txt'**

**if os.path.exists(order):**

**talk("El archivo ya existe")**

**else:**

**archivo = open(order,"w")**

**archivo.close()**

**talk("Se creo el archivo correctamente")**” para la creación de archivos de texto, “**cap = cv2.VideoCapture(0)**

**leido, frame = cap.read()**

**if leido == True:**

**nombre_foto = str(uuid.uuid4()) + ".png" # uuid4 regresa un objeto, no una cadena. Por eso lo convertimos**

**cv2.imwrite(nombre_foto, frame)**

```
    print("Foto tomada correctamente con el nombre {}".format(nombre_foto))
else:
    print("Error al acceder a la cámara")

"""
    Finalmente liberamos o soltamos la cámara
"""
```

**cap.release()**” para la captura de fotos y finalmente “**from google import Create_Services**

```
import base64
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

cliente = "trchatbot.json"
API_NAME = "gmail"
API_VERSION = "v1"
SCOPES = ["https://mail.google.com"]

service = Create_Services(cliente, API_NAME, API_VERSION, SCOPES)

mimeMessage["subject"] = "Evaluacion final programacion avanzada"
emailMsg = "Buen dia, este es mi trabajo"
mimeMessage["to"] = "danielmersolis@ustadistancia.edu.co"
mimeMessage = MIMEMultipart()

mimeMessage.attach(MIMEText(emailMsg, "plain"))
```

**raw_string = base64.urlsafe_b64decode(mimeMessage.as_bytes().decode)**

**message = service.users().messages().send(userId = "Me", body = {"raw":raw_string}).execute()**

**print(message)**" para el envió de correo electrónico.

## 4. Conclusión

Como hemos podido ver, se ha podido dar solución a lo presentado en el aula virtual de programación avanzada para la entrega del momento 4 de la evaluación en línea.

## 5. Referencias bibliográficas

Hinojosa Gutiérrez, Á. (2015). Python paso a paso. RA-MA Editorial. https://elibro.net/es/lc/usta/titulos/107213.

Chacon, Scott, and Ben Straub. Pro Git, Apress L. P., 2014. ProQuest Ebook Central, https://ebookcentral.proquest.com/lib/bibliotecausta-ebooks/detail.action?docID=6422698.

Sneeringer, Luke. Professional Python, John Wiley & Sons, Incorporated, 2015. ProQuest Ebook Central, https://ebookcentral.proquest.com/lib/bibliotecausta-ebooks/detail.action?docID=4187169.

Enlace de la carpeta GitHub: https://github.com/DanielmerSolis/evaluacion_final.