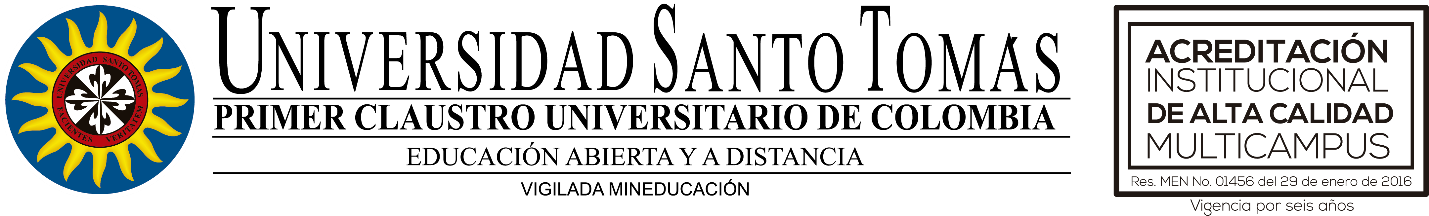
****

**Programación Avanzada**

**Momento 3 - 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 3](#_Toc119515110)

[2. Objetivos 4](#_Toc119515111)

[2.1 Objetivos generales. 4](#_Toc119515112)

[2.2 Objetivos específicos. 4](#_Toc119515113)

[3. Actividades a desarrollar. 5](#_Toc119515114)

[3.1 Escriba un programa con la funcionalidad de un asistente virtual y cumpla con los siguientes requerimientos: 5](#_Toc119515115)

[4. Conclusión 12](#_Toc119515116)

[5. Referencias bibliográficas 13](#_Toc119515117)

# 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.

# Objetivos

## Objetivos generales.

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

## Objetivos específicos.

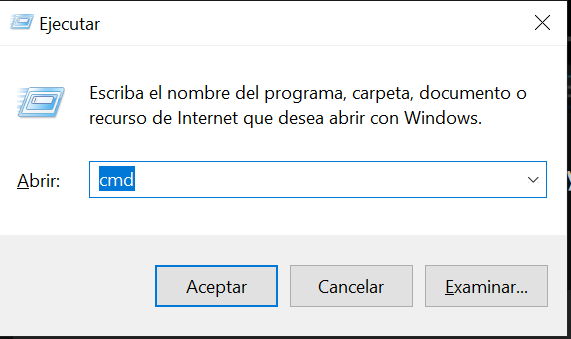
* + 1. Fortalecer la capacidad del estudiante para adaptarse a nuevos conceptos de programación.
    2. Fundamentar en el estudiante la cualidad de soportar de manera teórica y práctica los programas de software que se desarrollan.
    3. Documentar los programas y detallar teóricamente los conceptos desarrollados durante la práctica.
    4. Realizar y documentar las diferentes pruebas de software usando notaciones específicas.

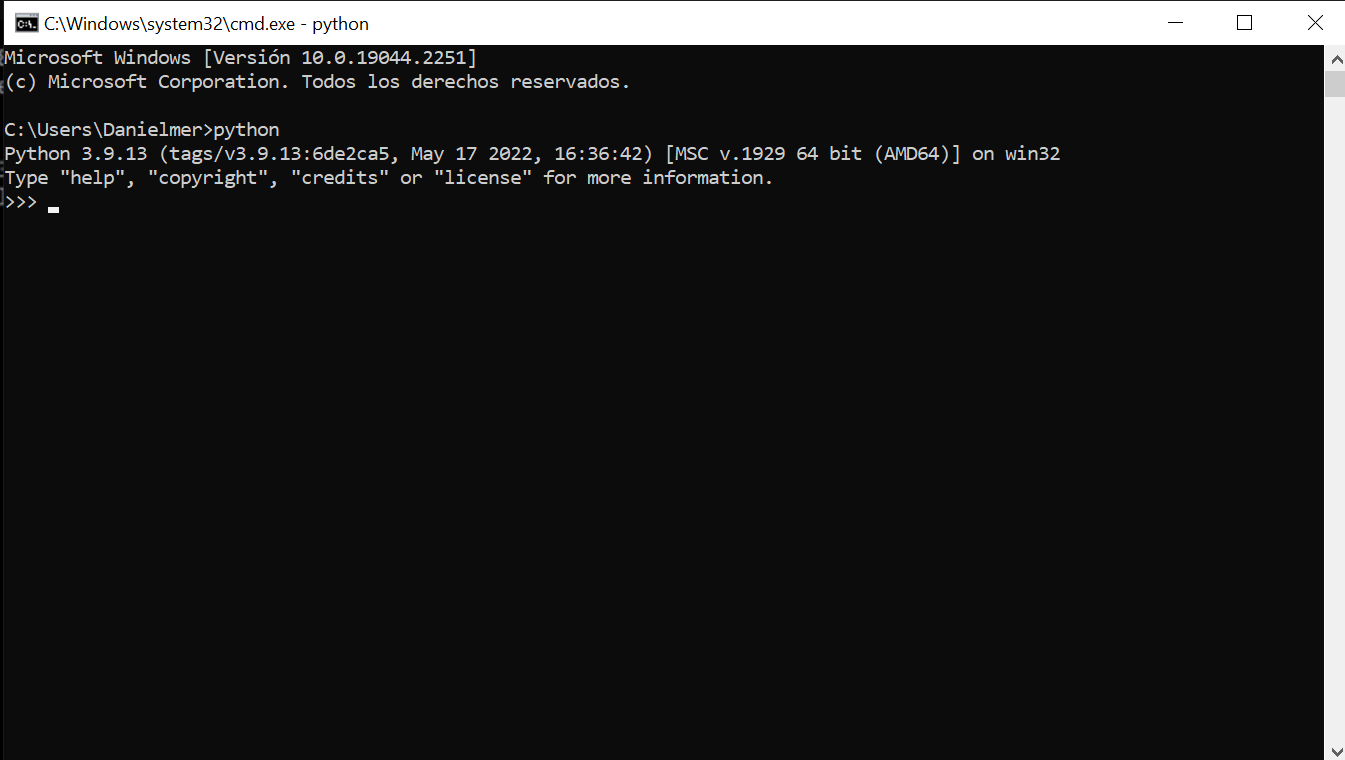
# Actividades a desarrollar.

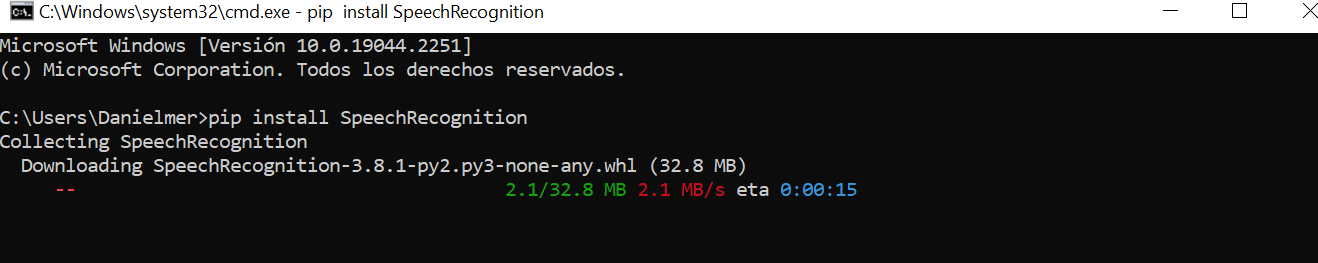
Utilizando el lenguaje de programación Python y un ambiente virtual, instale las librerías que considere necesarias.

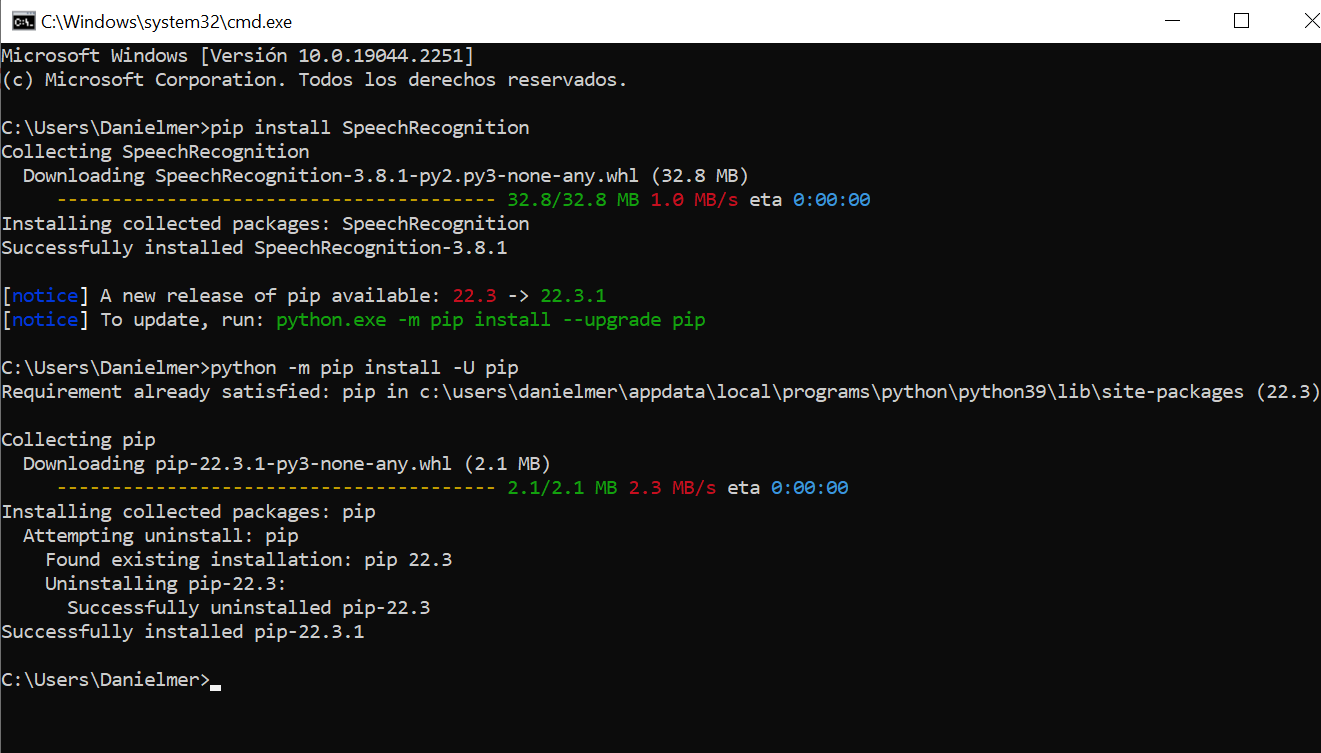
## Escriba un programa con la funcionalidad de un asistente virtual y cumpla con los siguientes requerimientos:

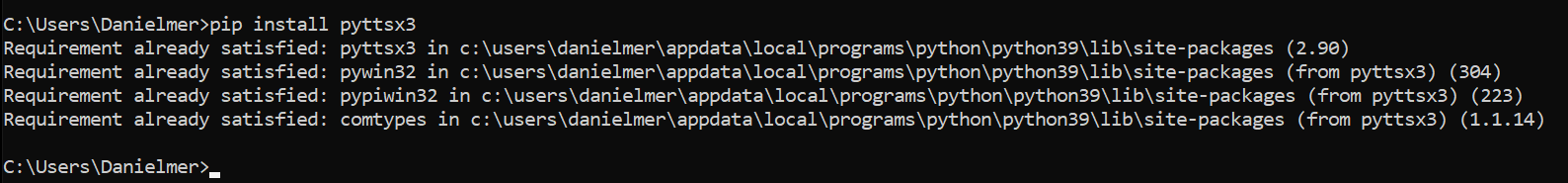
* El asistente virtual debe tener un nombre a través del cual se llama y recibe órdenes.
* El asistente virtual debe reconocer comandos por voz y convertirlos a texto para su posterior procesamiento.
* El asistente virtual debe convertir texto a voz.
* El asistente virtual debe reproducir un video en 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.

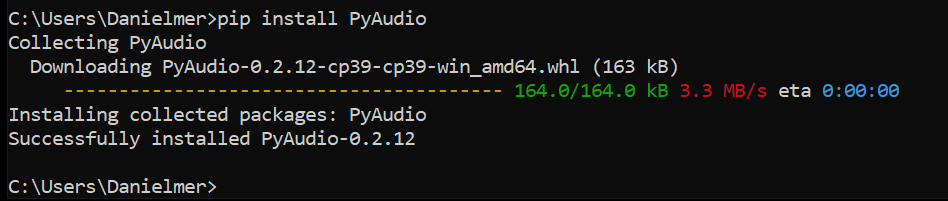


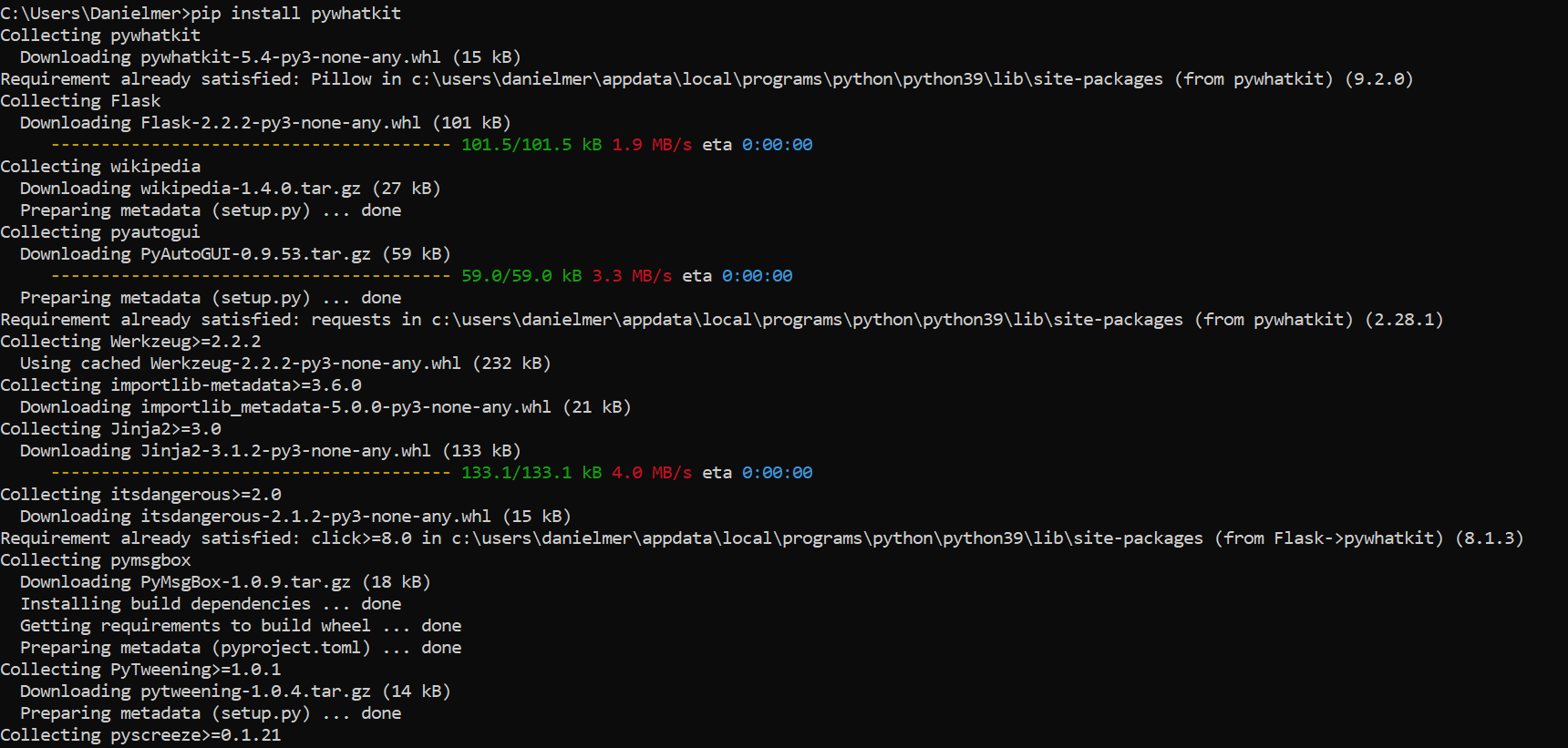


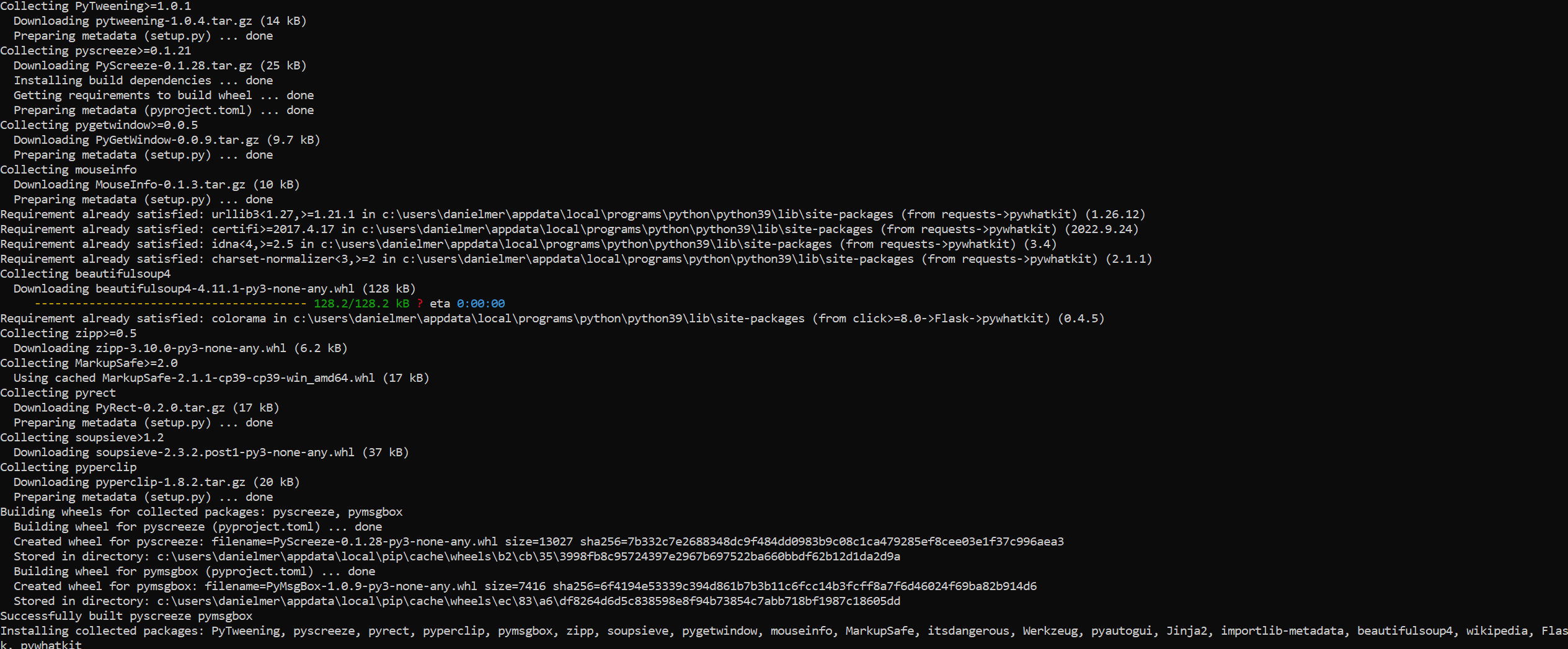


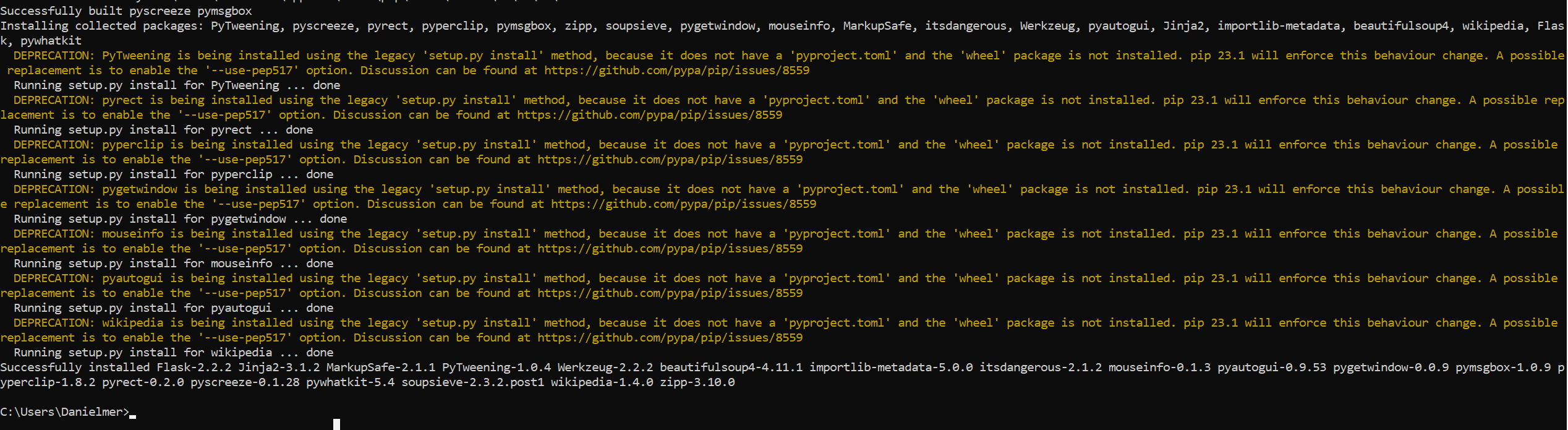


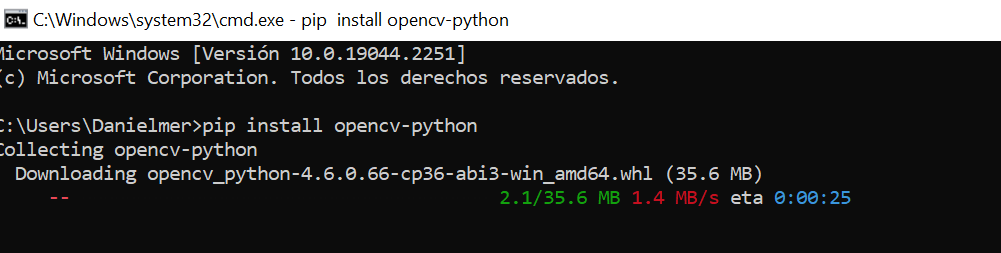












Código:

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()

# 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 3 de la evaluación en línea.

# 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.