JOSE ARTURO BUSTAMANTE LAZCANO

[nombre de la empresa]

INTELIGENCIA ARTIFICIAL

Alumno: Nombre Semestre Enero – Junio 2023

Contenido

[Semana 1 – Unidad 1 - Introducción a la Inteligencia Artificial. 2](#_Toc128724334)

[Contenido de la Sesión 1 - Evaluación Diagnostica. 2](#_Toc128724335)

[Semana 1 – Unidad 1 - Introducción a la Inteligencia Artificial. 3](#_Toc128724336)

[Contenido de la Sesión 2 - Instalación y configuración de Software. 3](#_Toc128724337)

[Semana 1 – Unidad 1 - Introducción a la Inteligencia Artificial. 5](#_Toc128724338)

[Contenido de la Sesión 3 - Pruebas y Ejecución de comandos CV2. 5](#_Toc128724339)

[Semana 2 – Unidad 1 - Introducción a la Inteligencia Artificial. 6](#_Toc128724340)

[Contenido de la Sesión 4 -5 - El proceso de razonamiento según la lógica (Axiomas, Teoremas, demostración). 6](#_Toc128724341)

[Semana 3 – Unidad 1 - Introducción a la Inteligencia Artificial. 7](#_Toc128724342)

[Contenido de la Sesión 6 - El modelo del agente inteligente, Sistemas Multi Agentes, Sistemas Ubicuos. 7](#_Toc128724343)

[Semana 4 – Unidad 1 - Introducción a la Inteligencia Artificial. 10](#_Toc128724344)

[Contenido de la Sesión 7 El papel de la heurística. 10](#_Toc128724345)

[Semana 4 – Unidad 1 - Introducción a la Inteligencia Artificial. 11](#_Toc128724346)

[Contenido de la Sesión 8 Aplicación con OPEn CV2. 11](#_Toc128724347)

[Semana 5 – Unidad 1 - Introducción a la Inteligencia Artificial. 12](#_Toc128724348)

[Contenido de la Sesión 9 Algoritmos de búsqueda local. 12](#_Toc128724349)

# Semana 1 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 1 - Evaluación Diagnostica.

1.- ¿Qué crees que sea la inteligencia artificial y qué crees que no sea?

2.- ¿Sabes el nombre de alguna inteligencia artificial y cual es de ser el caso?

3.- ¿Sabes que es una red neuronal?

4.- ¿Cómo crees que trabaja la inteligencia artificial?

5.- ¿Has escuchado hablar sobre la guerra de las máquinas y si es así que opinas?

Nombre:

Aciertos:

# Semana 1 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 2 - Instalación y configuración de Software.

1)Instalar python en especifico esta version

https://www.python.org/downloads/release/python-367/

Install the required build-tools (some might already be installed on your system).

$ sudo apt-get update

$ sudo apt-get install build-essential tk-dev libncurses5-dev libncursesw5-dev libreadline6-dev libdb5.3-dev libgdbm-dev libsqlite3-dev libssl-dev libbz2-dev libexpat1-dev liblzma-dev zlib1g-dev

wget https://www.python.org/ftp/python/3.6.0/Python-3.6.0.tgz

tar xzvf Python-3.6.0.tgz

cd Python-3.6.0/

./configure

make -j4

sudo make install

2)instalar desde la consola cmd ejecutada como administrador

pip install opencv-contrib-python

3)Comprovar las versiones instaladas

python 3.6

cv2 4.4.0 para eso ponemos un codigo simple

entramos al cmd ponemos python y damos enter

despues dentro ponemos import cv2 y enter

escribimos cv2.\_\_version\_\_ y enter

4)Instalación de tesseract para vision de caracteres

https://digi.bib.uni-mannheim.de/tesseract/tesseract-ocr-w64-setup-v5.0.0-alpha.20201127.exe

Manda un mensaje de precaución en windows damos mas opc y aceptamos

Para la instalación hay que agregar el idioma spanish, spanish old y matematico

5)instalar desde pip pytesseract desde consola cmd como administrador

pip install pytesseract

6)Nota cuando se ocupe hay que poner la ruta del tesseract

7)pip install mediapipe para mascarillas

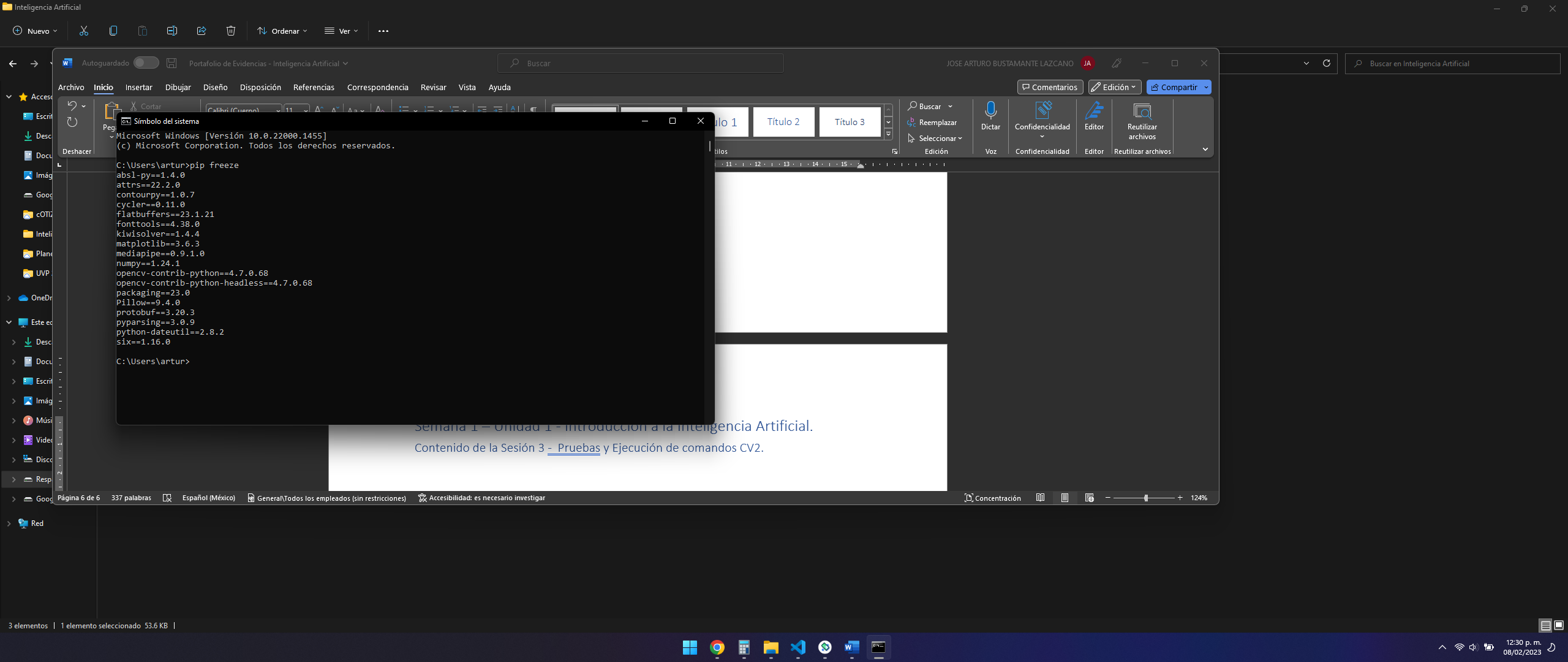
Entorno virtual -> Entorno aislado de instalaciones para

crear entorno de pyton con paquetes independientes

Opencv Snippets

# Semana 1 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 3 - Pruebas y Ejecución de comandos CV2.



# Semana 2 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 4 -5 - El proceso de razonamiento según la lógica (Axiomas, Teoremas, demostración).

El razonamiento lógico es un proceso que involucra la evaluación de argumentos y la deducción de conclusiones a partir de premisas. La lógica es la disciplina filosófica que estudia los métodos y principios de la inferencia y el razonamiento correcto.

En el razonamiento lógico, se parte de premisas o axiomas, que son afirmaciones consideradas verdaderas, y se llega a una conclusión a través de un proceso deductivo, en el que se aplican reglas lógicas a las premisas.

Este proceso debe ser riguroso y sólido, es decir, la conclusión debe seguir necesariamente de las premisas y no puede haber una falacia o error lógico en el razonamiento.

En resumen, el proceso de razonamiento lógico consiste en las siguientes etapas:

Identificación de las premisas y la conclusión

Aplicación de reglas lógicas a las premisas

Deducción de la conclusión

Verificación de la validez y coherencia del razonamiento

Es importante tener en cuenta que el razonamiento lógico no garantiza la verdad de las conclusiones, sólo su rigor y consistencia a partir de las premisas.

Un axioma es una afirmación o proposición que se considera verdadera sin necesidad de demostración. En lógica y matemáticas, los axiomas son los principios básicos a partir de los cuales se construyen los demás argumentos y teoremas.

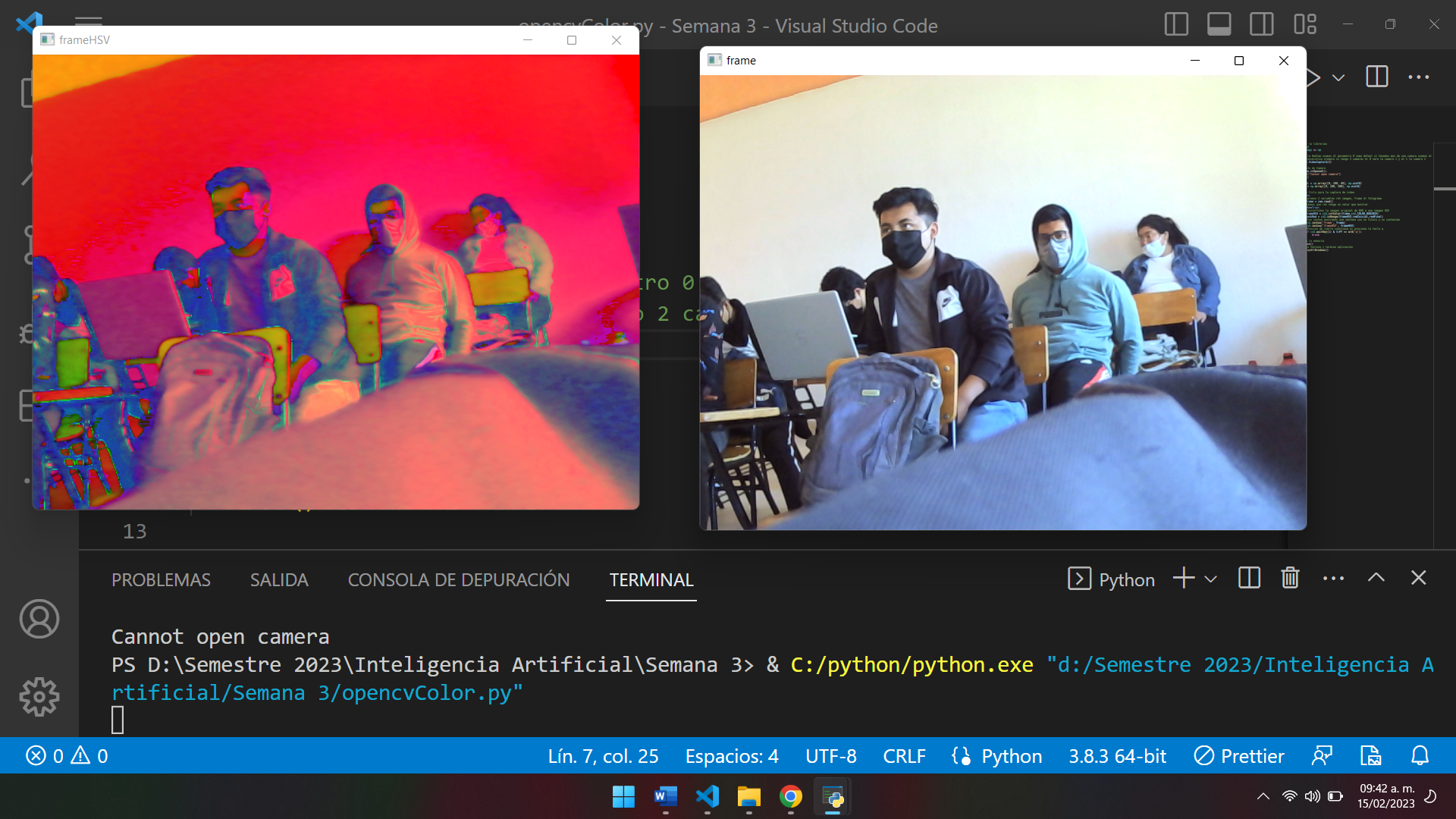
Los axiomas son considerados como verdaderos de manera previa y se utilizan como base para el razonamiento y la deducción de otras proposiciones. Por lo tanto, los axiomas son esenciales en la construcción de un sistema lógico o matemático.

Es importante tener en cuenta que los axiomas no se pueden demostrar, ya que se consideran verdaderos por definición. Por esta razón, es fundamental seleccionar cuidadosamente los axiomas de un sistema para asegurarse de que sean coherentes y no lleven a contradicciones.

# Semana 3 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 6 - El modelo del agente inteligente, Sistemas Multi Agentes, Sistemas Ubicuos.

Notas:



#Importar la librerias

import cv2

import numpy as np

#Activar la WebCam usamos el parametro 0 como defaul si tenemos mas de una camara usamos el

#numero consecutivo ejemplo si tengo 2 camaras el 0 sera la camara 1 y el 1 la camara 2

cam = cv2.VideoCapture(1)

#Validación de Camaaaara

if not cam.isOpened():

    print("Cannot open camera")

    exit()

#Crear un Ciclo para la captura de video

while True:

    #Declaramos 2 variables ret imagen, frame el fotograma

    ret,frame = cam.read()

    #Validamos que ret tenga un valor que mostrar

    if ret==True:

        #Con imshow mostramos una ventana con un titulo y un contenido

        cv2.imshow('frame', frame)

        #Funcion de cierre cuestiono si preciona la tecla q

        if cv2.waitKey(1) & 0xFF == ord('q'):

            break

#Refresco la memoria

cam.release()

#Cierro la Ventana y termino aplicacion

cv2.destroyAllWindows()

#Importar la librerias

import cv2

import numpy as np

#Activar la WebCam usamos el parametro 0 como defaul si tenemos mas de una camara usamos el

#numero consecutivo ejemplo si tengo 2 camaras el 0 sera la camara 1 y el 1 la camara 2

cam = cv2.VideoCapture(1)

#Validación de Camara

if not cam.isOpened():

    print("Cannot open camera")

    exit()

redBajo1 = np.array([0, 100, 20], np.uint8)

redAlto1 = np.array([5, 255, 255], np.uint8)

redBajo2=np.array([175, 100, 20], np.uint8)

redAlto2=np.array([177, 255, 255], np.uint8)

#Crear un Ciclo para la captura de video

while True:

    #Declaramos 2 variables ret imagen, frame el fotograma

    ret,frame = cam.read()

    #Validamos que ret tenga un valor que mostrar

    if ret==True:

        #Convertimos la imagen original de BGR a una imagen HSV

        frameHSV = cv2.cvtColor(frame,cv2.COLOR\_BGR2HSV)

        #Rangos Se mezclan

        maskRed1 = cv2.inRange(frameHSV,redBajo1,redAlto1)

        maskRed2 = cv2.inRange(frameHSV,redBajo2,redAlto2)

        maskRed = cv2.add(maskRed1,maskRed2)

        #Con imshow mostramos una ventana con un titulo y un contenido

        cv2.imshow('CamaraNormal', frame)

        #cv2.imshow('CamaraHSV', frameHSV)

        cv2.imshow('CamaraMaskColor',maskRed)

        maskRedvis = cv2.bitwise\_and(frame, frame, mask= maskRed)

        #cv2.imshow('Video con Redvis', maskRedvis)

        #Funcion de cierre cuestiono si preciona la tecla q

        if cv2.waitKey(1) & 0xFF == ord('q'):

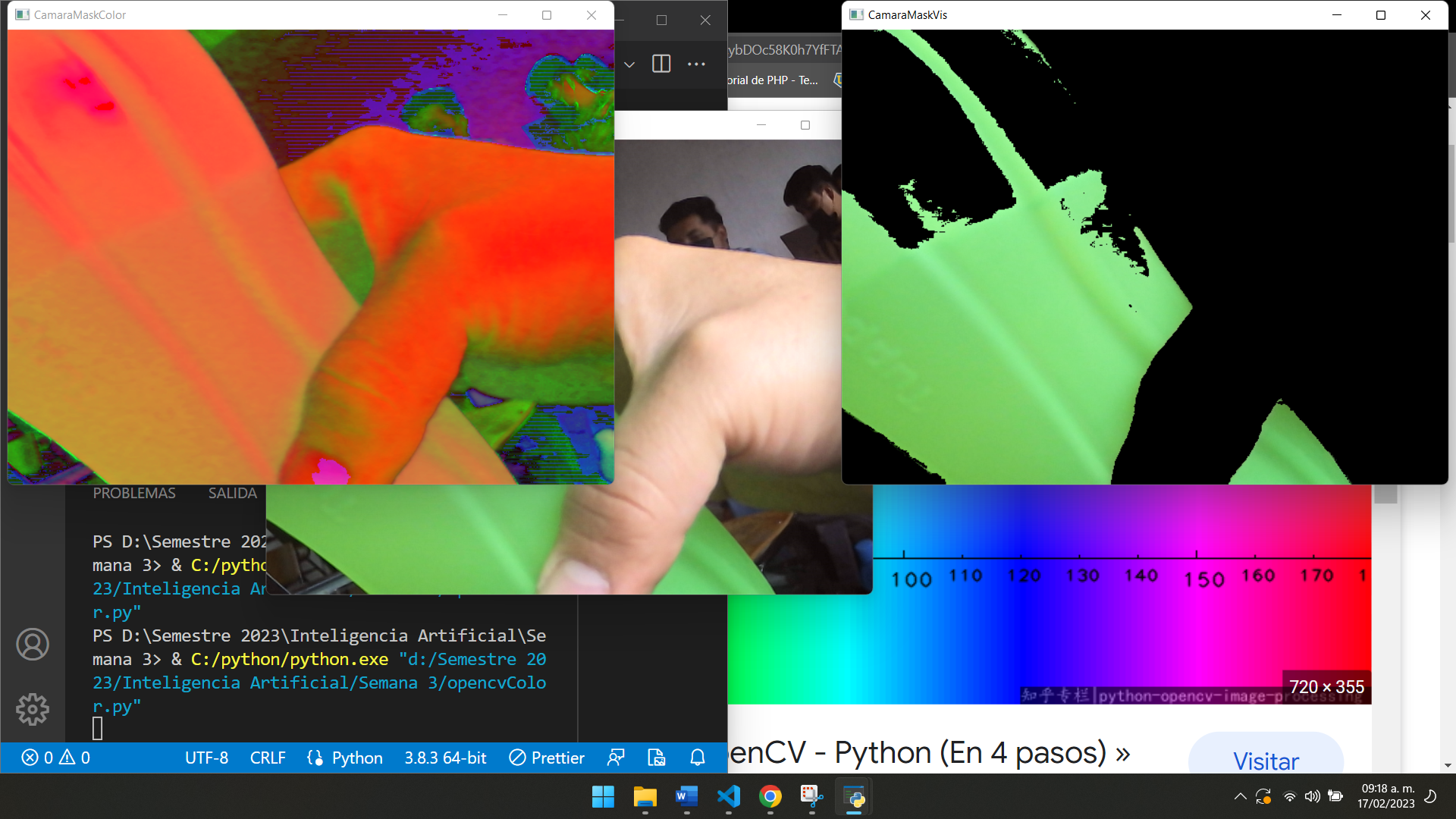
            break

#Refresco la memoria

cam.release()

#Cierro la Ventana y termino aplicacion

cv2.destroyAllWindows()



# Semana 4 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 7 El papel de la heurística.

la heurística es una estrategia mental que utilizamos para tomar decisiones o resolver problemas de manera eficiente, aunque no necesariamente siempre es la forma más precisa o exhaustiva de hacerlo. La heurística se basa en reglas generales y experiencias previas para llegar a una solución.

En muchos casos, la heurística es una herramienta muy útil que nos permite tomar decisiones rápidas y efectivas en situaciones complejas. Por ejemplo, cuando tenemos que elegir entre varias opciones en un corto período de tiempo, nuestra intuición y experiencia previa pueden ser muy valiosas.

Sin embargo, también es importante tener en cuenta que la heurística puede llevarnos a cometer errores o a tomar decisiones equivocadas si no se utiliza de manera adecuada. En algunos casos, la heurística puede hacernos caer en prejuicios o en estereotipos, y puede llevarnos a ignorar información relevante que no se ajusta a nuestras expectativas.

En resumen, el papel de la heurística es ayudarnos a tomar decisiones rápidas y efectivas, pero también es importante tener en cuenta sus limitaciones y estar abiertos a considerar información adicional que pueda ser relevante para la toma de decisiones más precisas y exhaustivas.

import cv2

#Leer imagenes

fotoJoven = cv2.imread("fotojoven.jpg")

fotoMayor = cv2.imread("fotomayor.jpeg")

#Crear variable y mezclar fotos

mixFotos = cv2.addWeighted(fotoJoven, 0.5, fotoMayor, 0.5, 0.0)

#Mostrar el Mix

cv2.imshow("Imagen Mix", mixFotos)

cv2.waitKey(0)

cv2.destroyAllWindows()

# Semana 4 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 8 Aplicación con OPEn CV2.

import cv2

cam = cv2.VideoCapture(1)

print(cam)

#Variable para salidad de video

salida = cv2.VideoWriter('pruebaColor.avi',

                         cv2.VideoWriter\_fourcc(\*'XVID'),

                         24.0,

                         (640,480)

                         )

print(salida)

while(cam.isOpened()):

    #usamos dos parametros ret que es la camara e imagen

    ret,imagen=cam.read()

    #si la camara esta activa mostramos el video en una ventana

    if ret == True:

        cv2.imshow('RED CAM',imagen)

        salida.write(imagen)

        #condición para terminar el proceso al precionar la tecla s

        if cv2.waitKey(1) & 0xFF == ord('s'):

            break

#Limpiar Buffer de Video

cam.release()

salida.release()

cv2.destroyAllWindows()

# Semana 5 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 9 Algoritmos de búsqueda local.

numero = int(input("Ingrese un valor de 4 digitos: "))

print("El valor fue:", numero)

paso = 0

#Algoritmo de busqueda forzada

for x in range(1000, 10000):

    paso = paso + 1

    if x == numero :

        print("Se tardo pasos : ", paso, "Valor de x", x)

import random

numero = int(input("Ingrese un valor de 4 digitos: "))

paso = 0

print("El valor fue:", numero)

#Algoritmo de busqueda aleatoria

for x in range(1000, 10000):

    paso = paso+1

    y = random.randint(1000,9999)

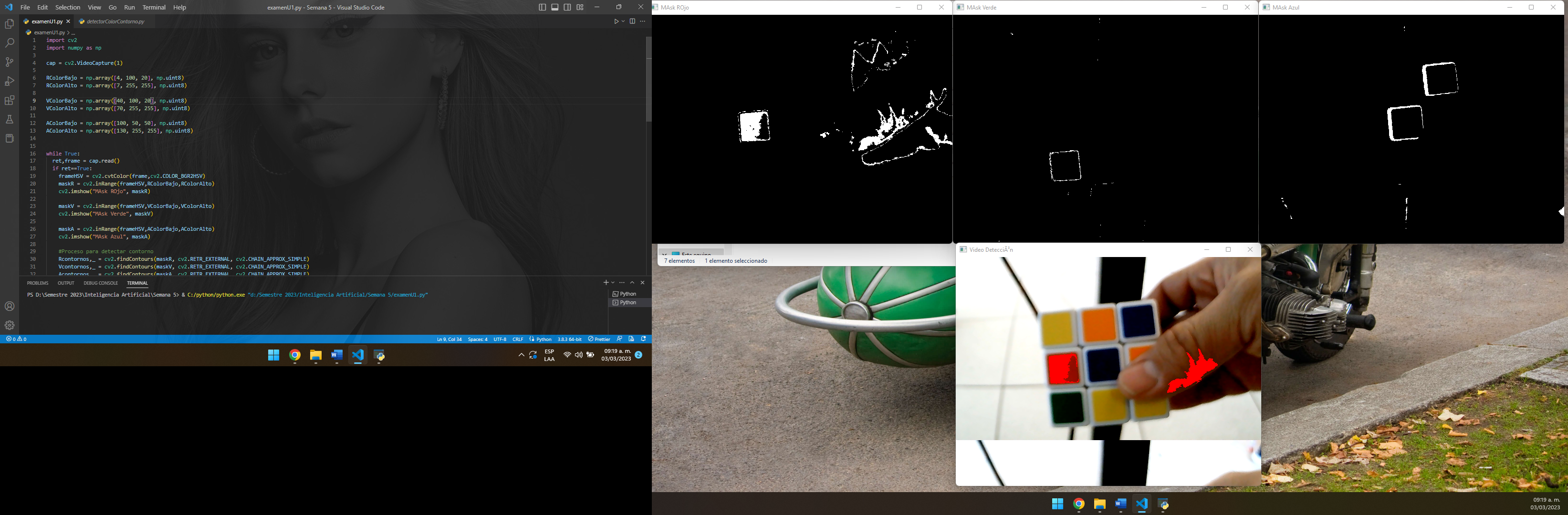
    if y == numero :

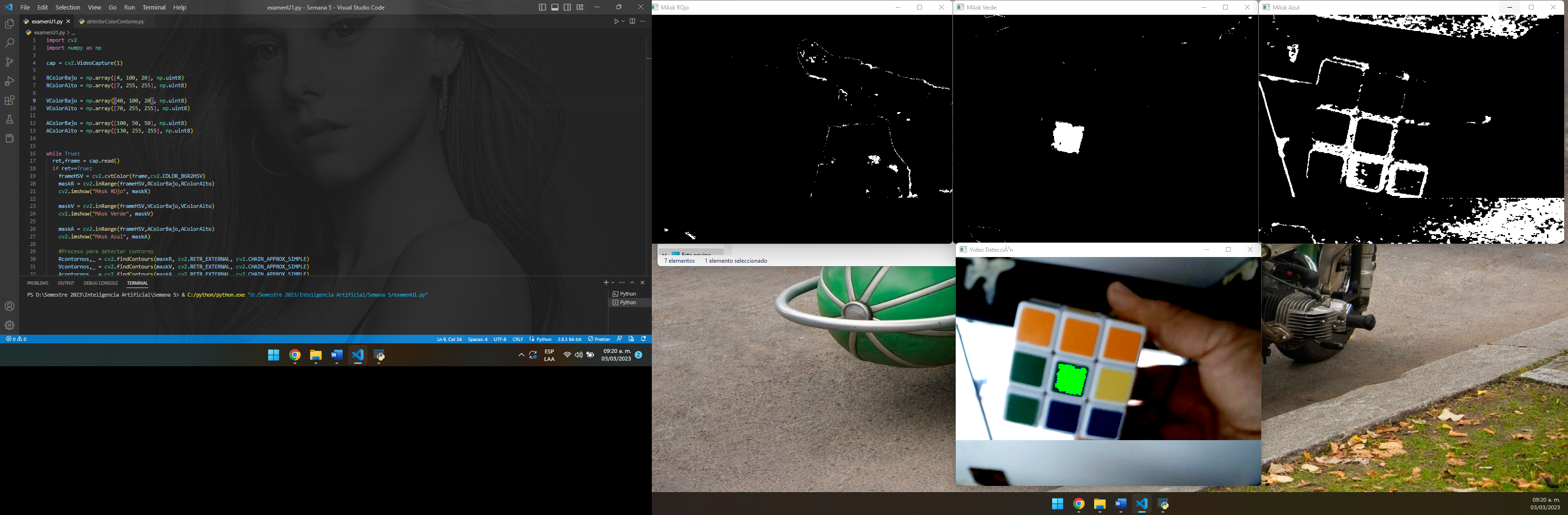
        print("Se tardo total: ", paso, "ciclo x:", x)

# Semana 5 – Unidad 1 - Introducción a la Inteligencia Artificial.

## Contenido de la Sesión 11 Examen.

Desarrollar una aplicación donde mediante los métodos vistos en clase se puedan identificar mediante áreas colores azul, verde y rojo de una escena. Adicionalmente utilizando las librerías de openCV mostrar rellenos con contornos.





Código py

import cv2

import numpy as np

cap = cv2.VideoCapture(1)

RColorBajo = np.array([4, 100, 20], np.uint8)

RColorAlto = np.array([7, 255, 255], np.uint8)

VColorBajo = np.array([40, 100, 20], np.uint8)

VColorAlto = np.array([70, 255, 255], np.uint8)

AColorBajo = np.array([100, 50, 50], np.uint8)

AColorAlto = np.array([130, 255, 255], np.uint8)

while True:

  ret,frame = cap.read()

  if ret==True:

    frameHSV = cv2.cvtColor(frame,cv2.COLOR\_BGR2HSV)

    maskR = cv2.inRange(frameHSV,RColorBajo,RColorAlto)

    cv2.imshow("MAsk ROjo", maskR)

    maskV = cv2.inRange(frameHSV,VColorBajo,VColorAlto)

    cv2.imshow("MAsk Verde", maskV)

    maskA = cv2.inRange(frameHSV,AColorBajo,AColorAlto)

    cv2.imshow("MAsk Azul", maskA)

    #Proceso para detectar contorno

    Rcontornos,\_ = cv2.findContours(maskR, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

    Vcontornos,\_ = cv2.findContours(maskV, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

    Acontornos,\_ = cv2.findContours(maskA, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

    #Proceso para dibujar contorno detectado con un ciclo

    for c in Rcontornos:

      area = cv2.contourArea(c)

      if area > 1000:

        nuevoContorno = cv2.convexHull(c) #esta funcion elimina picos

        cv2.drawContours(frame, [c], -1, (0,0,255), -1)

    for cv in Vcontornos:

      areav = cv2.contourArea(cv)

      if areav > 3000:

        nuevoContornoV = cv2.convexHull(cv) #esta funcion elimina picos

        cv2.drawContours(frame, [cv], -1, (0,255,0), -1)

    for ca in Vcontornos:

      areaa = cv2.contourArea(ca)

      if areaa > 3000:

        nuevoContornoA = cv2.convexHull(ca) #esta funcion elimina picos

        cv2.drawContours(frame, ca, -1, (255,0,0), -1)

    cv2.imshow('Video Detección',frame)

    if cv2.waitKey(1) & 0xFF == ord('s'):

      break

cap.release()

cv2.destroyAllWindows()