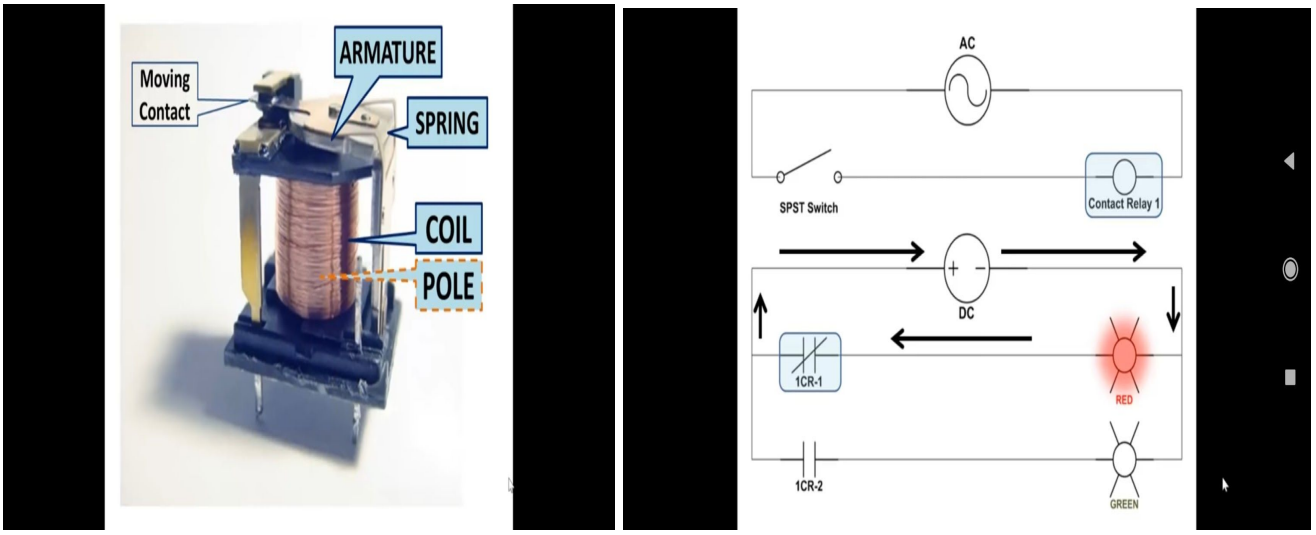
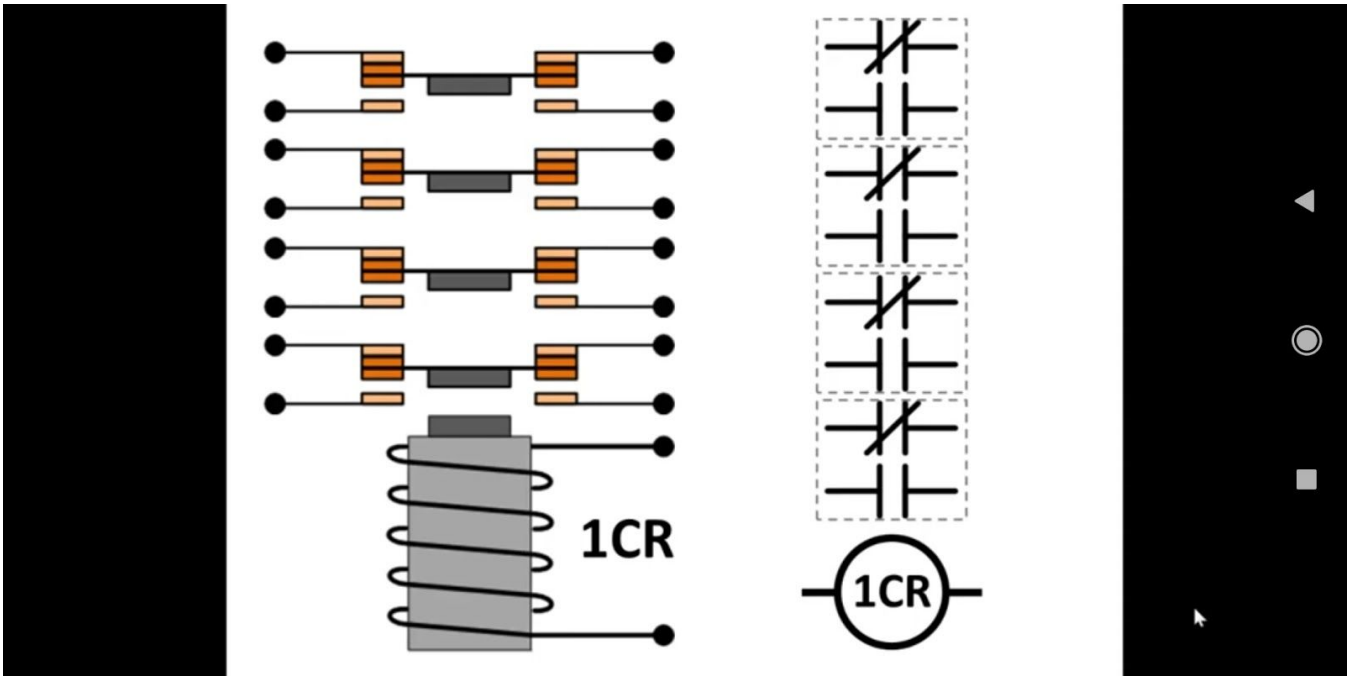
DAILY ASSESSMENT FORMAT

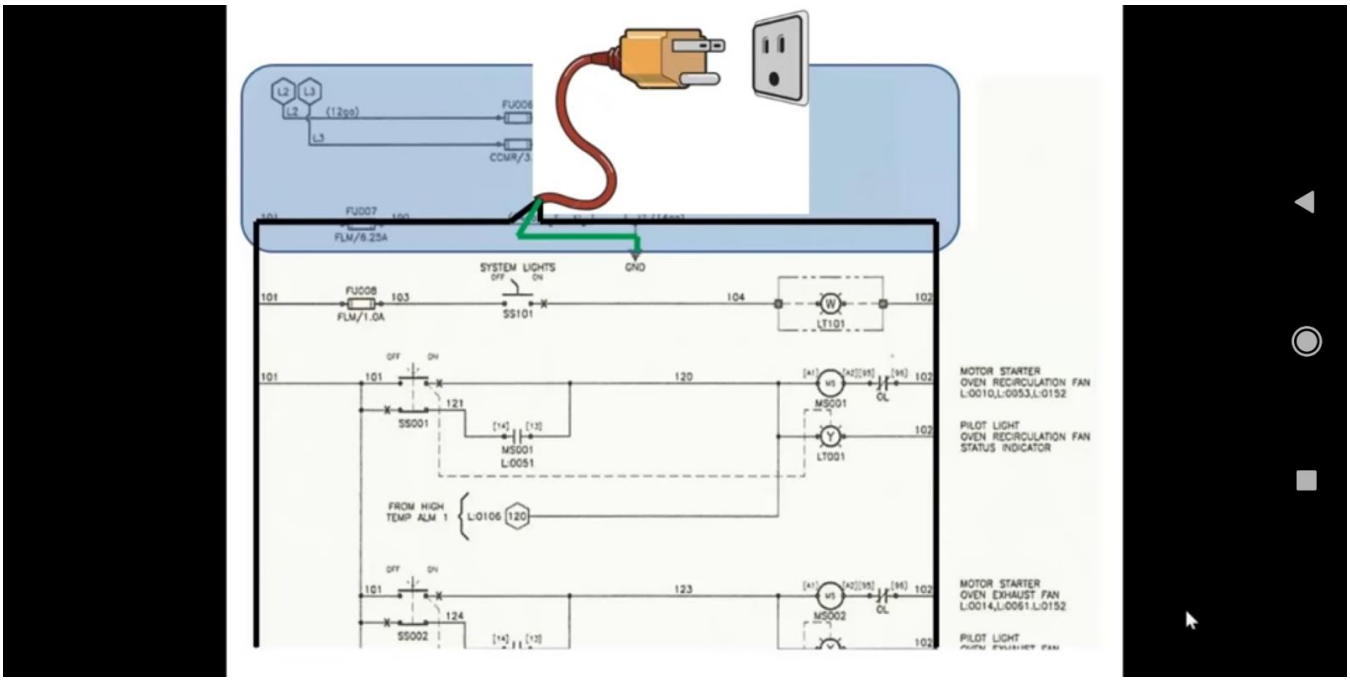
Date:	**30-05-2020**	**Name:**	**MOUNITHA D M**
Course:	**LOGIC DESIGN**	**USN:**	**4AL17EC055**
Topic:	**Application of programmable logic Controllers**	**Semester & Section:**	**6TH SEM "A" SEC**
Github Repository:	**Mounitha_-ec055**		

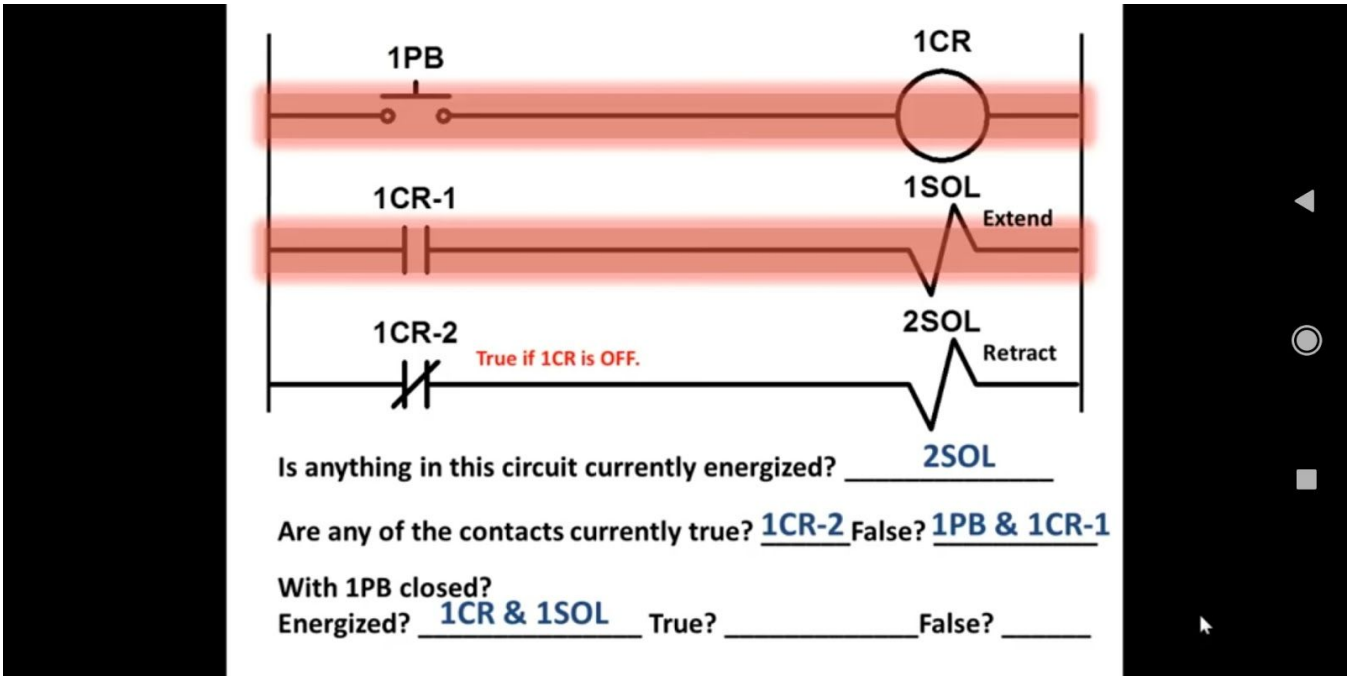
FORENOON SESSION DETAILS

Image of session









Is anything in this circuit currently energized? 2SOL

Are any of the contacts currently true? 1CR-2 False? 1PB & 1CR-1

With 1PB closed?
Energized? 1CR & 1SOL True? _____ False? _____

Report – Report can be typed or hand written for up to two pages.

Logic Design 30/05/2020

Day 3 :

Application of programmable logic Controllers

Relays to Bits

→ A programmable logic controller or programmable controller is an industrial digital computer which has been ruggedized and adapted for the control of manufacturing processes.

→ It is a simply a special computer device for industrial control system

→ PLC some of the process we can do is listed

1 Continuous mixing System
2 Batch mixing System
3 Stages air conditioning System
4 Traffic control.

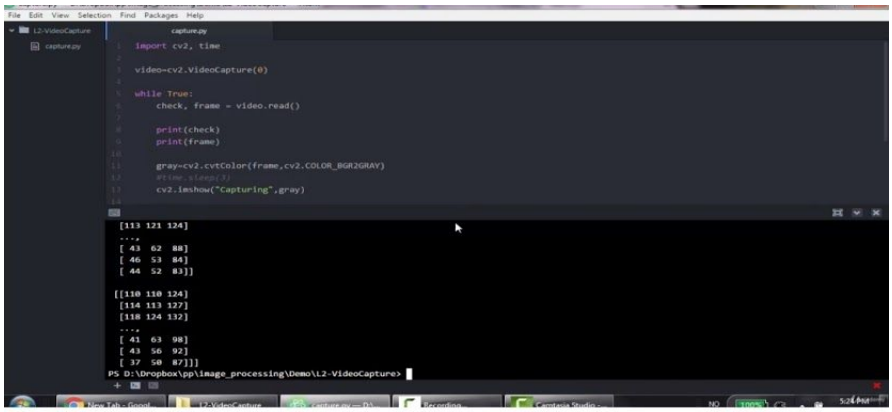
Advantages of PLC

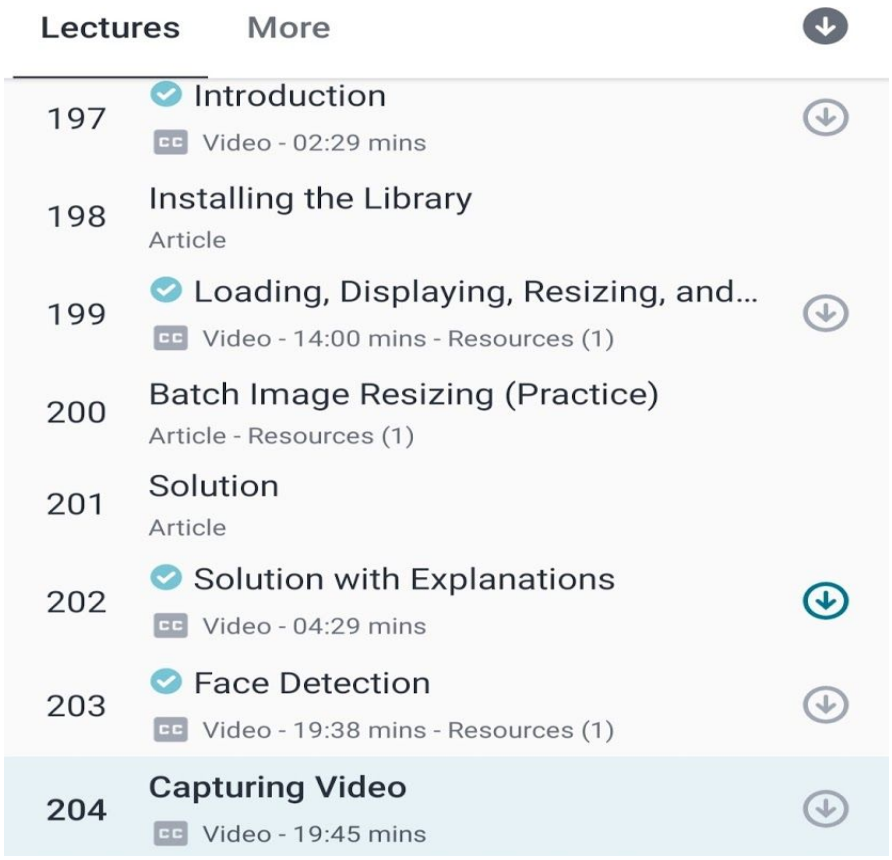
→ cost effective to control complex System
→ Flexible and can be reapplied to control other systems quickly and easily
→ Computer skills allow more sophisticated control
→ Less manpower for design
→ Downsizing and standardization
→ Improved maintainability.

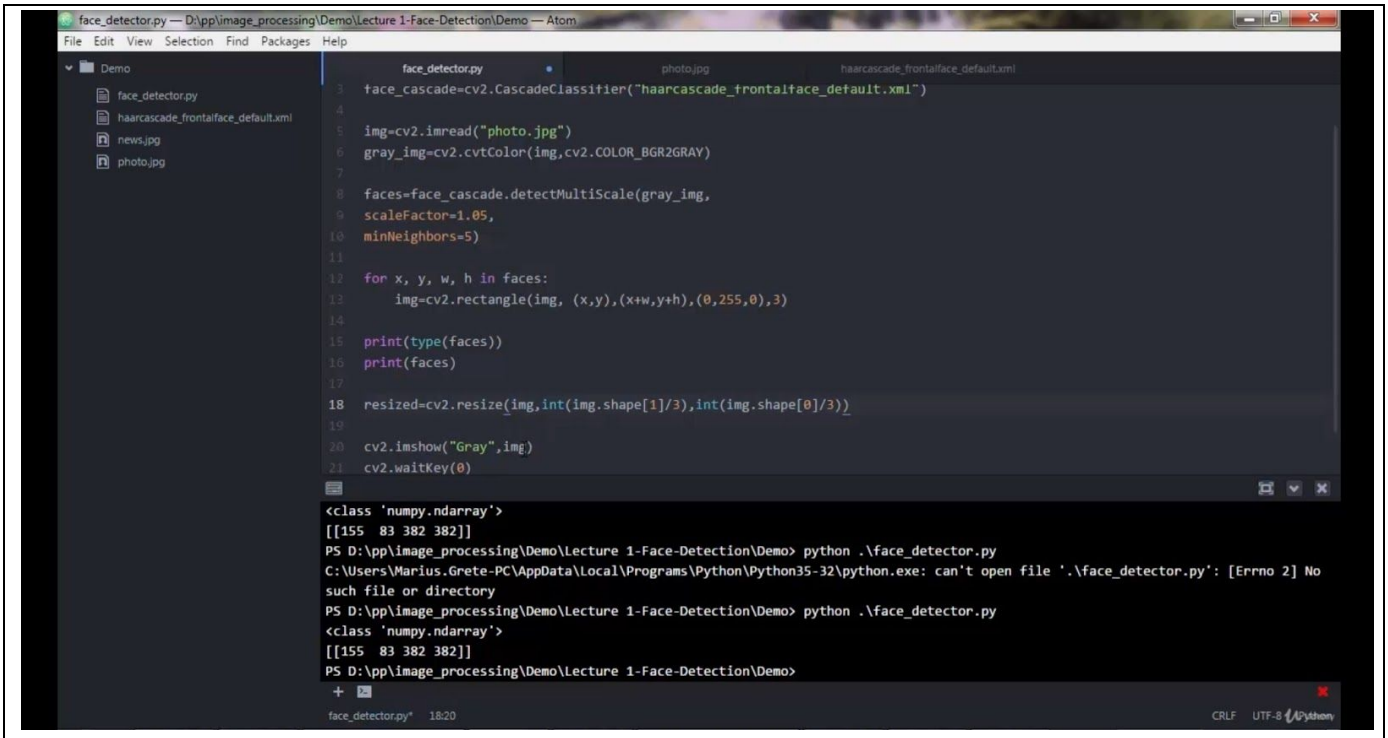
DATE	**30-05-2020**	**Name:**	**MOUNITHA DM**
Course:	**PYTHON**	**USN:**	**4AL17EC055**
Topic:	**Python for image and video processing with opencv**	**Semester & Section:**	**6TH SEM "A" SEC**

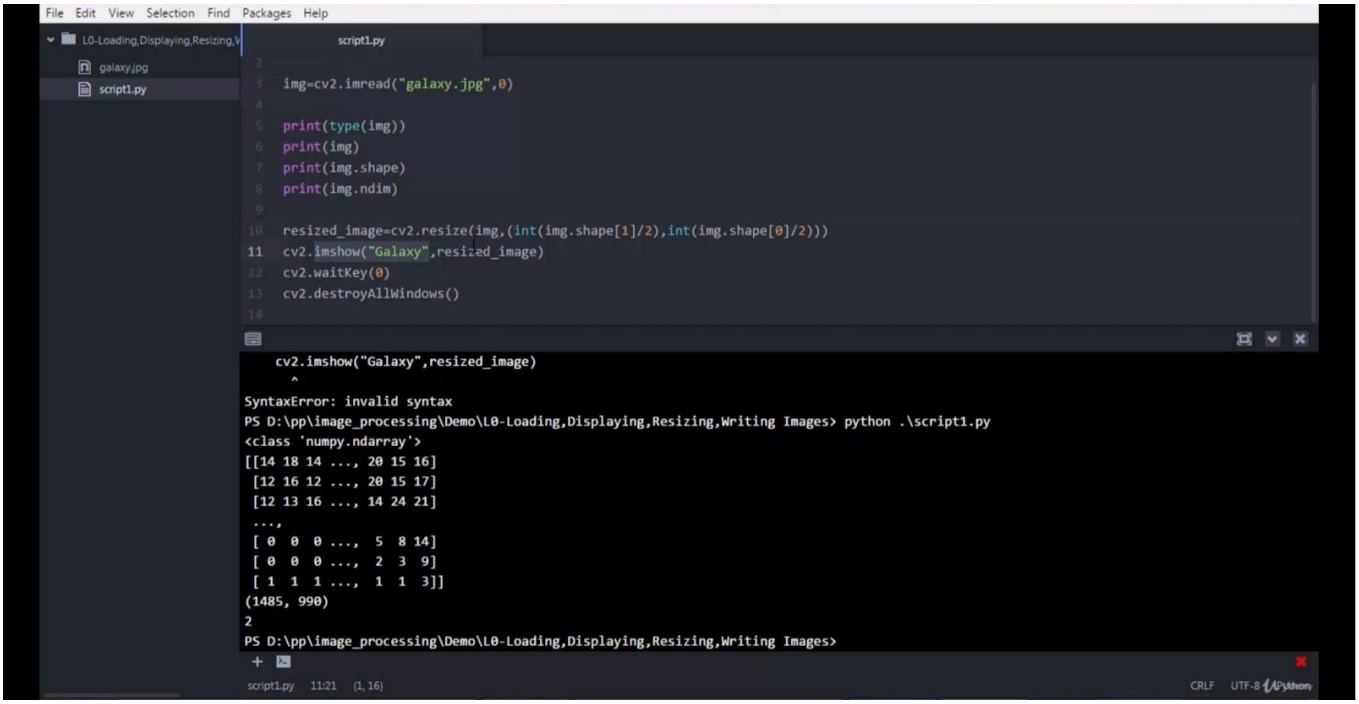
AFTERNOON SESSION DETAILS

Image of session

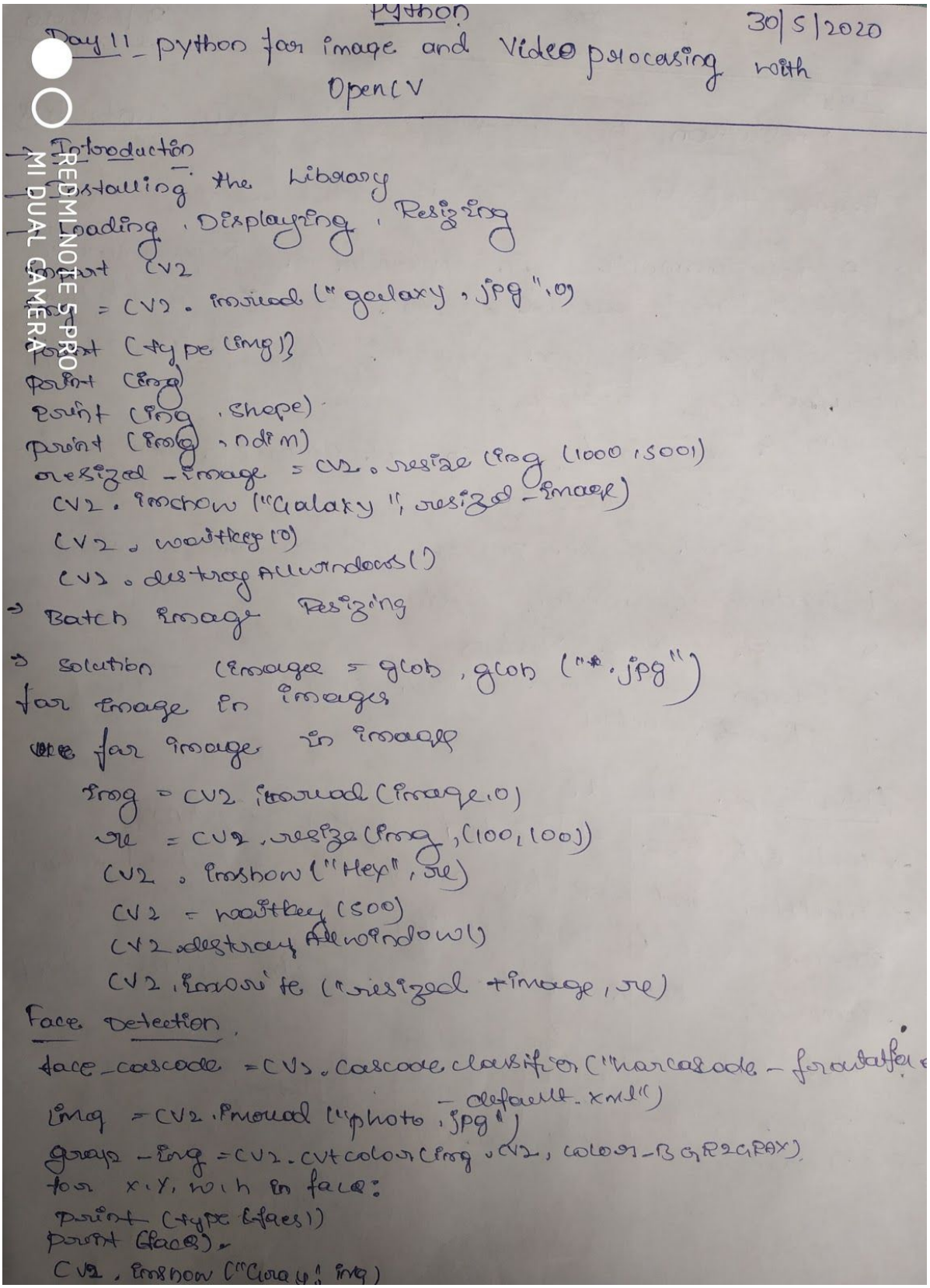





face_detector.py — D:\pp\image_processing\Demo\Lecture 1-Face-Detection\Demo — Atom
File Edit View Selection Find Packages Help
Demo
face_detector.py
haarcascade_frontalface_default.xml
news.jpg
photo.jpg
face_detector.py
photo.jpg
haarcascade_frontalface_default.xml
face_cascade=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
img=cv2.imread("photo.jpg")
gray_img=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
faces=face_cascade.detectMultiScale(gray_img,
scaleFactor=1.05,
minNeighbors=5)
for x, y, w, h in faces:
img=cv2.rectangle(img, (x,y),(x+w,y+h),(0,255,0),3)
print(type(faces))
print(faces)
resized=cv2.resize(img,int(img.shape[1]/3),int(img.shape[0]/3))
cv2.imshow("Gray",img)
cv2.waitKey(0)
<class 'numpy.ndarray'>
[[155 83 382 382]]
PS D:\pp\image_processing\Demo\Lecture 1-Face-Detection\Demo> python .\face_detector.py
C:\Users\Marius.Grete-PC\AppData\Local\Programs\Python\Python35-32\python.exe: can't open file '.\face_detector.py': [Errno 2] No such file or directory
PS D:\pp\image_processing\Demo\Lecture 1-Face-Detection\Demo> python .\face_detector.py
<class 'numpy.ndarray'>
[[155 83 382 382]]
PS D:\pp\image_processing\Demo\Lecture 1-Face-Detection\Demo>
face_detector.py* 18:20
CRLF
UTF-8


File Edit View Selection Find Packages Help
L0-Loading,Displaying,Resizing,W
galaxy.jpg
script1.py
script1.py
img=cv2.imread("galaxy.jpg",0)
print(type(img))
print(img)
print(img.shape)
print(img.ndim)
resized_image=cv2.resize(img,(int(img.shape[1]/2),int(img.shape[0]/2)))
cv2.imshow("Galaxy",resized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
cv2.imshow("Galaxy",resized_image)
^
SyntaxError: invalid syntax
PS D:\pp\image_processing\Demo\L0-Loading,Displaying,Resizing,Writing Images> python .\script1.py
<class 'numpy.ndarray'>
[[14 18 14 ..., 20 15 16]
[12 16 12 ..., 20 15 17]
[12 13 16 ..., 14 24 21]
...,
[0 0 0 ..., 5 8 14]
[0 0 0 ..., 2 3 9]
[1 1 1 ..., 1 1 3]]
(1485, 990)
2
PS D:\pp\image_processing\Demo\L0-Loading,Displaying,Resizing,Writing Images>
script1.py 11:21 (1, 16)
CRLF
UTF-8

Report – Report can be typed or hand written for up to two pages.



Python

30/5/2020

Day 11 - python for image and Video processing with OpenCV

→ Introduction
→ Installing the Library
→ Loading, Displaying, Resizing

```
import cv2
img = cv2.imread("galaxy.jpg",0)
print(type(img))
print(img)
print(img.shape)
print(img.ndim)
resized_image = cv2.resize(img,(1000,500))
cv2.imshow("Galaxy", resized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

→ Batch Image Resizing

→ Solution

```
images = glob.glob("*.jpg")
for image in images
for image in images
    img = cv2.imread(image,0)
    re = cv2.resize(img,(100,100))
    cv2.imshow("Hey", re)
    cv2.waitKey(500)
    cv2.destroyAllWindows()
    cv2.imwrite("resized"+image, re)
```

Face detection

```
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
img = cv2.imread('photo.jpg')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
for x,y,w,h in face:
print(type(faces))
print(faces)
cv2.imshow('Gray', img)
```