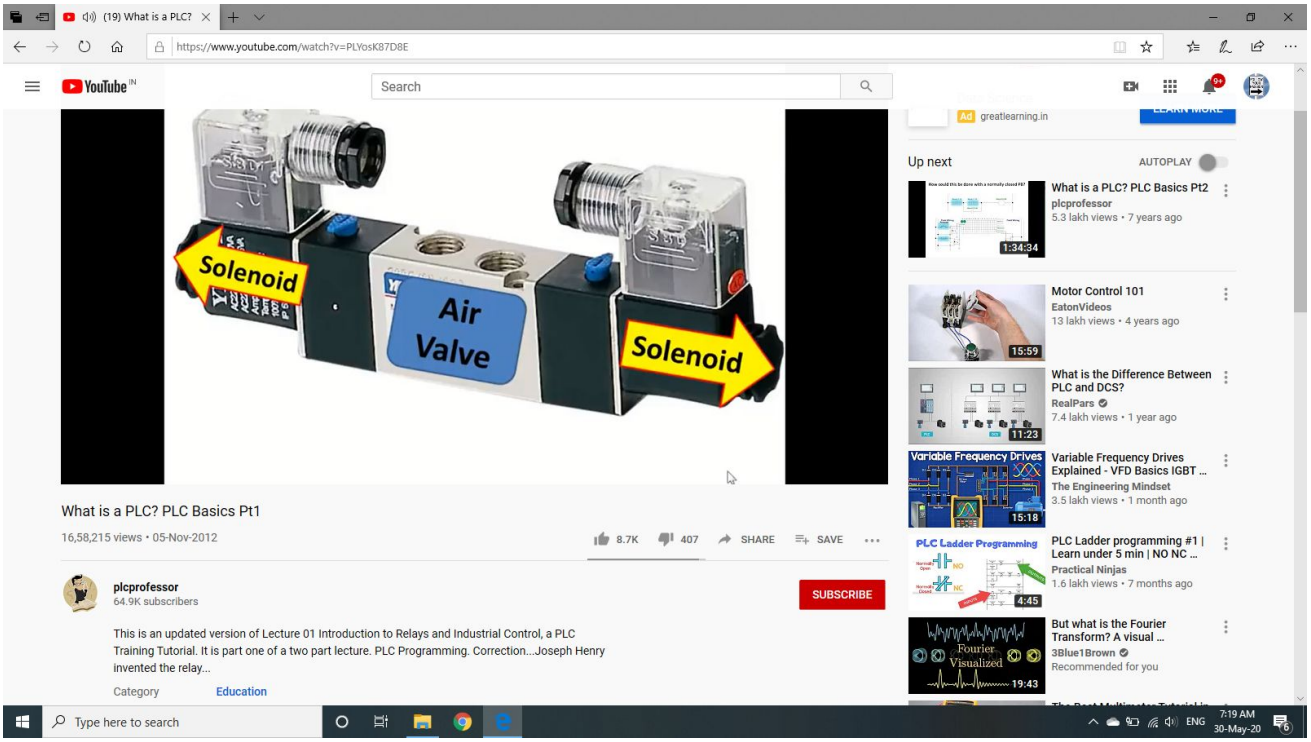
REPORT MAY 30

Date:	**30 MAY 2020**	**Name:**	**Rakshith B**
Course:	**Logic Design**	**USN:**	**4AL16EC409**
Topic:	**Applications of Programmable logic controllers.**	**Semester & Section:**	**6th SEM B**
Github Repository:	**Rakshith-B**		

FORENOON SESSION DETAILS

Image of session



Report –

Applications of Programmable logic controllers

What is PLC ?

A Programmable Logic Controller, or PLC, is a ruggedized computer used for industrial automation. These controllers can automate a specific process, machine function, or even an entire production line.

Working:

- **I/O – The PLC's CPU stores and processes program data, but input and output modules connect the PLC to the rest of the machine; these I/O modules are what provide information to the CPU and trigger specific results**
- **Communications – In addition to input and output devices, a PLC might also need to connect with other kinds of systems; for example, users might want to export application**

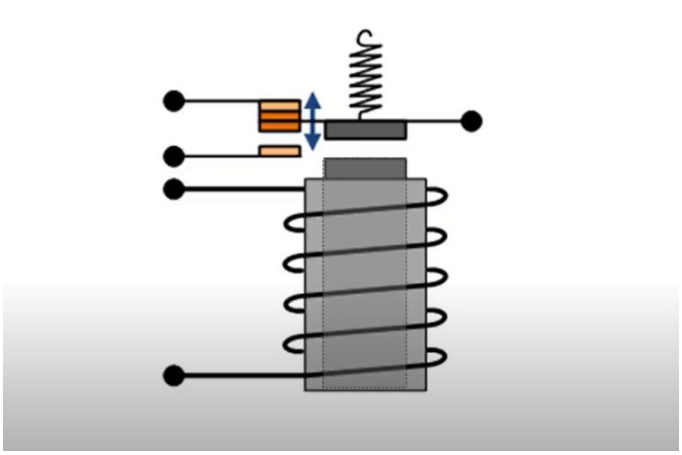
data recorded by the PLC to a supervisory control and data acquisition (SCADA) system, which monitors multiple connected devices.

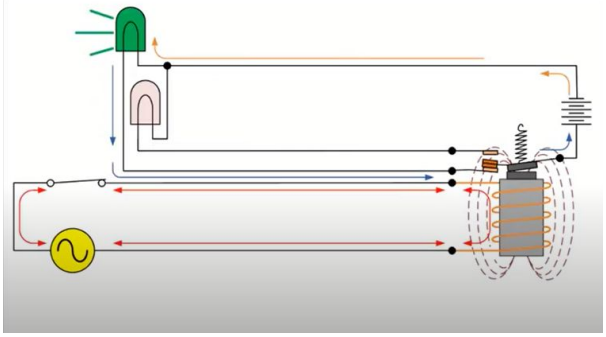
- **HMI – In order to interact with the PLC in real time, users need an HMI, or Human Machine Interface.**

What are the different types of PLC?

In addition to the traditional PLC described above, there are variations, including PLC + HMI controllers.

Different Applications Of Relays:
Construction of Relay:





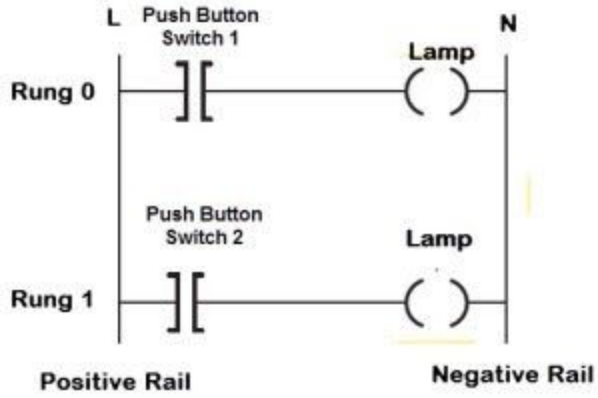
Solenoid:

PLC Programming Basics:

A CPU of the PLC executes two different programs:

- **The Operating System**
- **The User Program**

Ladder Logic PLC Programming



Among several programming languages, ladder logic diagram is the most basic and simplest form of programming the PLC. Before going to program the PLC with this language, one should know some basic information about it. The below figure shows the hardwired-ladder diagram wherein the same lamp load is controlled by two push button switches, In case if any one of the switches gets closed, the lamp glows. Here two horizontal lines are called rungs which are connected between two vertical lines called rails. Each rung establishes the electrical continuity between positive (L) and negative rails (N) so that the current flows from the input to output devices. Some of the symbols used in ladder logic programming are shown in the figure.

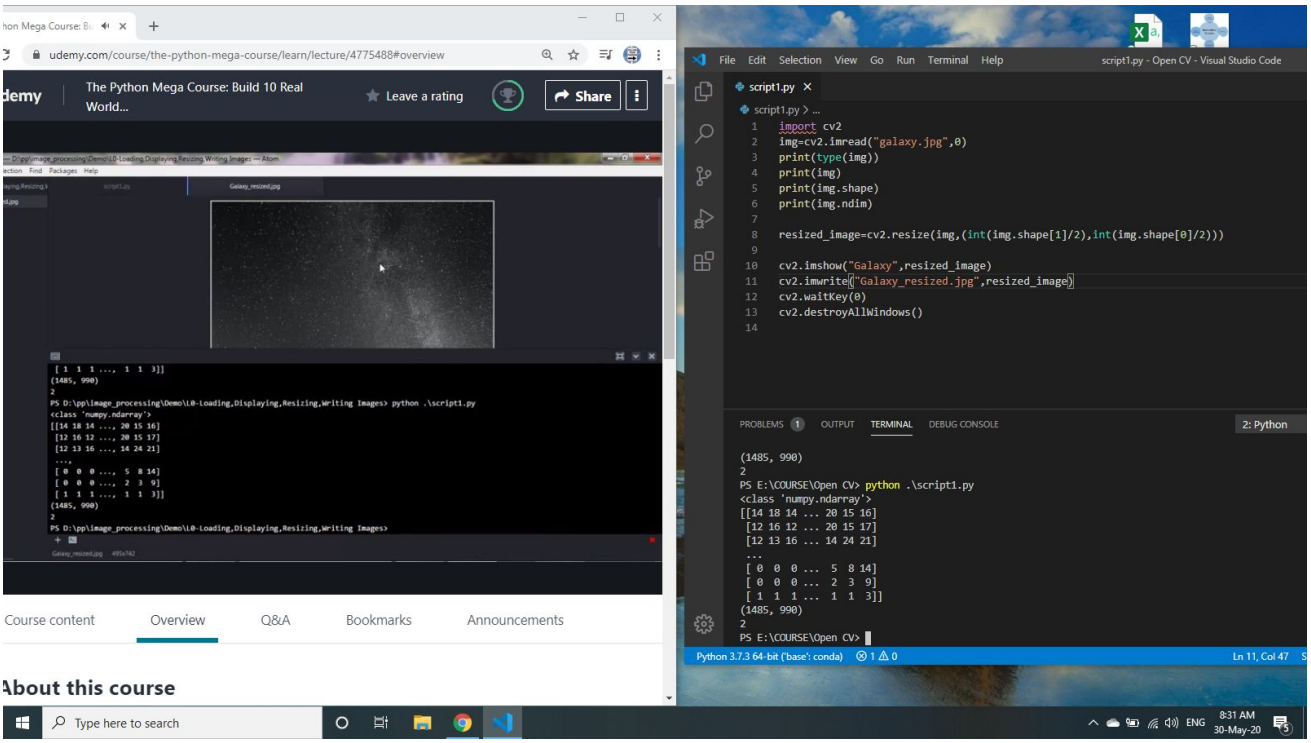
Input switches are types included normally closed and normally opened as shown above. In addition to above given functional symbols, there are several functions like timer, counter, PID, etc., which are stored in the standard library to program complex tasks.

Date: 30 MAY 2020
Course: Python On Udemy
Topic: Python for Video and Image Processing Using Open CV

Name:RAKSHITH B
USN:4AL16EC409
Semester & Section:6 B

AFTERNOON SESSION DETAILS

Image of session:Output



Report –

Loading,Displaying,Resizing and Writing Images:

```
import cv2
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.imwrite("Galaxy_resized.jpg",resized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Face Detection:

```
import cv2

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",resized)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Video Capturing:

```
import cv2,time
video=cv2.VideoCapture(0)
a=0
while True:
    a=a+1
    check,frame=video.read()
    print(check)
    print(frame)

    gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
   # time.sleep(3)
    cv2.imshow("Capturing",gray)

    key=cv2.waitKey(1)
    if key==ord('q'):
        break
print(a)
video.release()
cv2.destroyAllWindows
```