# Pre-requisites Done:

* Read and understood basics of Python
* Understood about installation and using python
* Learnt what is module, how it is imported and can be used in the code
* Drawing line, opening a jpeg file

## Draw a rectangle and write text inside it (using Turtle & Image Draw modules)

Analyzed the module *ImageDraw module* : The [ImageDraw](https://pillow.readthedocs.io/en/stable/reference/ImageDraw.html" \l "module-PIL.ImageDraw" \o "PIL.ImageDraw) module provides simple 2D graphics for [Image](https://pillow.readthedocs.io/en/stable/reference/Image.html#PIL.Image.Image) objects. Here, the TB diagram is aimed to construct on a *.jpg file*. Since it is easy to draw shapes/arrows in image file, “Image draw module” has been chosen. In the below script,

* Created an empty image \*.jpg file
  + *img = Image.new("RGB", (500, 500),"white")*
* Drew a rectangle in that image using “ImageDraw” module
* Saved the image & using the Image Draw module, inserted the text inside the rectangle (by changing the required dimensions in trial and error manner)
  + *Img1.rectangle(), img1.text() methods*
* Added that .jpg file into a .docx document & saved that. (using docx module)

Analyzed & tried with turtle

* *doc.add\_picture()*

*Using ImageDraw Module*

import docx

from PIL import Image,ImageDraw

doc = docx.Document()

img = Image.new("RGB", (500, 500),"white")

# create a image draw handle

img1 = ImageDraw.Draw(img)

img1.rectangle((200,125,300,200),fill ="orange", outline = "black",width = 1)

img1.text((210, 150), "CHECK TEXT", fill = "black",align = "center")

img.show()

#Choose the file path accordingly

img.save('C:\\Users\\path \\line.jpg');

doc.add\_picture('C:\\path \\line.jpg');

doc.save('C:\\path\\filename.jpg);

*Using Turtle :*

* *“Turtle”* is a pre-installed Python library that enables users to create pictures and shapes
* Created rectangle by traversing via directions and inserted text inside the rectangle

from turtle import \*

# Choose Color for rectangle

color("orange")

begin\_fill() # Enabling fill to color the shape

# Traverse in directions, to draw rectangle

forward(300);

right(90)

forward(150)

right(90)

forward(300);

right(90)

forward(150)

right(90)

# End the coloring

end\_fill()

color("BLACK") # Choose Black color to write

begin\_fill()

penup()

forward (150)

#right (45)

left(65)

backward (20)

write("TEXT INSIDE RECTANGLE USING TURTLE", True, align="center")

#write("TEXT", True, align="center")

## Draw a rectangle and inside that, two inner rectangles to denote AGENT, SEQUENCER, DRIVER (Also drawing arrows using opencv module)

* Analyzed the image draw diagram and to draw the AGENT – SEQR-DRV
* Using *Img1.rectangle(), img1.text() methods, draw rectangle with desired dimensions*
* Analyzed the opencv module to define arrows between two different blocks
  + *cv2.arrowedLine()*
* Completed the trial blocks for top, test, agent blocks with arrow and made it to open in a word document
* Analyzed the basics of regular expression to search for a particular pattern

import docx

import cv2

from PIL import Image,ImageDraw

doc = docx.Document()

img = Image.new("RGB", (500, 500),"white")

w = 500;

h = 500;

# create line image

img1 = ImageDraw.Draw(img)

#Outer most rectangle

w1 = 10

h1 = 5

img1.rectangle((w1,h1,w-10,h-10),fill ="orange", outline = "black",width = 1)

img1.text((w-40, h1+5), "TOP", fill = "black")

#Inner rectangle

w2 = w1 + 20

h2 = h1 + 15

img1.rectangle((w2,h2,w-w2,h-h2),fill ="VIOLET", outline = "black",width = 1)

img1.text((w-30-40, 26), "TEST", fill = "black")

#Inner rectangle ENV

img1.rectangle((38,40,460,472),fill ="YELLOW", outline = "black",width = 1)

img1.text((430, 50), "ENV", fill = "black")

#Inner rectangle AGENT

w3 = w2 + 20

h3 = h2 + 60

img1.rectangle((w3,h3,(w - w1\*20),(w - w1\*20)),fill ="pink", outline = "black",width = 1)

img1.text((250, h3), "AGENT", fill = "black")

'''img1.rectangle((50,10,300,300),fill ="pink", outline = "black",width = 1)

img1.text((250, 25), "AGENT", fill = "black")'''

#Another rectangle DRIVER

w4 = w3 + 20

h4 = h3 + 15

'''img1.rectangle((100,50,180,90),fill ="orange", outline = "black",width = 1)

#img1.text((105, 65), "SEQUENCER", fill = "black")'''

img1.rectangle((w4,h4,w4 + 100,h4 + 50),fill ="orange", outline = "black",width = 1)

img1.text((w4 + 10, h4 + 20), "SEQUENCER", fill = "black")

# Another rectangle sequencer

w5 = w4

h5 = h4 + h4 +15 #h3 + 200

'''img1.rectangle((100,140,190,170),fill ="orange", outline = "black",width = 1)

img1.text((125, 150), "DRIVER", fill = "black")'''

img1.rectangle((w5,h5,w5 + 100,h5 + 50),fill ="orange", outline = "black",width = 1)

img1.text((w5 + 20, h5 + 20), "DRIVER", fill = "black")

# Another rectangle for DUT

img1.rectangle((100,390,450,420),fill ="gray", outline = "black",width = 1)

img1.text((280,400), "DUT", fill = "black")

img.show()

img.save('C:\\Users\\goushik\\Desktop\\New folder\\line.jpg');

# Arrow Drawing

path = 'C:\\Users\\goushik\\Desktop\\New folder\\line.jpg'

# Reading an image in default mode

image = cv2.imread(path)

# Window name in which image is displayed

window\_name = 'Image'

# Start coordinate

start\_point = (w4 + 40, h4+50)

# End coordinate

end\_point = (w5 + 40, h5)

color = (0, 0, 0)

thickness = 3

# Using cv2.arrowedLine() method

image = cv2.arrowedLine(image, start\_point, end\_point,color, thickness)

cv2.imshow(window\_name, image)

start\_point = (200, 300)

# End coordinate

end\_point = (200,390)

color = (0, 0, 0)

thickness = 3

# Using cv2.arrowedLine() method

image = cv2.arrowedLine(image, start\_point, end\_point,color, thickness)

cv2.imshow(window\_name, image)

cv2.imwrite("C:\\Users\\goushik\\Desktop\\New folder\\line.jpg",image)

doc.add\_picture('C:\\Users\\goushik\\Desktop\\New folder\\line.jpg')

doc.save('C:\\Users\\goushik\\Desktop\\New folder\\pattern\_printing\_ex.docx')

## Script to generalize the dimensions of the diagram to take care the multiple agent blocks in the TB

Analyzed and updated a common method to draw rectangle inside a rectangle for top,test,env w.r.t. change in jpeg file aspect ratio

def draw\_rect(image,coordinates,fill,color,width=1):

rect\_start = (coordinates[0][0],coordinates[0][1]);

rect\_end = (coordinates[1][0], coordinates [1][1])

image.rectangle((rect\_start,rect\_end),fill=fill,outline = color)

def wr\_text\_in\_rect(image,start\_wr\_w,start\_wr\_h,str,tfill):

image.text((start\_wr\_w,start\_wr\_h),str, fill = tfill)

def call\_simple\_rect(w,h,n,text,bfill,tfill,img1):

w1 = w - (n\*10); #end of x should be max

h1 = h - (n\*10); #end of 'y' should be max

len\_init = n\*15 + 10;

top\_left = (w1,len\_init)

bottom\_right = (len\_init,h1)

start\_x = w1 - (25);

start\_y = len\_init + (n\*2);

outline\_width = 10

outline\_color = "black"

draw\_rect(img1,(top\_left, bottom\_right), fill=bfill ,color=outline\_color, width=outline\_width)

wr\_text\_in\_rect(img1,start\_x,start\_y,text,tfill)

print ("Dimensions are %0d %0d %0d %0d",top\_left, bottom\_right)

return w1;

import docx

import cv2

from PIL import Image,ImageDraw

doc = docx.Document()

# create line image of width and height

w = 500

h = 500

img = Image.new("RGB", (w, h),"white")

img1 = ImageDraw.Draw(img)

#Create first outer rectangle top n=1

n = 1;

top\_dim = call\_simple\_rect(w,h,n,"TOP","orange","black",img1);

#Create second inner rectangle test n=2

n = 2;

test\_dim = call\_simple\_rect(w,h,n,"TEST","green","black",img1);

#Create third inner rectangle env n=3

n = 3;

env\_dim = call\_simple\_rect(w,h,n,"ENV","yellow","black",img1);

#Create fourth inner rectangle sequences DUT

top\_left = (100,h-80)

bottom\_right = (h-100,h-50)

draw\_rect(img1,(top\_left, bottom\_right), fill="grey" ,color="black", width=10)

wr\_text\_in\_rect(img1,h-250,h-80,"DUT","BLACK")

#Create fifth inner rectangle VIF

#draw\_rect(img1,(top\_left, bottom\_right), fill=bfill ,color=outline\_color, width=outline\_width)

#wr\_text\_in\_rect(img1,start\_x,start\_y,text,tfill)

#Check for number of agents

n = 4;

agnt\_cnt = int(input("Enter no. of agents"))

#agt\_dim = call\_simple\_rect(w,h,n,"AGENT","pink","black",img1);

w1 = (env\_dim/2)

h1 = (env\_dim/2)

if agnt\_cnt == 1:

len\_init = (env\_dim/2) + 10;

tl = (w - (n\*10),n\*15 + 100)

br = (n\*15 + 10,h - (4\*n\*10))

top\_left = tl;

bottom\_right = br;

start\_x = w - (n\*10)- 35;

start\_y = len\_init + (n\*2) - 80;

outline\_width = 10

outline\_color = "black"

draw\_rect(img1,(top\_left, bottom\_right), fill="pink" ,color="black", width=outline\_width)

wr\_text\_in\_rect(img1,start\_x,start\_y,"AGENT","BLACK")

elif agnt\_cnt > 1:

len\_init = (env\_dim/2) + 10;

for val in range(agnt\_cnt):

w1 = (env\_dim/agnt\_cnt)

h1 = (env\_dim/agnt\_cnt)

if val== 0:

tl = ((env\_dim - (n\*100)),n\*15 + 100);#(w - (n\*10),w1 - (n\*10))

br = ((w1 + 10),(w1+10+100));#(w1 - (n\*10),h - (4\*n\*10))

top\_left = tl;

bottom\_right = br;

start\_x = w1 + 10 - 35;

start\_y = w1 - 75;

outline\_width = 10

outline\_color = "black"

draw\_rect(img1,(top\_left, bottom\_right), fill="pink" ,color="black", width=outline\_width)

wr\_text\_in\_rect(img1,start\_x,start\_y,"AGENT","BLACK")

elif val == 1:

print("CHECK VALUE %0d",val);

tl1 = (280,160);#((w1 + 25),(w1+25+100));

br1 = (460,342);#(n\*15 + 10,h - (4\*n\*10));

top\_left = tl1;

bottom\_right = br1;

start\_x = 460 - 30;

start\_y = 160;

outline\_width = 10

outline\_color = "black"

draw\_rect(img1,(top\_left, bottom\_right), fill="pink" ,color="black", width=outline\_width)

wr\_text\_in\_rect(img1,start\_x,start\_y,"AGENT","BLACK")