# if you are on 32 bit OS :

# import Image

from PIL import Image

import numpy as np

import matplotlib.pyplot as plot

# exemple as how to open an image (store it sort of)

exemple = Image.open(‘images/dotndot.png’)

# convert an image to array

exemple\_array = np.asarray(exemple)

# see values or R, G and B

print iar

# visualise an image using matplotlib

plot.imshow(exemple\_array)

plot.show()

# function definition, let’s start with Green detection

# Red and Blue will be copy pastes without commentary

def green\_detect(image\_array):

# note that ‘’tabs’’ here are important

# detection\_values will allow us to determine if the color is mostly green

# exemple : [0 255 0 0] is green [R G B ?]

detection\_values = []

# new\_image\_array will allow us to create a new image, black or green

new\_image\_array = image\_array

# if G value is above 100 from any other value, then it is green

for each\_row in image\_array

for each\_pixel in each\_row

list\_pixel\_0\_2 = [each\_pixel[0], each\_pixel[2]]

detect\_green = each\_pixel[1] - max(list\_pixel\_0\_2)

detection\_values.append(detect\_green)

counter = 0

for each\_row in new\_image\_array

for each\_pixel in each\_row

#if that doesn’t work, put detect\_green = each\_pixel[1] - max(list\_pixel\_0\_2)

if detection\_values[counter] > 100 :

each\_pixel[0] = 0

each\_pixel[1] = 255

each\_pixel[2] = 0

each\_pixel[3] = 255

else:

each\_pixel[0] = 0

each\_pixel[1] = 0

each\_pixel[2] = 0

each\_pixel[3] = 255

counter += 1

counter += 1

return new\_image\_array

def red\_detect(image\_array):

detection\_values = []

new\_image\_array = image\_array

for each\_row in image\_array

for each\_pixel in each\_row

list\_pixel\_0\_2 = [each\_pixel[0], each\_pixel[2]]

detect\_red = each\_pixel[1] - max(list\_pixel\_0\_2)

detection\_values.append(detect\_red)

counter = 0

for each\_row in new\_image\_array

for each\_pixel in each\_row

if detection\_values[counter] > 100 :

each\_pixel[0] = 0

each\_pixel[1] = 255

each\_pixel[2] = 0

each\_pixel[3] = 255

else :

each\_pixel[0] = 0

each\_pixel[1] = 0

each\_pixel[2] = 0

each\_pixel[3] = 255

counter += 1

counter += 1

return new\_image\_array

def blue\_detect(image\_array):

detection\_values = []

new\_image\_array = image\_array

for each\_row in image\_array

for each\_pixel in each\_row

list\_pixel\_0\_2 = [each\_pixel[0], each\_pixel[2]]

detect\_blue = each\_pixel[1] - max(list\_pixel\_0\_2)

detection\_values.append(detect\_blue)

counter = 0

for each\_row in new\_image\_array

for each\_pixel in each\_row

if detection\_values[counter] > 100 :

each\_pixel[0] = 0

each\_pixel[1] = 255

each\_pixel[2] = 0

each\_pixel[3] = 255

else :

each\_pixel[0] = 0

each\_pixel[1] = 0

each\_pixel[2] = 0

each\_pixel[3] = 255

counter += 1

counter += 1

return new\_image\_array