

۱. برای خواندن تصویر از هر دو کتابخانه **matplotlib** و **opencv** استفاده میشود، ابتدا یک نوتبوک با فرمت **ipynb** ساخته و تصویر فوق را به کمک این دو کتابخانه نمایش دهید. فرق این دو کتابخانه برای نمایش دادن تصویر در چیست؟ خروجی کتابخانه **matplotlib** را مانند **opencv** کنید.

تفاوت اصلی **opencv** و **matplotlib** در این است که در **opencv** ترتیب سه کانال عکس BGR هست ولی در **matplotlib** به ترتیب RGB هست برای اینکه خروجی **matplotlib** مثل **opencv** بشه باید جای کانال های B,R را در **matplotlib** با همدیگر **swap** کرد که کد آن به صورت زیر است:

```
import matplotlib.pyplot as plt
import matplotlib.image as image
import numpy as np
import cv2

img=image.imread(r"Q1.jpg")

pixel_map=img.copy()
red = pixel_map[:, :, 2].copy()
blue = pixel_map[:, :, 0].copy()
pixel_map[:, :, 0] = red
pixel_map[:, :, 2] = blue
img=pixel_map

plt.imshow(img)
cv2.imshow("image",img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

ب) خروجی **img.shape** را تفسیر کنید **img** (متغیری است که تصویر خوانده شده در آن قرار دارد)

وقتی **img.shape** را برای همین مثال پرینت کنیم سه عدد مطابق روبه رو نمایش داده می شود

(3, 1020, 680) که **height** و **width** و تعداد کانال های عکس هست که در اینجا سه کانال آبی و قرمز و سبز داریم.

ج) در ابتدا با استفاده از **os.mkdir** یک پوشه درست می کنیم و بعد با استفاده از کتابخانه **glob** تمام مسیر فایل هایی که در **directory** مشخص پسوند **jpeg** دارند را در یک آرایه ذخیره می کنیم وبعد یک حلقه روی آرایه مورد نظر میزنیم و تصاویر را سیاه و سفید می کنیم و به همراه ابعادشان در پوشه مورد نظر ذخیره می کنیم.

۲. در ابتدا دو تا حلقه تو در تو می زنیم تا تصویر مورد نظر را با استفاده از سینتکس `img[:,:]` کراپ کنیم و هم چنین چک می کنیم اگر جمع پیکسل های تصویر کراپ شده مخالف با ۰ است در آرایه `images` ذخیره شود. بعد هم تصاویر را از پوشه Q2 میخوانیم و فانکشن `crop` رو همگی تصاویر `call` می کنیم و همگی را در آرایه `All_cropped_images` ذخیره می کنیم و بعد با دو حلقه تودرتو از هر عدد 4 نمونه رندوم انتخاب می کنیم و تصویر مورد نظر را می سازیم.

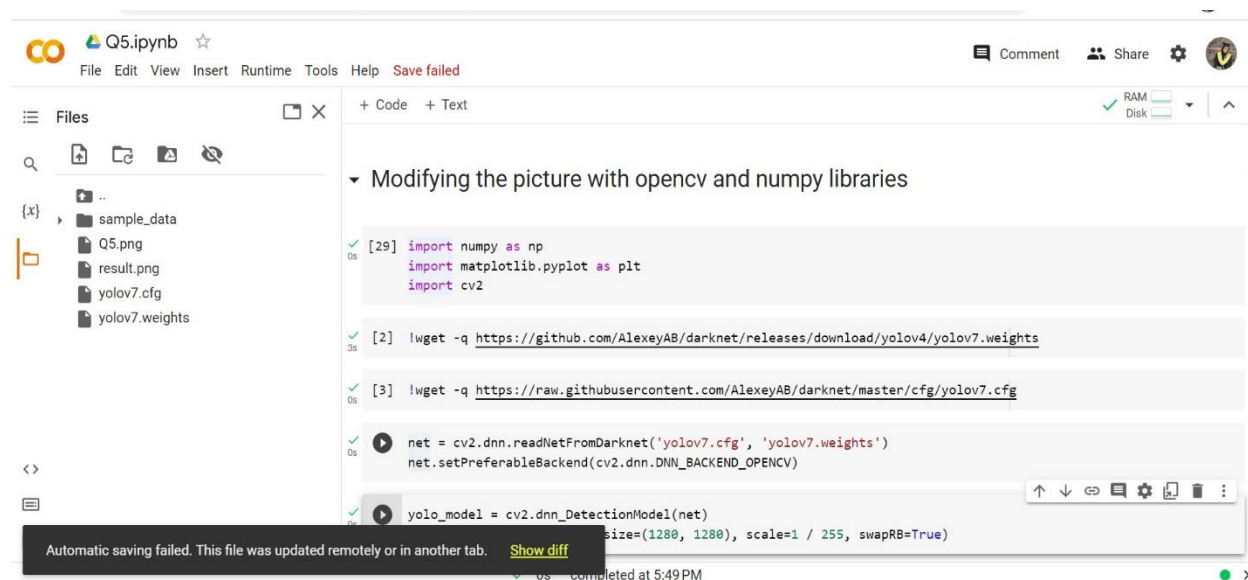
۳.الف) در ابتدا با استفاده از `np.random.randint` آرایه دوبعدی `numpy` را با اعداد رندوم پر می کنیم و بعد برای پیمایش کردن `matrix` دو حلقه تودرتو می زنیم و بعد در آرایه `findhasdigit` تمام رقم های یک عدد ذخیره می کنیم و بعد با یک حلقه دیگر به دیکشنری خود اضافه می کنیم و در نهایت دیکشنری را پرینت می کنیم.

ب) المنت هایی از ماتریس که روی یک خط اریب قرار دارند جمع محور `X` و محور `Y` ان ها با هم برابر است به همین یک آرایه به نام `trav` داریم که هر المنت ان نیز یک آرایه است و بعد دو حلقه تودرتو میزنیم و `X,Y` را با هم جمع می کنیم اگر زوج بود چون اول المانی که محور `X` ان بزرگتر است زودتر پیمایش می شود باید در اول لیست `insert` شود و بعد روی همون آرایه `trav` حلقه میزنیم و المنت ها رو چاپ می کنیم.

۴.الف) طبق روابط پیش میریم این سوال غیر از `broadcasting` نکته خاصی نداره

ب) برای این سوال دو حلقه تو در تو داریم که حلقه درونی سطرهای ما ثابت است و ماتریس را روی ستون میلغزاند و در حلقه بیرونی سطرها تغییر می کنند و در نهایت جواب در یک ماتریس 3×3 ذخیره میشود.

۵. برای این سوال از محیط `colab` استفاده می کنیم که عکس آن مطابق زیر است:



```
[29] import numpy as np
import matplotlib.pyplot as plt
import cv2

[2] !wget -q https://github.com/AlexeyAB/darknet/releases/download/yolov4/yolov7.weights

[3] !wget -q https://raw.githubusercontent.com/AlexeyAB/darknet/master/cfg/yolov7.cfg

net = cv2.dnn.readNetFromDarknet('yolov7.cfg', 'yolov7.weights')
net.setPreferableBackend(cv2.dnn.DNN_BACKEND_OPENCV)

yolo_model = cv2.dnn_DetectionModel(net)
size=(1280, 1280), scale=1 / 255, swapRB=True
```

Automatic saving failed. This file was updated remotely or in another tab. [Show diff](#)

Completed at 5:49 PM

در ابتدا تصویر را می خوانیم و \max, \min و میانگین پیکسل ها را نمایش می دهیم و بعد cv2.rectangle را از هر المان ارایه boxes به دست می آوریم و بعد با استفاده از cv2.rectangle کادر های مورد نظر را می کشیم. تصویر result به صورت زیر است:

