" به نام خدا "

مروارید لعل نور 9623097

گزارش تمرین سری 0 درس پردازش تصویر

---

: Q2

در این سوال ابتدا نیاز داریم یک لیست 100 تایی از اعداد رندوم بین -2 تا 7 داشته باشیم که اینکار را توسط تابع random_generate انجام میدهیم . این تابع ورودی نمیگیرد و لیست مورد نظر را تولید کرده و برمیگرداند . سپس تابع count را تعریف کردیم که لیست و عضو مورد نظر را به عنوان ورودی میگیرد و تعداد وقوع عضو ورودی در لیست ورودی را برمیکرداند . روند کار این برنامه به ان صورت است که ابتدا یک دیکشنری به نام tmp ایجاد میکند که در ان هر عدد و تعداد وقوع ان در لیست اصلی وارد میشود و سپس توسط یک حلقه میان اعضای این دیکشنری گردش انجام میدهد و به تعداد value هر عدد در ان ( تعداد وقوع ) این عدد را به لیستی جدید اضافه میکند و در انتها این عدد و لیست جدید را وارد دیکشنری به نام dct میکند که در واقع خروجی برنامه ماست . به عنوان مثال آرایه ی صد تایی اولیه مطابق زیر است :

[-1, -2, 0, 7, -1, 5, 6, 7, 0, 5, 3, 5, 1, 3, -1, 0, 0, 4, 5, 0, 4, 3, 5, 6, 5, -2, 0, 4, 1, 1, 1, -2, 2, 1, 3, 3, 2, -1, -1, 2, 2, 4, 1, 6, 1, 6, 3, 1, 5, 6, 1, 2, 0, -2, 6, -2, 3, 3, 0, 5, 4, -2, 1, 7, 3, 4, 4, -2, -2, -2, 0, 2, 7, 4, 6, 3, 6, 3, 4, -2, 5, 6, 3, 7, 1, 5, 6, 7, 3, -1, 2, -2, 5, 5, -1, 5, 1, 2, 4, -1]

دیکشنری tmp به این شکل است ( هر عدد با تعداد تکرارش):

{7: 6, 6: 10, 5: 13, 4: 10, 3: 13, 2: 8, 1: 12, 0: 9, -1: 8, -2: 11}

توسط tmp دیکشنری dct را میسازیم :

{1: [7, 7, 7, 7, 7, 7], 7: [6, 6, 6, 6, 6, 6, 6, 6, 6, 6], 17: [5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5], 30: [4, 4, 4, 4, 4, 4, 4, 4, 4, 4], 40: [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3], 53: [2, 2, 2, 2, 2, 2, 2, 2], 61: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], 73: [0, 0, 0, 0, 0, 0, 0, 0, 0], 82: [-1, -1, -1, -1, -1, -1, -1, -1], 90: [-2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2]}

: Q3

در این سوال ابتدا تابع computeGCD را تعریف کردیم که ب م م دو عدد را محاسبه میکند و سپس تابع topositive که قدر مطلق عدد ورودی را برمیگرداند و تابع compute که دو عدد دریافت میکند و هر کدام را بر ب م م دو عدد تقسیم میکند و اعداد جدید را باز میگرداند که در واقع همان کاری است که در ریاضیات برای ساده کردن کسر ها استفاده میشود . روند کلی کار نیز به این صورت است که ابتدا دو ورودی دریافت میشود و چک میکنیم که اگر مخرج 0 با بی نهایت نبود و یا اگر در تبدیل به ورودی به int مشکلی نداشتیم ( در صورت وارد کردن هر چیزی به جز عدد صحیح به valueError exception بر میخوریم و None چاپ میشود) اعداد ساده شده ی صورت و مخرج نمایش داده میشوند و در غیر این صورت None به عنوان خروجی نمایش داده میشود .

مثال ها :

Enter numerator:

6

Enter denumrator:

-39

(13-,2)

---

Enter numerator:

-26

Enter denumrator:

91

(-2,7)

---

Enter numerator:

21

Enter denumrator:

14

(3,2)

---

Enter numerator:

18

Enter denumrator:

0

None

---

Enter numerator:

True

Enter denumrator:

6

None

---

Enter numerator:

19

Enter denumrator:

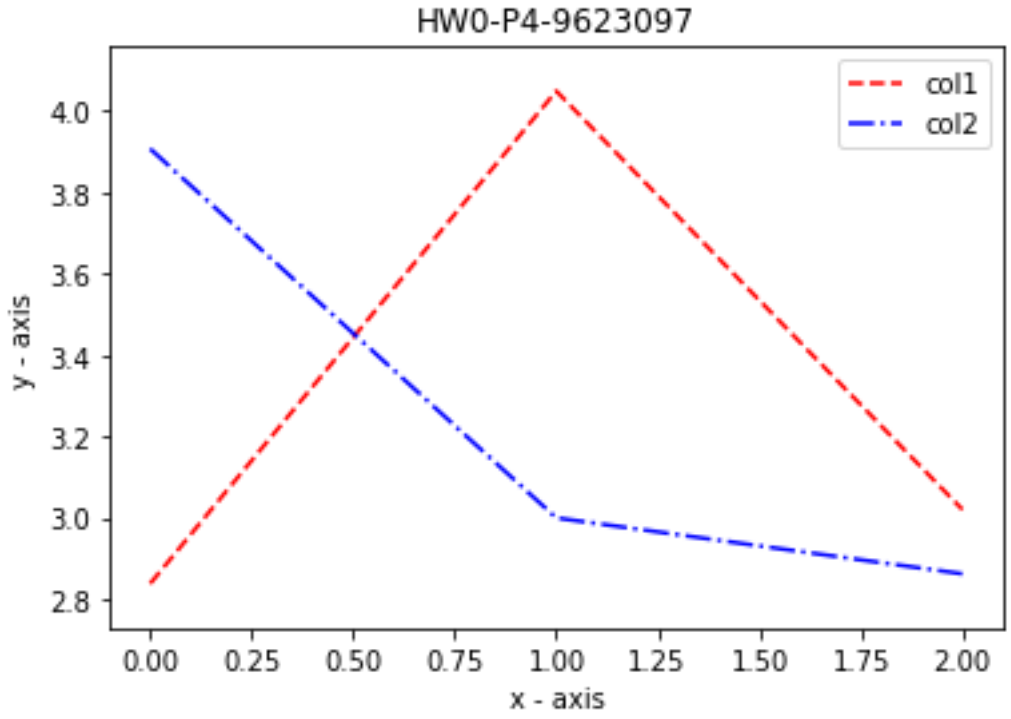
2

(19,2)

: Q4

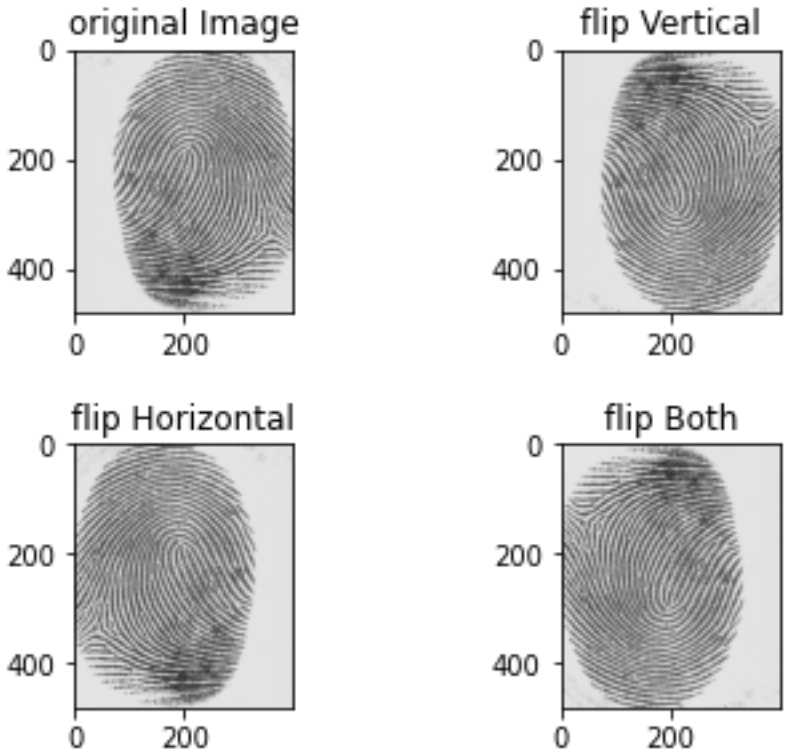
در این سوال همه کار ها در تابع random_matrix انجام میشود ، .ورودی این تابع به ترتیب : حد پایین بازه رندوم ، حد بالای بازه رندوم ، تعداد سطر های ماتریس ، تعداد ستون های ماتریس است .ابتدا توسط دستور random.uniform ماتریسی با سایز داده شده و در بازه ی مشخص شده تعریف میکنیم و آنرا چاپ میکنیم ، سپس با دستور A[:,0] که نشان دهنده ستون اول ماتریس است آن را رسم مکنیم ( مطابق سوال با رنگ قرمز و با خط چین و لیبل گزاری مناسب ) و همین کار را بری ستون دوم هم انجام میدهیم و بعد از ان نام نمودار که HW0-P4-9623097 است را تعیین میکنیم و سپس نام محور های x و y را مشخص میکنیم و در آخر برای نسبت یکسان محور عرض و طول از دستور plt.axis('equal') استفاده میکنیم . نتیجه به این صورت خواهد بود :

```
[[2.83782965 3.90692483 3.57163134 3.97701808]
 [4.04885491 2.99932583 2.36498242 3.48814742]
 [3.01985654 2.86191094 2.64394684 3.30287461]]
```



: Q5

در این قسمت ابتدا با استفاده از کتابخانه cv2 و دستور imread عکس موردنظر را میخوانیم و سپس با استفاده از دستور flip و تعیین دومین ورودی این تابع با 1 و 0 و -1 عکس خوانده شده ر ا نسبت به محور افقی و عمودی و در آخر هر دو قرینه میکنیم و پس از آن با دستور plt.subplots(2, 2) یک آرایه ی 2 در 2 از نمودار هایی که قرار است بکشیم درست میکنیم و سپس به هر یک از درایه های این آرایه یکی از عکس های قرینه شده و عنوان مربوط به آن را نسبت میدهیم و در آخر این بخش برای تعیین اندازه مناسب نمودار ها از یکدیگر از دستور subplots_adjust استفاده میکنیم . نتیجه این بخش به این شکل خواهد شد :

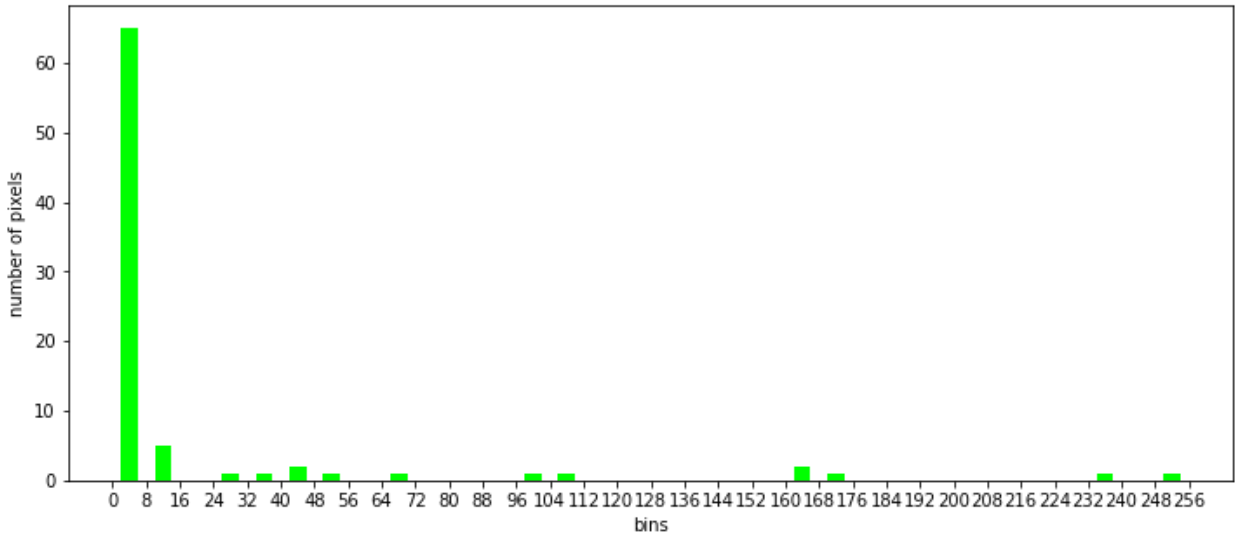



در بخش بعدی باید نمودار فراوانی را رسم کنیم ، برای محاسبه مقادیر از دستور cv2.calcHist و برای رسم آن نیز از plt.hist استفاده میکنیم . ورودی های این توابع مطابق زیر اند :

```
hist = cv2.calcHist([originalImage],[0],None,[256],[0,256])
plt.hist(hist,np.arange(0,257,8)  ,color =['lime'],rwidth = 0.5)
```

که برای بدست آوردن متغیر hist ورودی های تابع مورد نظر را به این صورت تعیین کرده ایم که ورودی اول عکس مورد نظر را تعیین میکند و ورودی دوم تعداد کانال ها ( که در اینجا یک کانال است ) ، در ورودی سوم باید ماسک مورد نظر برای تصویر مشخص شود که در اینجا از ماسک خاصی استفاده نکرده ایم و در ورودی چهارم سایز هیستوگرام که 256 است را تعیین میکنیم( بدلیل قرار گرفتن داده های هر پیکسل از 0 تا 255 ) و در آخر بازی تغییرات داده ها که در قسمت قبل هم به آن اشاره شد را تعیین میکنیم .

برای رسم این نمودار از plt استفاده کرده ایم ورودی های دستور مورد نظر در ابتدا آرایه ای است که در قسمت قبل از تصویر بدست آورده ایم و سپس نحوه ی دسته بندی و تعیین bin ها است که طبق خواسته سوال از 0 تا 256 و به صورت 8 تا 8 تقسیم شده اند و سپس رنگ میله ها و عوض نسبی آنها که 0.5 است تعیین شده است . نمودار بدست آمده در صفحه بعد آورده شده است .




در بخش بعد اطلاعات خواسته شده در مورد تصویر را چاپ میکنیم که شامل اندازه عکس ، تعداد پیکسل ها و .... است . با استفاده از دستور originalImage.shape به طول و عرض و تعداد کانال های تصویر پی میبریم ( که در اینجا چون gray scale تنها یک کانال داریم که انگار سه بار تکرار شده است) ، با استفاده از دستور sys.getsizeof به اندازه تصویر بر حسب بایت پی میبریم که در ادامه به کیبی بایت هم تبدیل شده و نمایش داده میشود :

```
 Height of photo : 480
 Width of photo :400
 Number of pixels :192000
 Image size 192000
 Maximum RGB value in this image 251
 Minimum RGB value in this image 27
 Pixel in [100,50] location contains : 227
 Data type of a pixel uint8
 Image size is : 192112 (Bytes)
 so we need 187.609375 kibibyte (KiB) of memory if we want to save it (
size / 1024)
```