**دانشگاه علم و صنعت**

**تمرین پنجم مبانی بینایی کامپیوتر**

**نام و نام خانوادگی:**

فرناز خوش دوست آزاد

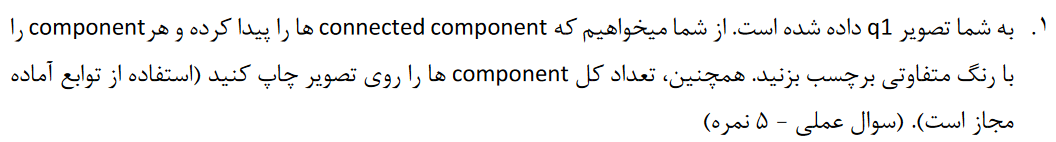
**شماره دانشجویی:**

99521253

**نام استاد:**

دکتر محمدرضا محمدی

1)

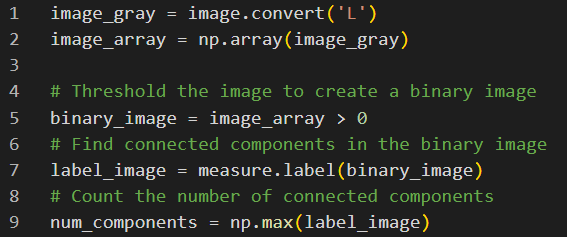


پاسخ:

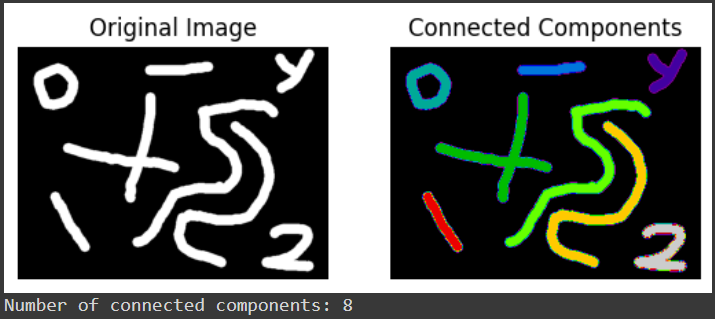
همانطور که در فایل hw5\_q1.ipynb نیز موجود است در ابتدا کتابخانه های مورد نظر را import کرده ام و سپس از پوشه images عکس مورد نظر را با کتابخانه PIL خوانده ام و سپس با استفاده از plt تصویر را به هر دو صورت رنگی و gray\_scale خوانده ام که واضح است که تفاوتی با یکدیگر ندارند که در تصویر زیر نیز به خوبی مشخص است:

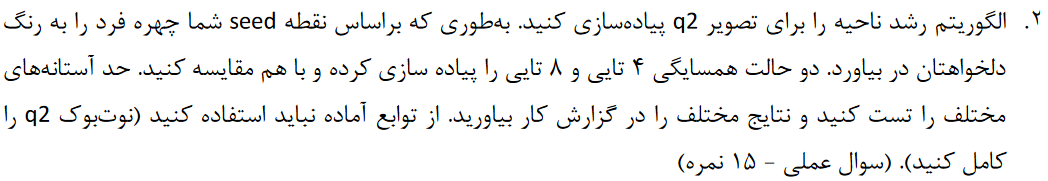


سپس همانطور که در کد زیر نیز مشخص است و کامنت گذاری شده است در ابتدا تصویر به gray تبدیل شده است، سپس تصویر را به یک آرایه ای از numpy تبدیل کرده ام و سپس threshold = 0 در نظر گرفته ام که به صورت boolean مقادیر را بر برای دو کلاس باینری بر می گرداند و سپس با استفاده از measure.label تصاویر binary شده را در label\_image ذخیره کرده ام و سپس max آن را برگردانده ام که همان تعداد components ما می باشد و در آخر آن را پایین عکس های خود print کرده ام.



که نتیجه کد خود را در آخر با استفاده از plt نشان داده ام که به صورت زیر است:



2) 

پاسخ:

چالش:

چالشی که در این سوال برخوردم، این بود که در ابتدا با استفاده از cv2 به رسم اشکال با threshold های مختلف پرداختم، اما همانطور که در انتهای نوتبوک نیز مبرهن است، خطوطی آبی بدون دلیل در نقاط x و y برای seed\_point نیز در آن رسم شده است که دلیل آن را هنوز پیدا نکرده ام، اما با استفاده از plt توانستم تصاویر بدون خطوط درستی را تولید کنم.

شرح کد:

در ابتدا برای پیدا کردن جای حدودی seed\_point کدی ردم تا جای خدودی آن با شکل لوزی آبی دیده شود و سپس به سراغ فانکشن خواسته شده رفتم.

در کد زده شده در فانکشن segment1 در ابتدا کپی ای از تصویر ورودی می گیرم و سپس offset مربوط به connectivity 4 و 8 connectivity را نوشتم. و سپس نقاط یا نقطه ی seed\_point را به استک اضافه می کنم و تا زمانی که استک من خالی نشود لوپی را در آن تکرار می کنم.

در این لوپ با توجه به ورودی در ابتدا offset های خود را از میان دو کلاس انتخاب می کنم. سپس با توجه به seed\_point نقاط کناری آن را در صورتی که شرط گذاشته شده را برآورده کند در صورت unvisited بودن به داخل استک اضافه می کنم و دوباره همین روند را ادامه می دهم که کد آن نیز به صورت زیر می باشد. شرط نوشته شده بدین صورت است که سه کانال نقطه ی seed\_point را می گیرم و سپس اختلاف آن را با سه کانال نقطه ی مورد نظر حساب می کنم و در صورتی که میانگین آنها از threshold کمتر باشد، آن را به نقطه ی قرمز رنگ تبدیل می کنم و همان نقطه را visited در نظر می گیرم و به داخل استک خود نیز اضافه می کنم.

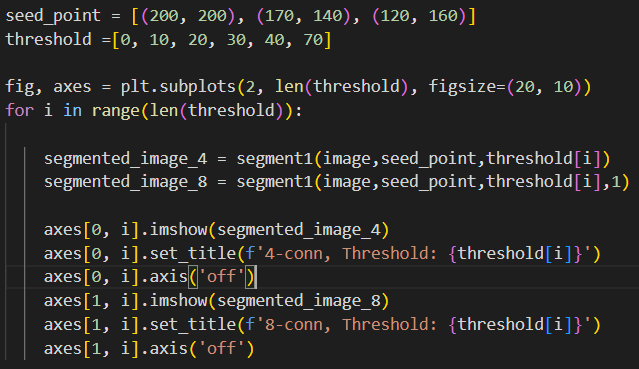
حالت های مختلف:

در داخل نوتبوک نیز موارد مختلف نشان داده شده اند.

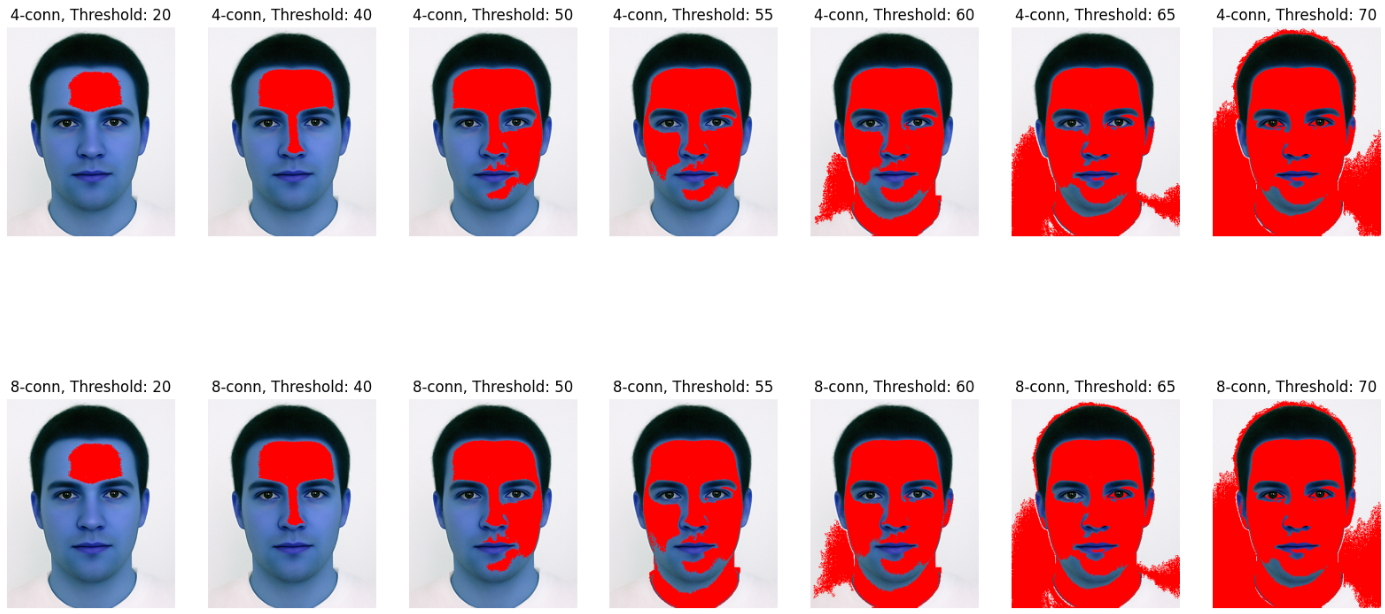
برای مثال در ابتدا چند نقطه ی Seed\_point را اضافه کرده ام که نتیجه ی آن نیز به صورت زیر شد:



نقاط گرفته شده اولیه در تصاویر بالا نیز به شرح زیر است که سه نقطه می باشند.



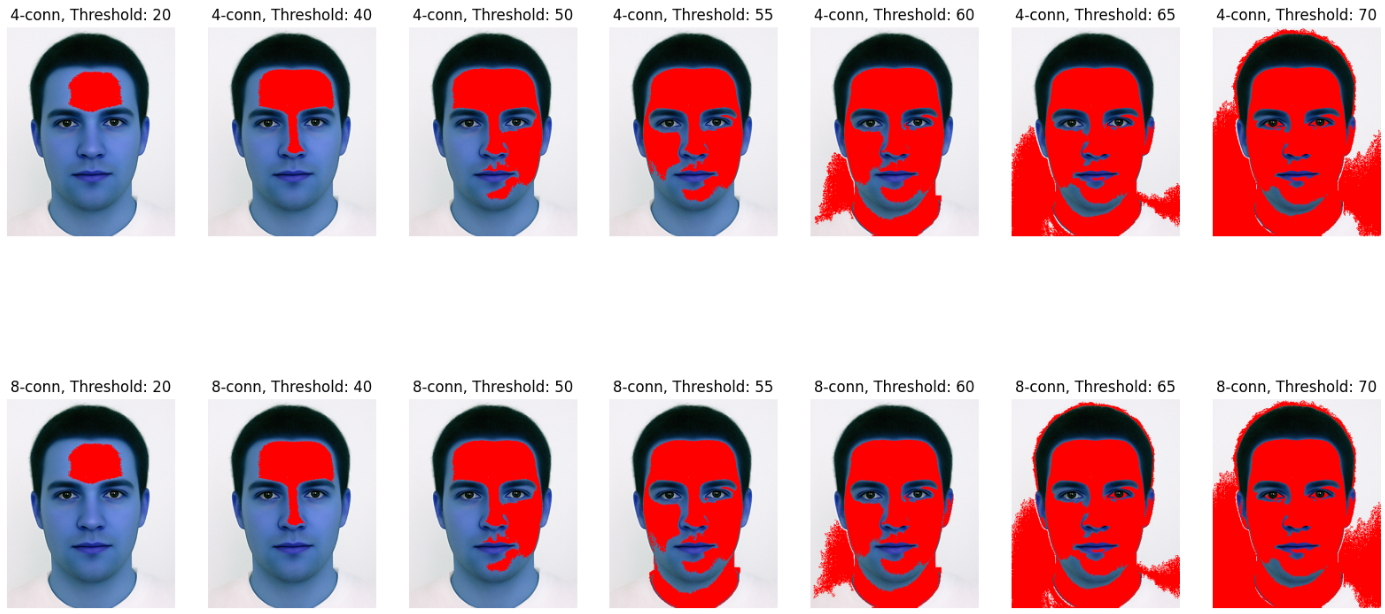
حال تنها برای نقطه ی (150و 150) تصاویر به دست آمده را رسم می کنم که نتیجه به صورت زیر است:



می بینیم که در تصاویر بالا قسمت زیادی از تصویر پشت زمینه شخص نیز به عنوان تصویر فرد تشخیص داده می شود، لذا نقطه ی دیگری را برای نتیجه ی دیگر انتخاب کردم.

در تصویر اولی که گذاشتم دیده می شود که تصویر 4 connectivity و 8 connectivity فرقی ندارند اما در تصویر بالا برای threshold = 55 می بینیم که به در صورت استفاده از 8 connectivity گردن فرد نیز تشخیص داده می شود. که به دلیل جستجوی منحصر به فرد آن است.

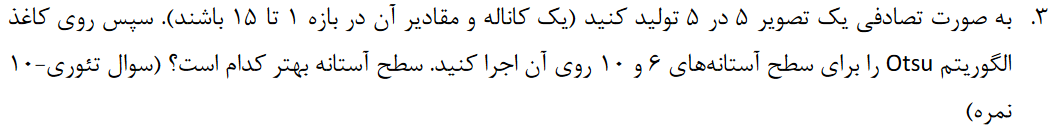
حال می بینیم با انتخاب نقطه ی (200و 200) تصویر زیر به دست آمد.



همانطور که دیده می شود با انتخاب نقطه ای دیگر نتیجه ای بهتری نیز به دست آوردیم. که از میان تصاویر بالا تصویر وسط بهترین threshold ما می باشد.

در ادامه نوتبوک نیز چالشی که به آن برخورده بودم را آورده ام که هنوز دلیلی برای آن پیدا نکرده ام.

3)



پاسخ:

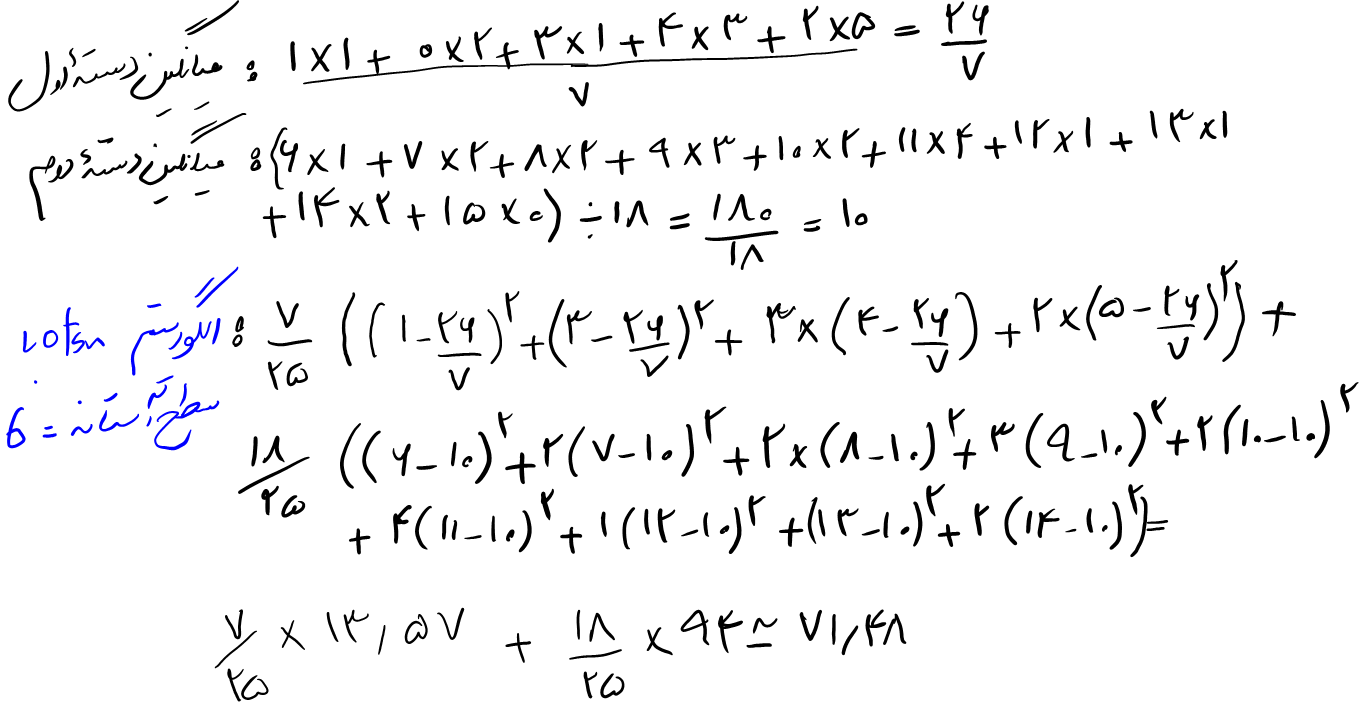
| 4 | 3 | 7 | 6 | 12 |
| --- | --- | --- | --- | --- |
| 5 | 13 | 4 | 5 | 10 |
| 11 | 11 | 4 | 10 | 11 |
| 9 | 9 | 7 | 1 | 14 |
| 8 | 9 | 14 | 11 | 8 |

در ابتدا هیستوگرام تصویر را حساب می کنیم که به صورت زیر است

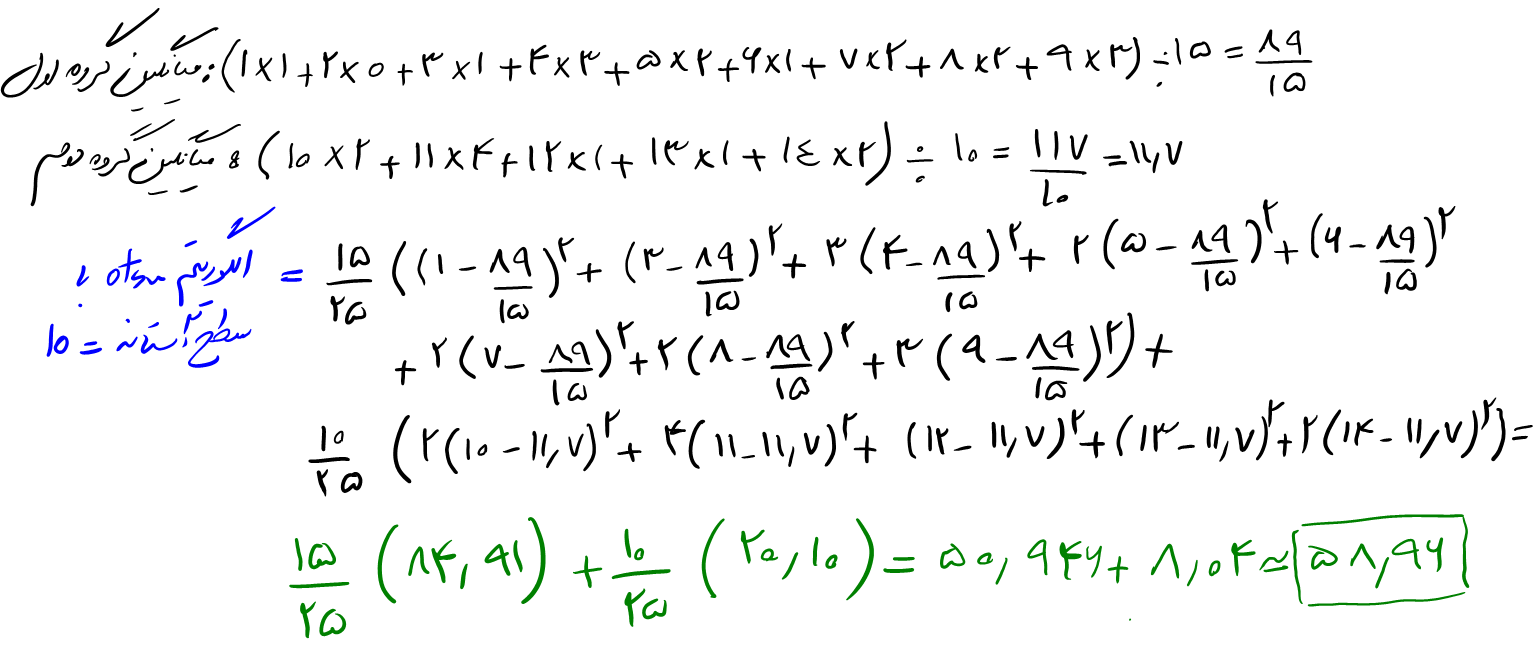
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 2 | 1 | 1 | 4 | 2 | 3 | 2 | 2 | 1 | 2 | 3 | 1 | 0 | 1 |

حال با سطح آستانه 6 دو کلاس خواهیم داشت که در کلاس اول تنها یک تا 5 و در کلاس دیگر مقادیر 6 تا 15 قرار دارند. حال باید با توجه به فرمول زیر مقادیر را حساب کنیم که داریم:

برای سطح آستانه = 6

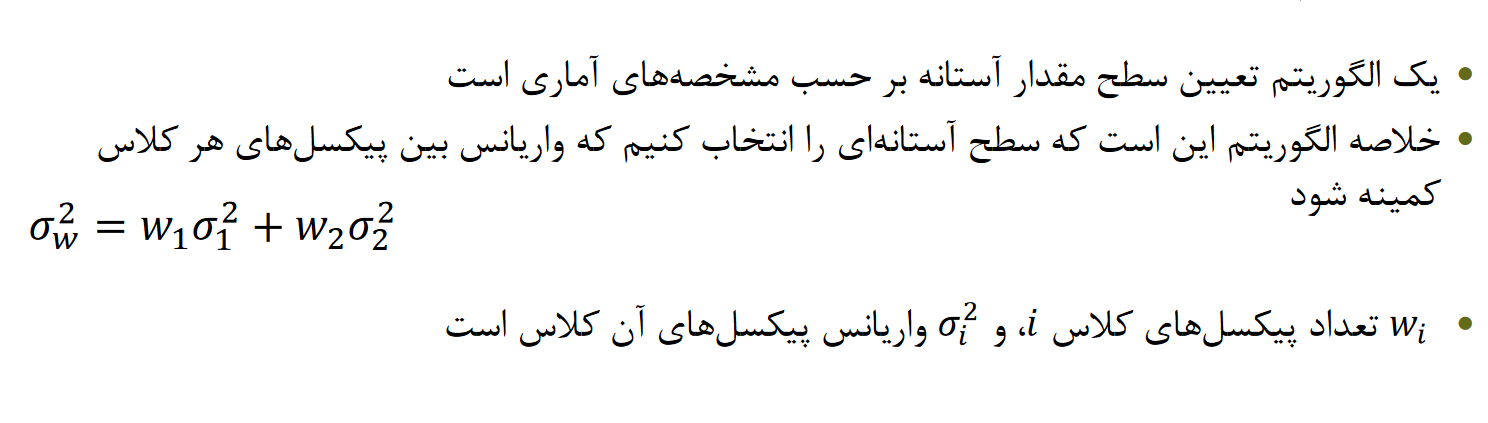


و برای سطح آستانه = 10

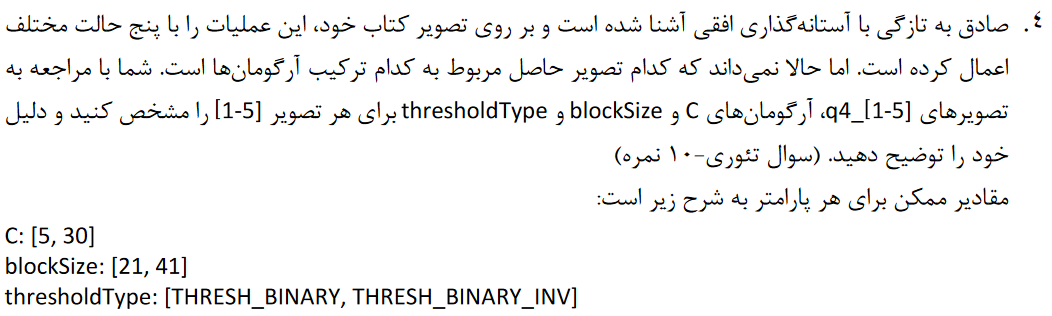


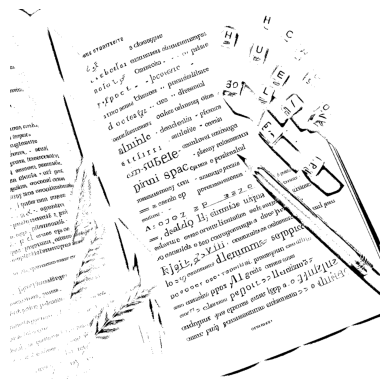
با توجه به جواب های به دست آمده می بینیم که برای مقدار دهی رندوم من سطح آستانه 10 جواب بهتری به ما می دهد و تقریبا 14 واحد کمتر از زمانی است که سطح آستانه را 6 قرار می دهیم که برای مقادیر رندوم دیگر ممکن است برعکس و یا حتی این دو مقدار برابر هم شوند.

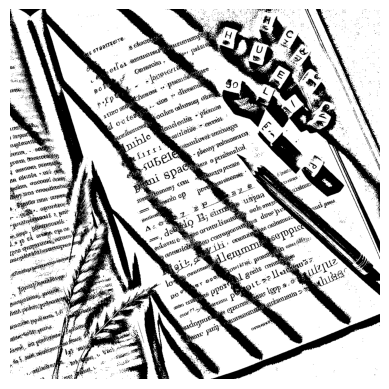
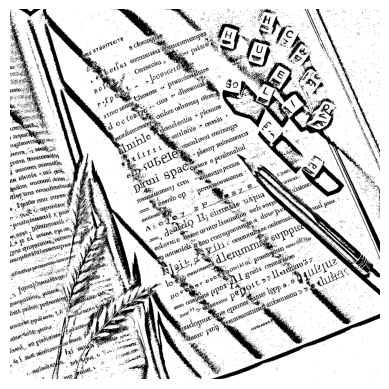
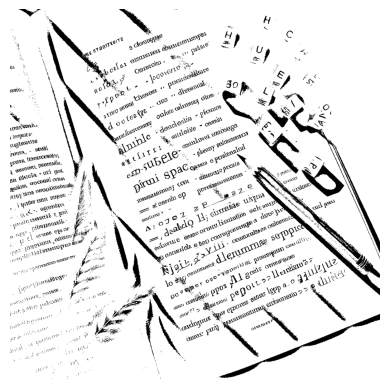
من برای به دست آوردن این فرمول از فرمول داخل پی دی اف استفاده کردم که به شرح زیر است:

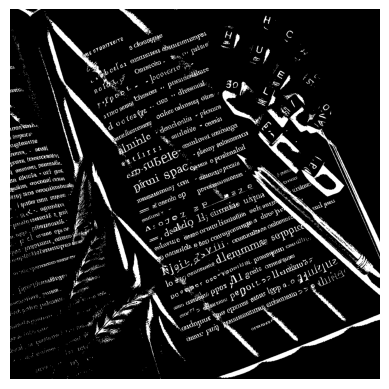


4)









پاسخ:

می دانیم که آستانه گذاری افقی با کمک تابع cv2.adaptive Threshold صورت می گیرد که برخی ورودی های آن عبارتند از thresholdType و BlockSize و C. پارامتر بیانگر آن است که اگر mean + c < src باشد به رنگ سفید نمایش داده شود و یا در صورت برعکس بوده به صورت مشکی دیده می شود. پارامتر دوم نیز ابعاد پنجره ای است که برای محاسبه میانگین در نظر گرفته می شود و هر چه ابعاد بزرگتر باشد به آستانه گذاری سراسری نزدیکتر می شود. لذا باید ابعاد پنجره ی متناسب برای خود را انتخاب کنیم و پارامتر آخر نیز برای شیفت دادن آستانه نسبت به میانگین پنجره می باشد.

در تصویر اول، سمت چپ پایین تصویر نوشته ها واضح نیستند و به خوبی binary نشده اند. بنابراین این تصویر از blockSize بزرگتر از 41 استفاده می کند. همچنین این تصویر نسبت به سایرین کمی روشنتر است و احتمالا c کوچکتری داشته است. (C = 5)

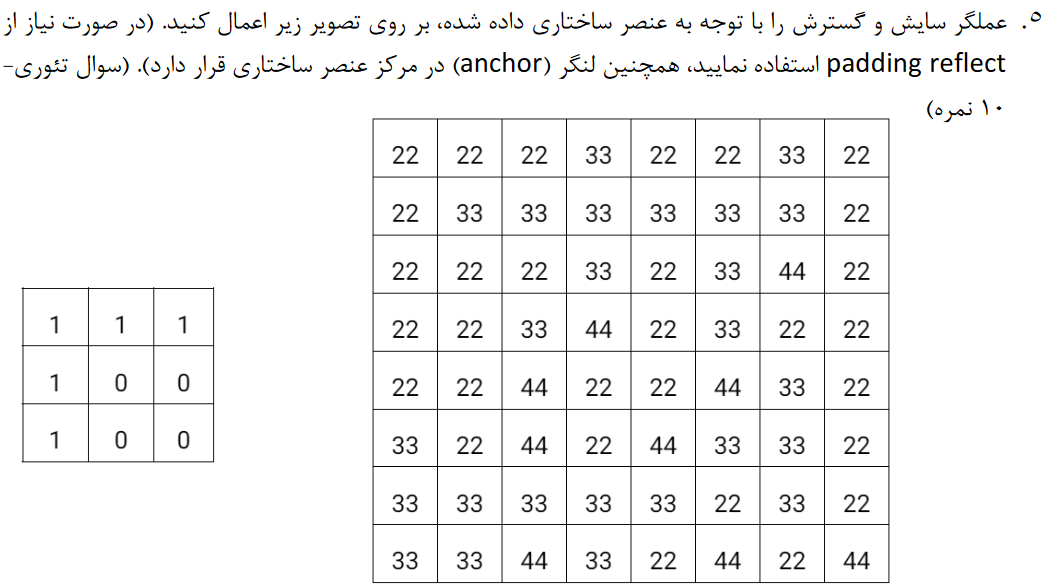
در تصویر دوم همه نواحی با دقت قابل قبولی binary شده اند، پس پنجره ی کوچکتری استفاده شده است(block size 21)، اما تصویر به طور کلی کمی تیره تر از حالت عادی به نظر می رسد. می توان حدس زد که c = 30 مقدار بیشتری داشته است و باعث شده تصویر تیره تر از حالت میانگین باشد.

در تصویر سوم نیز نواحی به خوبی binary شده اند، پس پنجره کوچکتری داشته(blockSize = 21). همچنین در مقایسه با تصویر دوم، به نظر می رسد که خطوط حاشیه مداد و خطوط مورب در حالت عادی باشد و پررنگ تر یا کمرنگ تر از حد معمول نیستند! پس مقدار C = 5 می باشد تا آستانه ما حدودا همان میانگین پنجره باقی بماند.

در تصویر چهارم مجددا ناحیه پایین چپ حالت محوشدگی دارد. پس پنجره ی بزرگتری داشته است (blockSize = 41)، همچنین خطوط مداد و مورب پررنگ تر از حالت عادی به نظر می رسند. به همین خاطر احتمالا مقدار c بزرگتری داشته اند.(c = 30).

در تصویر آخر نیز که تصویری با پس زمینه ی مشکی می باشد متوجه می شویم که thresholdtype برابر با همان THREH\_BINARY\_INV است. لذا متوجه می شویم که روشنایی بیشتر از آستانه به صفر تناظر یافته و کمتر از آستانه به 255 تبدیل می شود. حال مجددا ناحیه ی سمت چپ پایین تصویر نیز به خوبی binary نشده است. لذا blockSize = 41 و c = 5 می باشد.

5)

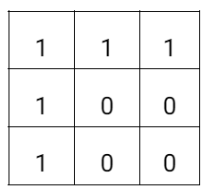


پاسخ:

در ابتدا reflect را بر روی تصویر اعمال می کنیم که داریم که نتیجه ی آن مانند شکل زیر می باشد:

| 22 | 22 | 33 | 22 | 22 | 33 | 22 | 22 | 22 | 22 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 22 | 33 | 22 | 22 | 33 | 22 | 22 | 22 | 22 |
| 22 | 22 | 33 | 33 | 33 | 33 | 33 | 33 | 22 | 22 |
| 22 | 22 | 44 | 33 | 22 | 33 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 33 | 22 | 44 | 33 | 22 | 22 | 22 |
| 22 | 22 | 33 | 44 | 22 | 22 | 44 | 22 | 22 | 22 |
| 22 | 22 | 33 | 33 | 44 | 22 | 44 | 22 | 33 | 33 |
| 22 | 22 | 33 | 22 | 33 | 33 | 33 | 33 | 33 | 33 |
| 44 | 44 | 22 | 44 | 22 | 33 | 44 | 33 | 33 | 33 |
| 44 | 44 | 22 | 44 | 22 | 33 | 44 | 33 | 33 | 33 |

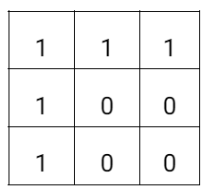
حال برای سایش برای هر پیکسل باید از بین مربع هایی که یک هستند، مینیمم آنها را انتخاب کرده و سپس به جای هر پیکسل جایگذاری کنیم.



که پاسخ آن به شرح زیر است:

| 22 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 33 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| 22 | 22 | 22 | 22 | 33 | 33 | 33 | 33 |

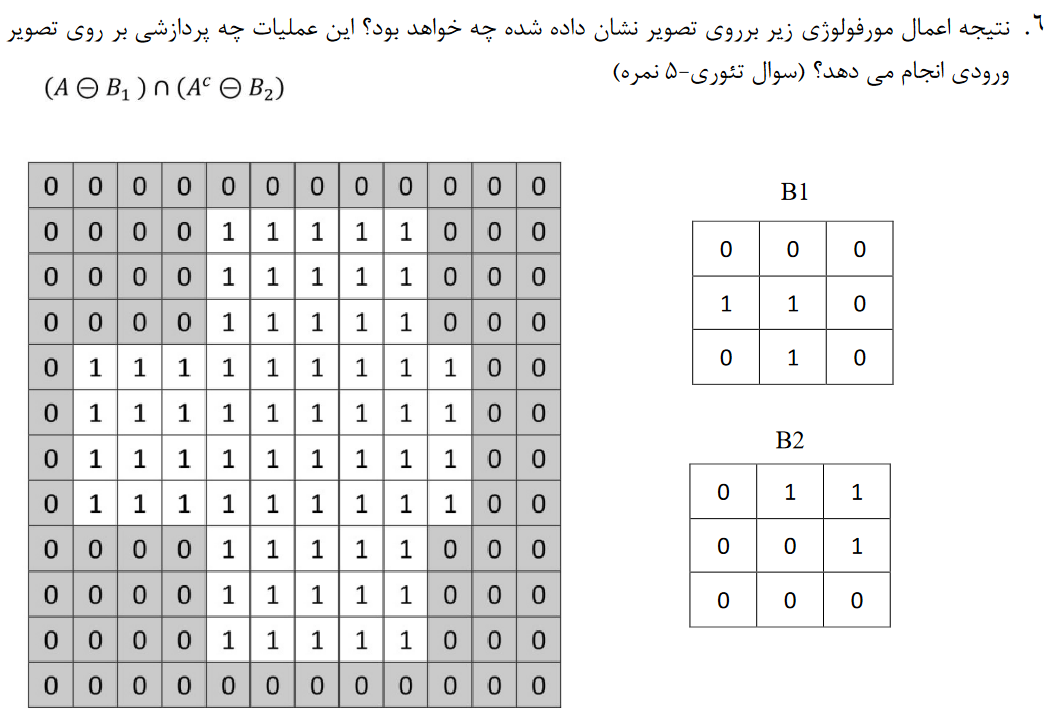
که این شکل حاصل از سایش توسط فیلتر بالا می باشد. حال برای گسترش باید در ابتدا فیلتر را 180 درجه دوران دهیم و سپس برای هر پیکسل از بین خانه هایی که یک می باشند، maximum را انتخاب می کنیم که فیلتر ما به صورت زیر می شود.



و پاسخ ما نیز پس از reflect به صورت زیر می شود:

| 33 | 33 | 33 | 33 | 33 | 33 | 33 | 33 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 44 | 44 | 44 | 33 | 33 | 33 | 33 | 33 |
| 22 | 33 | 44 | 44 | 44 | 44 | 33 | 33 |
| 33 | 44 | 44 | 44 | 44 | 44 | 44 | 22 |
| 33 | 33 | 44 | 44 | 44 | 44 | 44 | 33 |
| 33 | 33 | 33 | 44 | 44 | 33 | 44 | 33 |
| 44 | 44 | 44 | 44 | 44 | 44 | 44 | 33 |
| 44 | 44 | 44 | 44 | 44 | 44 | 44 | 33 |

که جدول بالا شکل حاصل از گسترش می باشد که ماکزیمم را برای هر یک به دست آورده ایم.

6)

پاسخ:

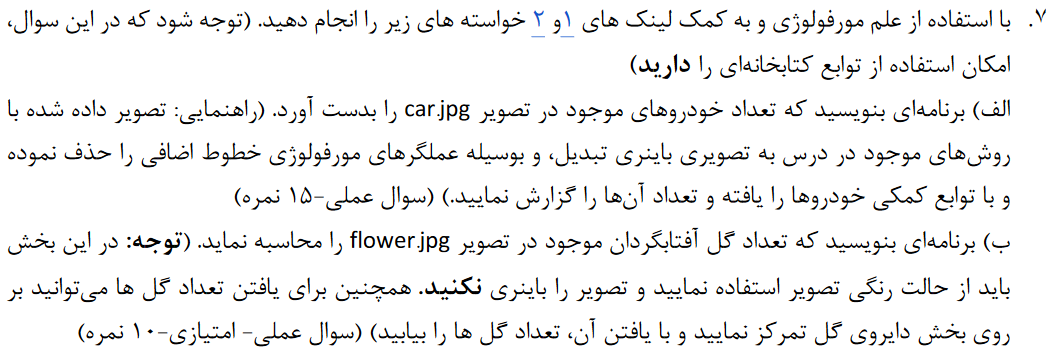
A, B1

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Ac , B2

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | & |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 |  |  |  |  |  |  |  |  |  | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 |  |  |  |  |  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

نقطه ی اشتراک در شکل بالا با رنگ قرمز نمایش داده شده است که نمایش می دهد که این فرمول گوشه های بالای سمت راست را نمایش می دهد.

7)

پاسخ:

در ابتدا تصویر اصلی و باینری و gray scale آن را به نمایش می گذاریم:

تصویر اصلی



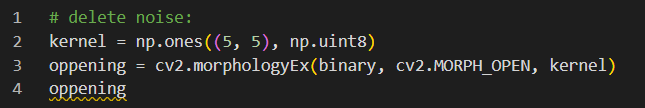
تصویر gray\_Scale:



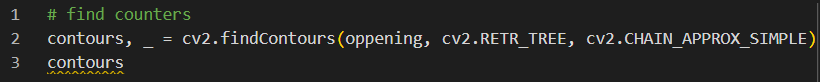
تصویر باینری:



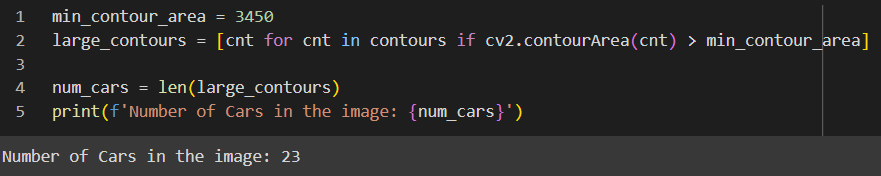
حال باید به سراغ نویز برویم و آن را با استفاده از عملیات های مورفولوژی حذف کنیم که کد آن به شرح زیر است:

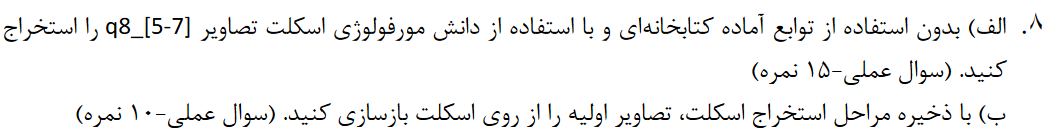


حال باید contours ها را در تصویر پیدا کنیم که کد آن به صورت زیر است:



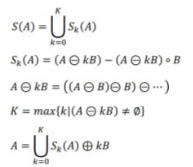
حال باید contours را بر اساس ناحیه فیلتر کرده و تعداد ماشین ها را می شماریم و مقدار دیگر را که در کد زیر است را با آزمون و خطا به دست می آوریم و نتیجه و کد در تصویر زیر آورده شده اند:



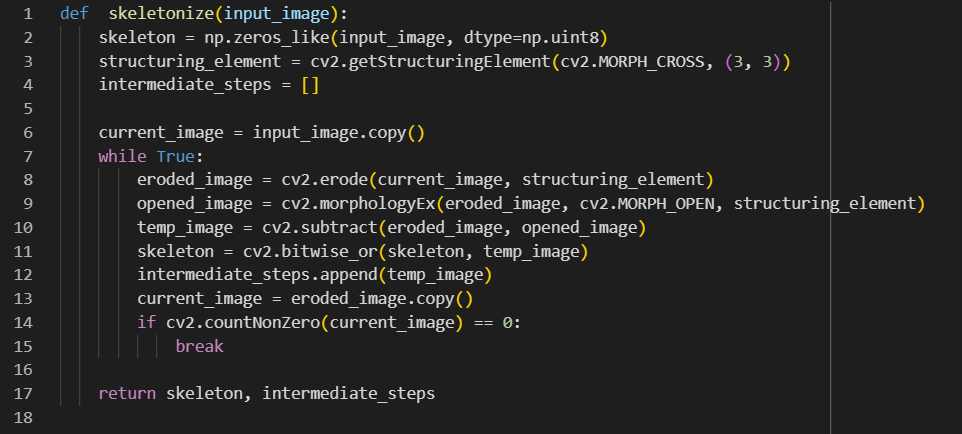
8)

پاسخ:

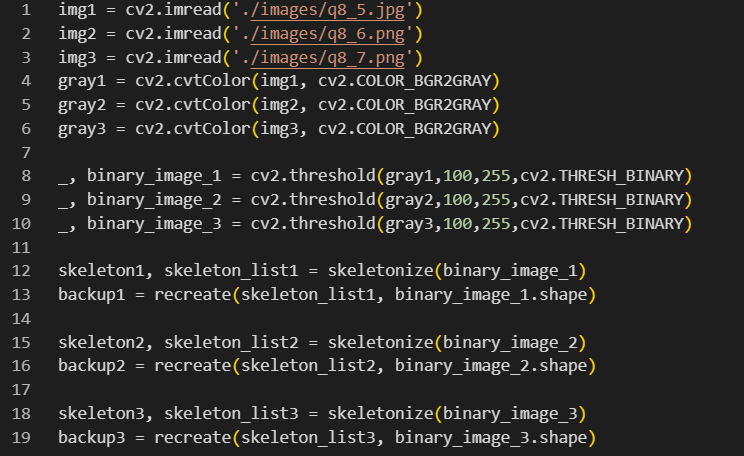
با توجه به فرمولی که در اسلاید ها اشاره شده است باید به صورت زیر عمل کنیم.



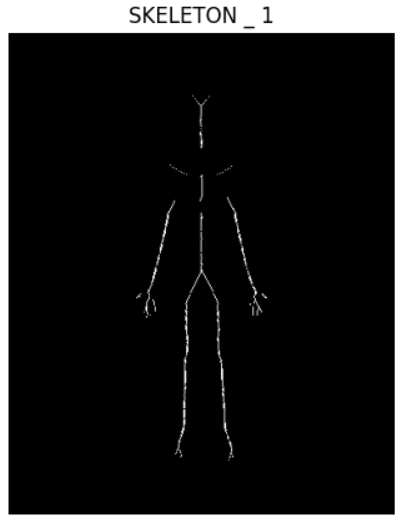
که کد آن به شرح زیر است:



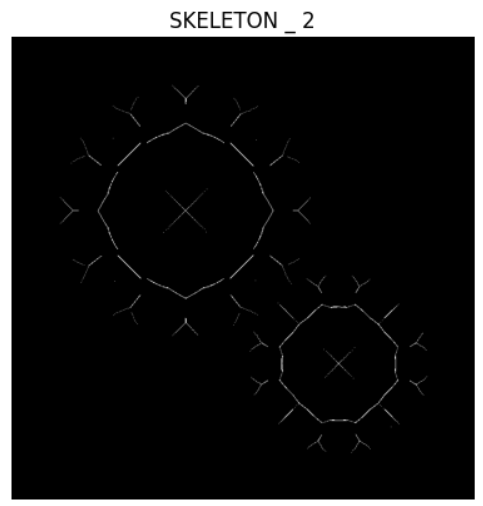
و همچنین برای در هر مرحله sk ما سیو می شود و در نهایت ما اسکلت داریم که کد آن نیز به شرح زیر است:

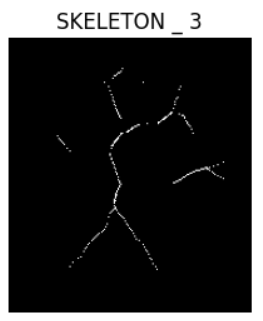


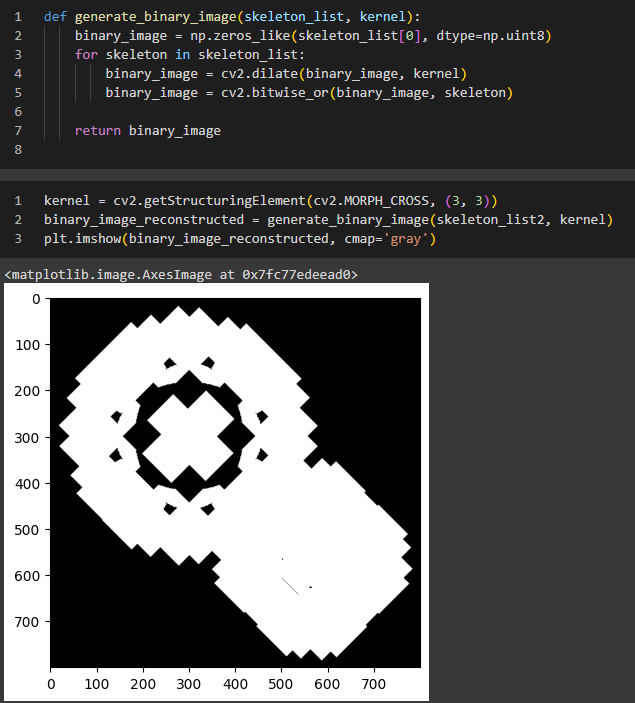
که در تصویر بالا مقدار هر سه عکس را می خوانیم و به برا ی آن threshold می گذاریم و سپس برای هر کدام از عکس ها خروجی زیر دریافت می شود و اسکلت ها به صورت زیر خواهند بود.



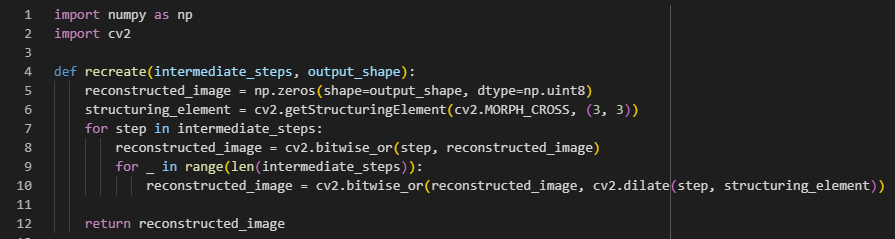


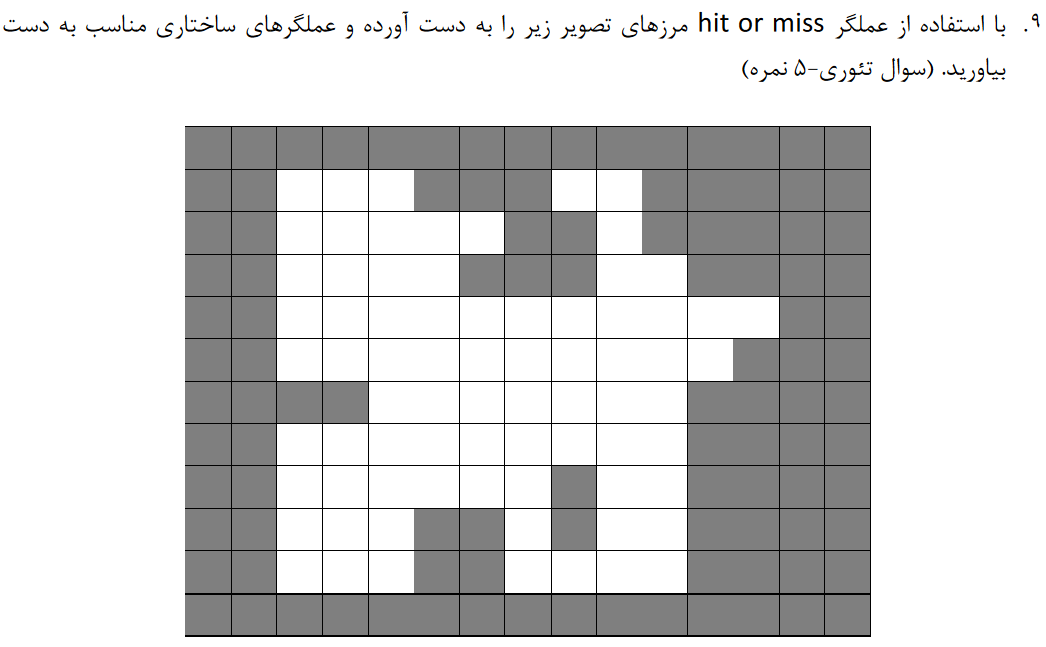






همانطور که در برخی از تصاویر بالا دیده می شود، برای بازسازی هر کدام از تصاویر نیز مانند فرمول گفته شده در اسلایدها باید اجتماع Sk ها را با گسترش kb به دست آوریم که کد آن نیز به صورت زیر می باشد:



9)

پاسخ:

برای این کار در ابتدا 4 فیلتر 2 \* 2 در نظر می گیریم که هر کدام مرز مخصوص به خود را تشخیص می دهند. از آنجا که حاشیه ی تصویر همگی صفر هستند لذا نیازی به padding های مختلف نیست

برای هر فیلتر نیز بدین صورت عمل می کنیم که پیکسل اصلی را همان نقطه ی مساوی با یک در نظر می گیریم و سپس باید نقطه ای که روی آن قرار می گیریم باید یک و نقطه ی منفی یک باید صفر و نقاط صفر نیز هر چیزی می توانند باشند. که با توجه به توضیحات گفته شده نقاط فیروزه ای برای هر کدام از فیلترها جواب می باشند.

| -1 | 1 |
| --- | --- |
| 0 | 0 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

| 0 | 1 |
| --- | --- |
| 0 | -1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

| 1 | -1 |
| --- | --- |
| 0 | 0 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

| 0 | -1 |
| --- | --- |
| 0 | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

حال از 4 شکل بالا اشکال فیروزه ای رنگ را اجتماع می گیریم و داریم:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | & | & |  |  |  | & |  |  |  |  |  |
|  |  |  |  | & |  |  |  |  |  | & |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | & |  | & |  |  | & |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | & |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

که نقاط به دست آمده همان جواب ما می باشند و نقاط مرزی محسوب می شوند. به غیر از راه نوشته شده نیز می توانستیم از فیلتر 3 در 3 استفاده کنیم و یا نقاط گوشه ای را نیز حساب کنیم که برای این کار حتما باید از فیلتر 3 در 3 استفاده کنیم و به جواب های ما نقاطی که با علامت & نمایش داده شده اند نیز به جواب های ما افزوده می شوند.