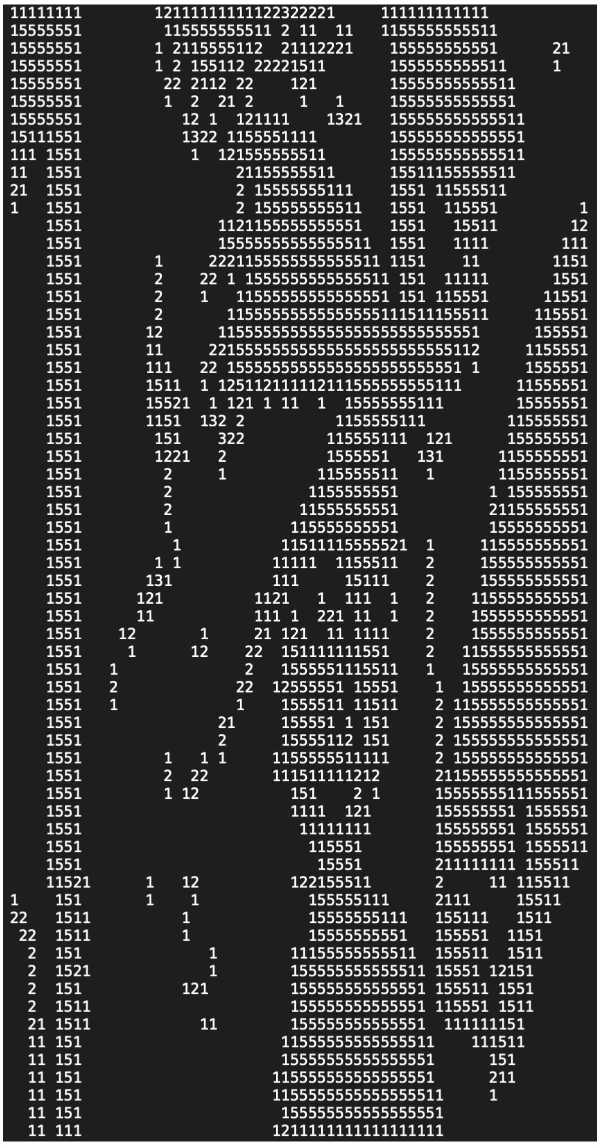
# Howework 6 Report

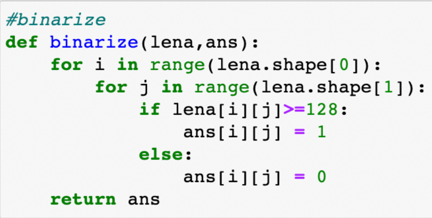
Student:鞠之浩  ID:P10942A08

Output：



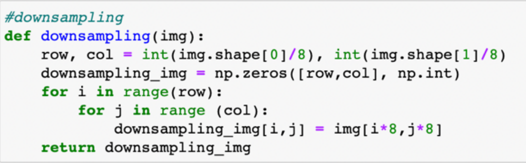
```
11111111        12111111111122322221     111111111111
15555551         115555555511 2 11  11   1155555555511
15555551        1 2115555112  21112221    155555555551      21
15555551        1 2 155112 22221511       1555555555511     1
15555551         22 2112 22    121        15555555555511
15555551         1  2  21 2     1   1     15555555555551
15555551           12 1  121111    1321   155555555555511
15111551           1322 1155551111        155555555555551
111 1551            1  121555555511       155555555555511
11  1551                 21155555511      15511555555511
21  1551                 2 15555555111    1551 11555511
1   1551                 2 155555555511   1551  115551         1
    1551               1121155555555551   1551   15511        12
    1551               15555555555555511  1551   1111        111
    1551        1     2221155555555555511 1151    11        1151
    1551        2    22 1 1555555555555511 151  11111       1551
    1551        2    1   11555555555555551 151 115551      11551
    1551        2       11555555555555555511511155511     115551
    1551       12      1155555555555555555555555551      155551
    1551       11     221555555555555555555555555112     1155551
    1551       111   22 15555555555555555555555551 1     1555551
    1551       1511  1 125112111121115555555555111      11555551
    1551       15521  1 121 1 11  1  15555555111        15555551
    1551       1151  132 2          1155555111         115555551
    1551        151    322         115555111  121      155555551
    1551        1221   2           1555551   131      1155555551
    1551         2     1          115555511   1       1155555551
    1551         2               1155555551          1 155555551
    1551         2              11555555551          21155555551
    1551         1             115555555551          15555555551
    1551          1           11511115555521  1     115555555551
    1551        1 1          11111  1155511   2     155555555551
    1551       131           111     15111    2     155555555551
    1551      121          1121   1  111  1   2    1155555555551
    1551      11           111 1  221 11  1   2    1555555555551
    1551    12       1     21 121  11 1111    2    1555555555551
    1551     1      12    22  151111111551    2   11555555555551
    1551   1              2   1555551115511   1   15555555555551
    1551   2             22  12555551 15551    1  15555555555551
    1551   1             1    1555511 11511    2 115555555555551
    1551               21     155551 1 151     2 155555555555551
    1551               2      1555112  151     2 155555555555551
    1551         1   1 1     1155555511111     2 155555555555551
    1551         2  22       111511111212      21155555555555551
    1551         1 12          151    2 1      15555555511555551
    1551                       1111  121       155555551 1555551
    1551                        11111111       155555551 1555551
    1551                         115551        155555551 1555511
    1551                          15551        211111111 155511
    11521      1   12          122155511       2     11 115511
1    151       1    1            155555111     2111     15511
22   1511          1             15555555111   155111   1511
 22  1511          1             15555555551   155551  1151
  2  151              1        11155555555511  155511  1511
  2  1521             1        155555555555511 15551 12151
  2  151           121         155555555555551 155511 1551
  2  1511                      155555555555551 115551 1511
  21 1511            11        155555555555551  111111151
  11 151                      11555555555555511    111511
  11 151                      15555555555555551      151
  11 151                     115555555555555551      211
  11 151                     1155555555555555511     1
  11 151                      155555555555555551
  11 111                     1211111111111111111
```

Code describe：step1:先將 lena binarize



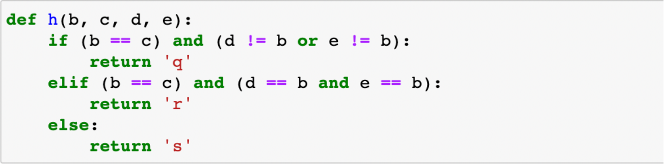
```
#binarize
def binarize(lena,ans):
    for i in range(lena.shape[0]):
        for j in range(lena.shape[1]):
            if lena[i][j]>=128:
                ans[i][j] = 1
            else:
                ans[i][j] = 0
    return ans
```

Step2:將 binary image downsampling

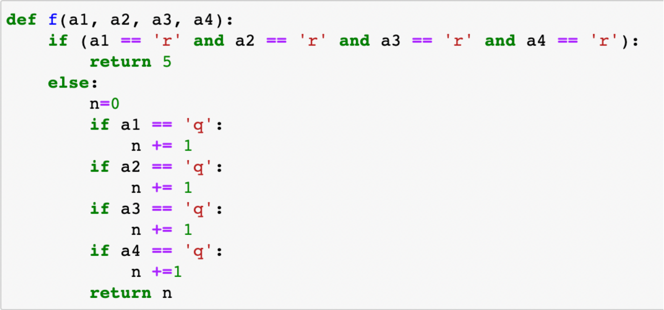


```
#downsampling
def downsampling(img):
    row, col = int(img.shape[0]/8), int(img.shape[1]/8)
    downsampling_img = np.zeros([row,col], np.int)
    for i in range(row):
        for j in range (col):
            downsampling_img[i,j] = img[i*8,j*8]
    return downsampling_img
```

Step3:計算 h 跟 f function

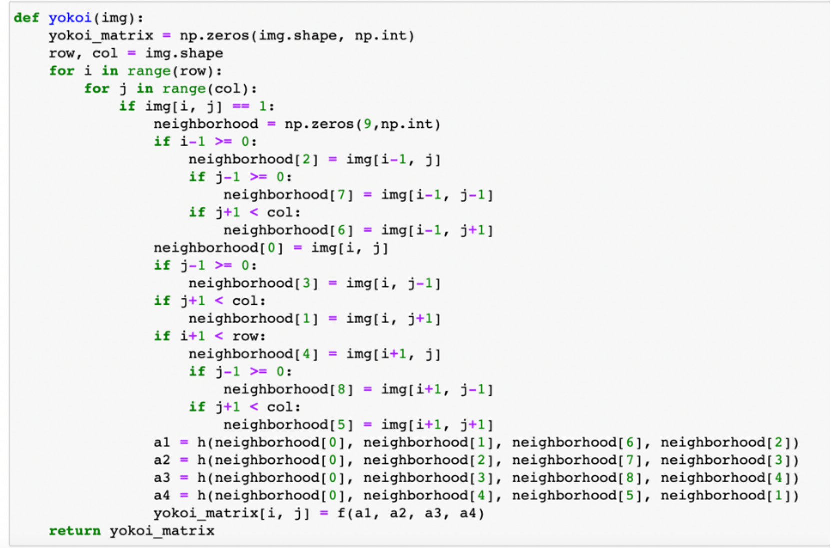


```python
def h(b, c, d, e):
    if (b == c) and (d != b or e != b):
        return 'q'
    elif (b == c) and (d == b and e == b):
        return 'r'
    else:
        return 's'
```



```python
def f(a1, a2, a3, a4):
    if (a1 == 'r' and a2 == 'r' and a3 == 'r' and a4 == 'r'):
        return 5
    else:
        n=0
        if a1 == 'q':
            n += 1
        if a2 == 'q':
            n += 1
        if a3 == 'q':
            n += 1
        if a4 == 'q':
            n +=1
        return n
```

Step4:計算個 pixel 的 yokoi connectivity number



```python
def yokoi(img):
    yokoi_matrix = np.zeros(img.shape, np.int)
    row, col = img.shape
    for i in range(row):
        for j in range(col):
            if img[i, j] == 1:
                neighborhood = np.zeros(9,np.int)
                if i-1 >= 0:
                    neighborhood[2] = img[i-1, j]
                    if j-1 >= 0:
                        neighborhood[7] = img[i-1, j-1]
                    if j+1 < col:
                        neighborhood[6] = img[i-1, j+1]
                neighborhood[0] = img[i, j]
                if j-1 >= 0:
                    neighborhood[3] = img[i, j-1]
                if j+1 < col:
                    neighborhood[1] = img[i, j+1]
                if i+1 < row:
                    neighborhood[4] = img[i+1, j]
                    if j-1 >= 0:
                        neighborhood[8] = img[i+1, j-1]
                    if j+1 < col:
                        neighborhood[5] = img[i+1, j+1]
                a1 = h(neighborhood[0], neighborhood[1], neighborhood[6], neighborhood[2])
                a2 = h(neighborhood[0], neighborhood[2], neighborhood[7], neighborhood[3])
                a3 = h(neighborhood[0], neighborhood[3], neighborhood[8], neighborhood[4])
                a4 = h(neighborhood[0], neighborhood[4], neighborhood[5], neighborhood[1])
                yokoi_matrix[i, j] = f(a1, a2, a3, a4)
    return yokoi_matrix
```