

Arifullah Jan - 186943 - BSCS 6A

Saqib Shakeel - 176783 - BSCS 6A

LAB 04 - DIP



```
'''  
python task1.py path/to/input  
'''  
  
import PIL  
from PIL import Image, ImageEnhance  
import itertools  
import numpy  
import sys  
from PIL import ImageDraw  
import matplotlib.pyplot as plt
```

```

import itertools
image = Image.open(sys.argv[1])
image = image.convert("L") # convert to singal channeled image
image = image.point(lambda x: 0 if x < 150 else 255, '1') # binarized
image

width, height = image.size

lables = numpy.zeros((width, height), dtype=numpy.int)

currentLabel = 0

shouldMerge = dict()

white = 255

# print(lables)

# initial

for x,y in itertools.product(range(height),range(width)):

    # print('=====', x, y)
    if image.getpixel((x, y)) == 0:
        continue
    this = image.getpixel((x, y))
    top = None
    if not y == 0:
        top = image.getpixel((x, y-1))
        # print('top ', x, y ,top)

    left = None
    if not x == 0:
        left = image.getpixel((x-1, y))
        # print('left ', x, y , left)

    if (top == None or top == 0) and (left == None or left == 0):
        currentLabel += 1
        lables[x, y] = currentLabel
    elif (top == white) and (left == None or left == 0):
        # print(x,y)
        lables[x, y] = lables[x, y-1]

    elif (left == white) and (top == None or top == 0):
        # print(x,y)
        lables[x, y] = lables[x-1, y]

```

```

elif (left == white or top == white):
    print(x,y)
    if(lables[x-1, y] < lables[x, y-1]):
        lables[x, y] = lables[x-1, y]
        shouldMerge[lables[x, y-1]] = lables[x-1, y] # this label
    shoul be merged
    else:
        lables[x, y] = lables[x, y-1]
        shouldMerge[lables[x-1, y]] = lables[x, y-1]

    # print(lables)

print(shouldMerge)

# second pass

for x,y in itertools.product(range(height),range(width)):

    lables[x,y] = shouldMerge.get(lables[x,y], lables[x,y])
    # shouldMerge[lables[x,y]] if shouldMerge.has_key(lables[x,y]) == None
    else lables[x,y]

print('===== RESULT =====')

print(lables)


def plot_matrix(rm, title='Color Map', cmap=plt.cm.Blues):
    plt.imshow(rm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.tight_layout()
    plt.show()

plot_matrix(lables,cmap = plt.cm.Blues)

```