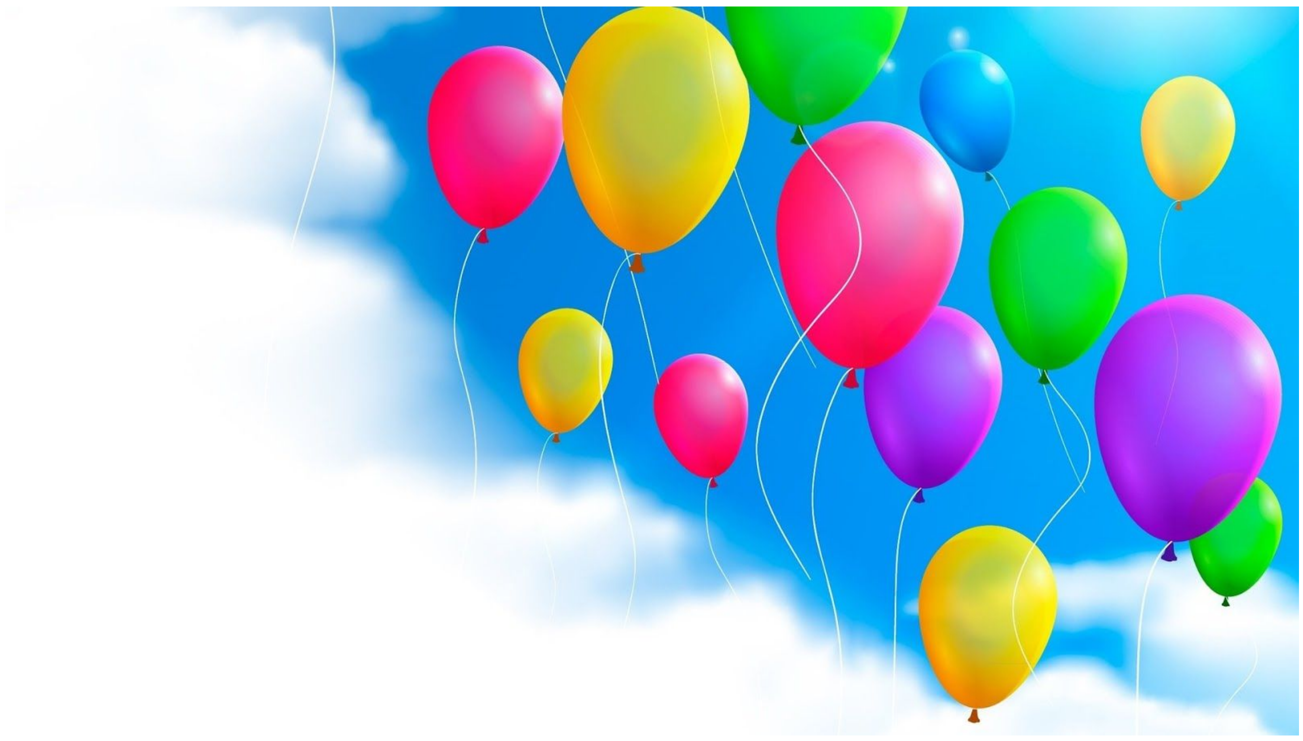
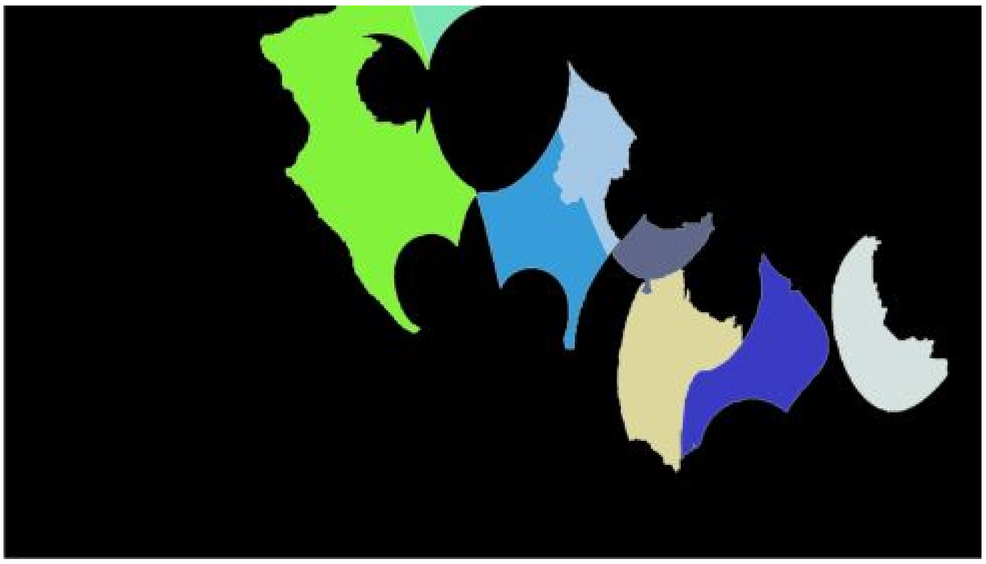
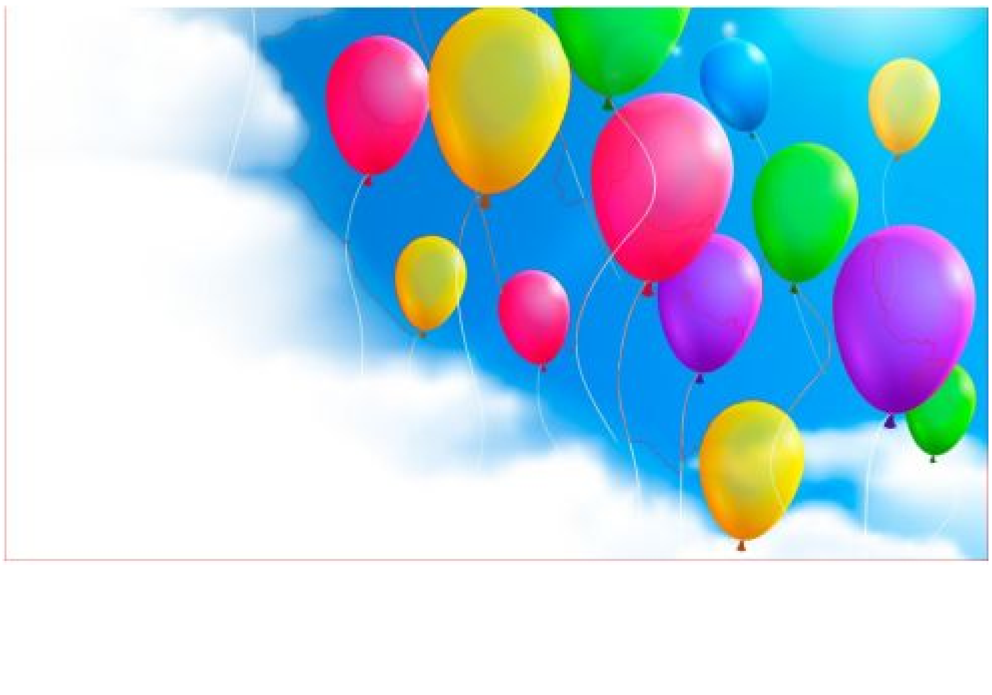
КУРЕЦ Любовь ИСУ
ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ
Сегментация изображений. Метод роста областей. Разделение и слияние областей.

Используя python библиотеку **OpenCV** и **jupyterthemes** можно выделить границы объектов на изображении.

ВХОДНОЕ ИЗОБРАЖЕНИЕ



РЕЗУЛЬТАТ

Листинг программы

```
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt
import random
from jupyterthemes import jtplot

jtplot.style()

def show_images(im_dict, cols, shape = (1,1), length = None, size = 8):
  if not length:
     length = len(im_dict)
  else:
     length = max(length, len(im_dict))
  w_size = shape[1] / sum(shape) * size
  h_size = shape[0] / sum(shape) * size
  rows = np.ceil(length / cols)
  fig = plt.figure(figsize=(w_size*cols, h_size*rows))
  i = 0
  for key, value in im_dict.items():
     i += 1
     if len(value.shape) > 2:
        conv = cv.COLOR_BGR2RGB
     else:
        conv = cv.COLOR_GRAY2RGB
     im = cv.cvtColor(value, conv)
     sub = fig.add_subplot(rows, cols, i)
     sub.set_title(key)
     plt.axis('off')
     plt.imshow(im)
  return fig

fn = 'coins.jpg'
src = cv.imread(fn)
```

```
dst = src.copy()
src_gray = cv.imread(fn, 0)
ret,binary = cv.threshold(src_gray,127,255,cv.THRESH_BINARY)
binary = 255 - binary
kernel = np.ones((3,3))
closing = cv.morphologyEx(binary, cv.MORPH_CLOSE, kernel)
opening = cv.morphologyEx(closing, cv.MORPH_CLOSE, kernel)

# sure background area
sure_bg = cv.dilate(binary,kernel,iterations=3)
# Finding sure foreground area
dist_transform = cv.distanceTransform(opening,cv.DIST_L2,5)
ret, sure_fg =
cv.threshold(dist_transform,0.7*dist_transform.max(),255,0)
dist_transform = cv.normalize(dist_transform, None, 0, 255,
cv.NORM_MINMAX, cv.CV_8U)
# Finding unknown region
sure_fg = np.uint8(sure_fg)
unknown = cv.subtract(sure_bg,sure_fg)

# Marker labelling
ret, markers = cv.connectedComponents(sure_fg)
# Add one to all labels so that sure background is not 0, but 1
markers = markers+1
# Now, mark the region of unknown with zero
markers[unknown==255] = 0

markers = cv.watershed(src, markers)
dst[markers == -1] = [0,0,255]

vis = np.zeros(src.shape, dtype=np.uint8)
m = markers.max()
vis[markers < 0] = [255,255,255]
for i in range(2,m+1):
  b = 50+np.random.randint(200)
  g = 50+np.random.randint(200)
```

```
    r = 50+np.random.randint(200)
    vis[markers == i] = [b, g, r]

im_dict = {
    'Source':src,
    'Binary':binary,
    'Distance Transform':dist_transform,
    'Sure Foreground':sure_fg,
    'Sure Background':sure_bg,
    'Visualize':vis,
    'Result':dst
}

cols = 3
size = 16
fig = show_images(im_dict, cols, shape = src.shape, size = size)
```