In [ ]:

```
'''
• read image “tulips.jpg”
• change its background to blue, black, green, red and record all of
them as new images
• using matplotlib display original image and four images that have
been generated
'''
```

In [1]:

```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
```

In [ ]:

```
def write_images():
    cv.imwrite("images/blue_background_image.png", blue_background_image)
    cv.imwrite("images/black_background_image.png", black_background_image)
    cv.imwrite("images/green_background_image.png", green_background_image)
    cv.imwrite("images/red_background_image.png", red_background_image)

def plot_images():
    fig, axs = plt.subplots(1, len(images))

    for i in range(0, len(images)):
        axs[i].imshow(cv.cvtColor(images[i], cv.COLOR_BGR2RGB))

def set_blue(i, j, image):
    image[i, j, 0] = 255
    image[i, j, 1] = 0
    image[i, j, 2] = 0

def set_black(i, j, image):
    image[i, j, 0] = 0
    image[i, j, 1] = 0
    image[i, j, 2] = 0

def set_green(i, j, image):
    image[i, j, 0] = 0
    image[i, j, 1] = 255
    image[i, j, 2] = 0

def set_red(i, j, image):
    image[i, j, 0] = 0
    image[i, j, 1] = 0
    image[i, j, 2] = 255

original_image = cv.imread("images/tulips.jpg", 1)
blue_background_image = cv.imread("images/tulips.jpg", 1)
black_background_image = cv.imread("images/tulips.jpg", 1)
green_background_image = cv.imread("images/tulips.jpg", 1)
red_background_image = cv.imread("images/tulips.jpg", 1)

images = [original_image,
          blue_background_image,
          black_background_image,
          green_background_image,
          red_background_image]

height = original_image.shape[0]
width = original_image.shape[1]

for i in range(0, height):
    for j in range(0, width):

        if original_image[i, j, 0] in range(252, 256):
            if original_image[i, j, 1] in range(252, 256):
                if original_image[i, j, 2] in range(252, 256):

                    set_blue(i, j, blue_background_image)
                    set_black(i, j, black_background_image)
                    set_green(i, j, green_background_image)
                    set_red(i, j, red_background_image)


write_images()
plot_images()
```

In [8]:

```
# 2
image = np.zeros((400, 400, 3), dtype = np.uint8)

red_radius = 150
blue_radius = 140
thickness = -1
center_coordinates = (200, 200)
red_color = (0, 0, 255)
blue_color = (255, 0, 0)

image = cv.circle(image, center_coordinates, red_radius, red_color, thickness)
image = cv.circle(image, center_coordinates, blue_radius, blue_color, thickness)

cv.imshow("Circle with opencv", image)
cv.waitKey(0)
cv.destroyAllWindows()
cv.waitKey(1)
```

Out[8]:

```
-1
```

In [7]:

```
# 3
import math

def calculate_distance(a, b):
    x1, y1 = a
    x2, y2 = b
    return math.sqrt(math.pow((x1 - x2), 2) + math.pow((y1 - y2), 2))


image = np.zeros((400, 400, 3), dtype = np.uint8)

red_radius = 150
blue_radius = 140

center_coordinates = (200, 200)

for i in range(0, 400):
    for j in range(0, 400):

        distance = calculate_distance((i, j), center_coordinates)
        if distance <= red_radius:
            image[i, j, 0] = 0
            image[i, j, 1] = 0
            image[i, j, 2] = 255

        distance = calculate_distance((i, j), center_coordinates)
        if distance <= blue_radius:
            image[i, j, 0] = 255
            image[i, j, 1] = 0
            image[i, j, 2] = 0

cv.imshow("Circle without opencv", image)
cv.waitKey(0)
cv.destroyAllWindows()
cv.waitKey(1)
```

Out[7]:

```
-1
```

In [ ]: