```
# a1.py
# Nick Ladd
import numpy as np
import math
from matplotlib import pyplot as plt

# === Main function ===
def main():

  plt.ion();
  smiley = step1();
  face = step2();
  face_div_2 = step3(face);
  step4(face_div_2);
  step5(face);
  step6(face);
  step7(face);
  step8(face);
  extra_credit(face);

# === Step 1 ===
def step1():
  smiley=np.array([\
                   [1, 1, 1,  1,  0,  0, 0, 0, 0, 0,  0,  0,  1, 1, 1],\
                   [1, 1, 1,  0,  1,  1, 1, 1, 1, 1,  1,  1,  0, 1, 1],\
                   [1, 1, 0,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 0, 1],\
                   [1, 0, 1,  1,  1,  0, 0, 1, 1, 0,  0,  1,  1, 1, 0],\
                   [1, 0, 1,  1,  1,  0, 0, 1, 1, 0,  0,  1,  1, 1, 0],\
                   [1, 0, 1,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 1, 0],\
                   [1, 0, 1, .5, .5, .5, 1, 1, 1, 1, .5, .5, .5, 1, 0],\
                   [1, 0, 1, .5,  0,  1, 1, 1, 1, 1,  1,  0, .5, 1, 0],\
                   [1, 0, 1, .5,  1,  0, 1, 1, 1, 1,  0,  1, .5, 1, 0],\
                   [1, 1, 0,  1,  1,  1, 0, 0, 0, 0,  1,  1,  1, 0, 1],\
                   [1, 1, 1,  0,  0,  1, 1, 1, 1, 1,  1,  0,  0, 1, 1],\
                   [1, 1, 1,  1,  1,  0, 0, 0, 0, 0,  0,  1,  1, 1, 1],\
                   [1, 1, 1,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 1, 1]])
  smiley=smiley*255;

  plt.subplot(121),plt.imshow(smiley, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest');
  plt.subplot(122),plt.imshow(smiley, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'bicubic');
  plt.pause(1)
  plt.draw();

  return smiley;

# === Step 2 ===
def step2():
  face=np.array([\
                   [1, 1, 1,  1,  0,  0, 0, 0, 0, 0,  0,  0,  1, 1, 1],\
                   [1, 1, 1,  0,  1,  1, 1, 1, 1, 1,  1,  1,  0, 1, 1],\
                   [1, 1, 0,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 0, 1],\
                   [1, 0, 1,  1,  1,  0, 0, 1, 1, 0,  0,  1,  1, 1, 0],\
                   [1, 0, 1,  1,  1,  0, 0, 1, 1, 0,  0,  1,  1, 1, 0],\
                   [1, 0, 1,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 1, 0],\
```

```
                    [1, 0, 1, .5, .5, .5, 1, 1, 1, 1, .5, .5, .5, 1, 0],\
                    [1, 0, 1, .5,  0,  1, 1, 1, 1, 1,  1,  0, .5, 1, 0],\
                    [1, 0, 1, .5,  1,  0, 1, 0, 0, 1,  0,  1, .5, 1, 0],\
                    [1, 1, 0,  1,  1,  1, 0, 0, 0, 0,  1,  1,  1, 0, 1],\
                    [1, 1, 1,  0,  0,  1, 1, 1, 1, 1,  1,  0,  0, 1, 1],\
                    [1, 1, 1,  1,  1,  0, 0, 0, 0, 0,  0,  1,  1, 1, 1],\
                    [1, 1, 1,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 1, 1]])
  face=face*255;

  return face;

# === Step 3 ===
def step3(face):
  face_div_2=face/2
  face_div_4=face/4

  plt.subplot(131),plt.imshow(face, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(132),plt.imshow(face_div_2, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(133),plt.imshow(face_div_4, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.pause(1)
  plt.draw()

  return face_div_2

# === Step 4 ===
def step4(face_div_2):
  face_div_2_plus_50 = face_div_2+50
  face_div_2_plus_100 = face_div_2+100

  plt.subplot(131),plt.imshow(face_div_2, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(132),plt.imshow(face_div_2_plus_50, vmin=0, vmax=255,cmap = 'gray
      ', interpolation = 'nearest')
  plt.subplot(133),plt.imshow(face_div_2_plus_100, vmin=0, vmax=255,cmap =
      'gray', interpolation = 'nearest')
  plt.pause(1)
  plt.draw()

# === Step 5 ===
def step5(face):
  face_flip_x = np.flipud(face)
  face_flip_y = np.fliplr(face)

  plt.subplot(131),plt.imshow(face, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(132),plt.imshow(face_flip_x, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(133),plt.imshow(face_flip_y, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.pause(1)
  plt.draw()

# === Step 6 ===
```

```
def step6(face):
  face_rotate_90 = np.rot90(face)
  face_rotate_180 = np.rot90(face, 2)
  face_rotate_270 = np.rot90(face, 3)

  plt.subplot(221),plt.imshow(face, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(222),plt.imshow(face_rotate_90, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(223),plt.imshow(face_rotate_180, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(224),plt.imshow(face_rotate_270, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.pause(1)
  plt.draw()

# === Step 7 ===
def step7(face):
  face_cropped = face[3:10, 3:13]

  plt.subplot(111),plt.imshow(face_cropped, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.pause(1)
  plt.draw()

# === Step 8 ===
def step8(face):

  face_rotated_45 = rotate_image(45, face)
  face_rotated_60 = rotate_image(60, face)
  face_rotated_90 = rotate_image(90, face)
  face_rotated_135 = rotate_image(135, face)
  face_rotated_360 = rotate_image(360, face)

  plt.subplot(321),plt.imshow(face, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(322),plt.imshow(face_rotated_45, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(323),plt.imshow(face_rotated_60, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(324),plt.imshow(face_rotated_90, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(325),plt.imshow(face_rotated_135, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.subplot(326),plt.imshow(face_rotated_360, vmin=0, vmax=255,cmap = 'gray',
      interpolation = 'nearest')
  plt.pause(1)
  plt.draw()

# === Extra Credit ===
def extra_credit(face):

  padding = np.array((1, 1, 1,  1,  1,  1, 1, 1, 1, 1,  1,  1,  1, 1, 1))
  padding = padding*255
  face = np.vstack((padding, face))
```

```
    face_rotated_45 = rotate_image(45, face)
    face_rotated_60 = rotate_image(60, face)
    face_rotated_90 = rotate_image(90, face)
    face_rotated_135 = rotate_image(135, face)
    face_rotated_360 = rotate_image(360, face)

    plt.subplot(321),plt.imshow(face, vmin=0, vmax=255,cmap = 'gray',
        interpolation = 'nearest')
    plt.subplot(322),plt.imshow(face_rotated_45, vmin=0, vmax=255,cmap = 'gray',
        interpolation = 'nearest')
    plt.subplot(323),plt.imshow(face_rotated_60, vmin=0, vmax=255,cmap = 'gray',
        interpolation = 'nearest')
    plt.subplot(324),plt.imshow(face_rotated_90, vmin=0, vmax=255,cmap = 'gray',
        interpolation = 'nearest')
    plt.subplot(325),plt.imshow(face_rotated_135, vmin=0, vmax=255,cmap = 'gray',
        interpolation = 'nearest')
    plt.subplot(326),plt.imshow(face_rotated_360, vmin=0, vmax=255,cmap = 'gray',
        interpolation = 'nearest')
    plt.pause(1)
    plt.draw()

    print ("Extra Credit Answer (#2): By adding an additional row of pixels to
        the face array,"
         " the top line of the smiley face is retained since there is some
             padding to preserve it"
         " during the rotation.")

# === Rotation Calculation ===
def new_location(rotation_matrix, x, y):
    new_coordinate = np.dot([x, y], rotation_matrix)
    new_coordinate = [int(new_coordinate[0]), int(new_coordinate[1])]
    return new_coordinate

# === Populate New Image ===
def populate_image(image, image_rotated, new_coordinates, x, y):
    x_new = new_coordinates[x][y][0]
    y_new = new_coordinates[x][y][1]
    image_rotated[x_new, y_new] = image[x, y]

# === Move New Array ===
def move_array(new_coordinates, x, y, move_by_x, move_by_y):
    new_coordinates[x][y][1] = new_coordinates[x][y][1] + move_by_y + 1
    new_coordinates[x][y][0] = new_coordinates[x][y][0] + move_by_x + 1

# === Rotation Function ===
def rotate_image(angle, image):

    # We're rotating counterclockwise
    angle = -1*angle

    # Convert to radians
    angle = np.radians(angle)

    # Our rotation matrix
    rotation_matrix = [[ np.cos(angle), (-1 * np.sin(angle))], [ np.sin(angle),
        np.cos(angle) ]]
```

```
    # Move image origin to center of array
    x_size = int(image.shape[0]/2)
    y_size = int(image.shape[1]/2)
    array_range_x = range(-1*x_size, x_size)
    array_range_y = range(-1*y_size, y_size)

    # Rotate the image
    new_coordinates = map(lambda x: map(lambda y: new_location(rotation_matrix, x
        , y), array_range_y), array_range_x)

    # Get min/max height
    min_height = np.amin(new_coordinates, axis = (1,0))[1]
    max_height = np.amax(new_coordinates, axis = (1,0))[1]

    # Get min/max width
    min_width = np.amin(new_coordinates, axis = (0,1))[0]
    max_width = np.amin(new_coordinates, axis = (0,1))[0]

    # New array boundaries with padding for safety
    new_array_height = (abs(min_height) + abs(max_height))+4
    new_array_width = (abs(min_width) + abs(max_width))+4

    # Move array
    map(lambda x: map(lambda y: move_array(new_coordinates, x, y, abs(max_width),
        abs(max_height)), array_range_y), array_range_x)

    # Empty new array
    image_rotated = np.zeros((new_array_width, new_array_height))
    image_rotated.fill(255)

    # Fill in new array
    map(lambda x: map(lambda y: populate_image(image, image_rotated,
        new_coordinates, x, y), array_range_y), array_range_x)

    return image_rotated

# === Execute the main function ===
main();
```