d11922022 資工所 楊寓鈞

Environment

- python3.9
- Pillow 10.0.0, numpy 1.25.2

basic setups and utility functions

```
from PIL import Image
import numpy as np
import copy

img = Image.open('./lena.bmp') # load lena.bmp
img_array = np.array(img) # pixel content saved in np.array
width, height = img_array.shape # get `width` and `height`
img_list = img_array.tolist() # transform pixel content into list

def save_image(img, path='./lena.bmp'):
        img_ = Image.fromarray(np.array(img, dtype='uint8'), mode='L')
        img_.save(path)
        return img_
```

Part 1

a. upside-down

```
def upside_down(img, height=height, width=width):
        for y in range(height//2):
                for x in range(width):
                        elm = img[y][x]
                        img[y][x] = img[(height-1)-y][x]
                        img[(height-1)-y][x] = elm
        return img

result = copy.deepcopy(img_list) # remove mutability of nested list
save_image(upside_down(result), './lena_upside_down.bmp')
```

1. the function loop through half the rows

2. for each row `y`, swap it with row `(height-1) - y`

b. right-side-left

```
def rightside_left(img, height=height, width=width):
        for y in range(height):
                for x in range(width//2):
                        elm = img[y][x]
                        img[y][x] = img[y][(width-1)-x]
                        img[y][(width-1)-x] = elm
        return img

result = copy.deepcopy(img_list) # remove mutability of nested list
save_image(rightside_left(result), './lena_rightside_left.bmp')
```

1. the function loop through the rows
2. in each row `y`, loop through half the columns

3. for each column `x` (of the row `y`), swap the column `x` with column `(width-1)-x`

c. diagonal mirrored

```
def digonal_flip(img, height=height, width=width):
        img = upside_down(img, height, width)
        img = rightside_left(img, height, width)
        return img

result = copy.deepcopy(img_list) # remove mutability of nested list
save_image(digonal_flip(result), './lena_diagonal_mirrored.bmp')
```

1. the function first do the upside-down

2. next do the rightside-left

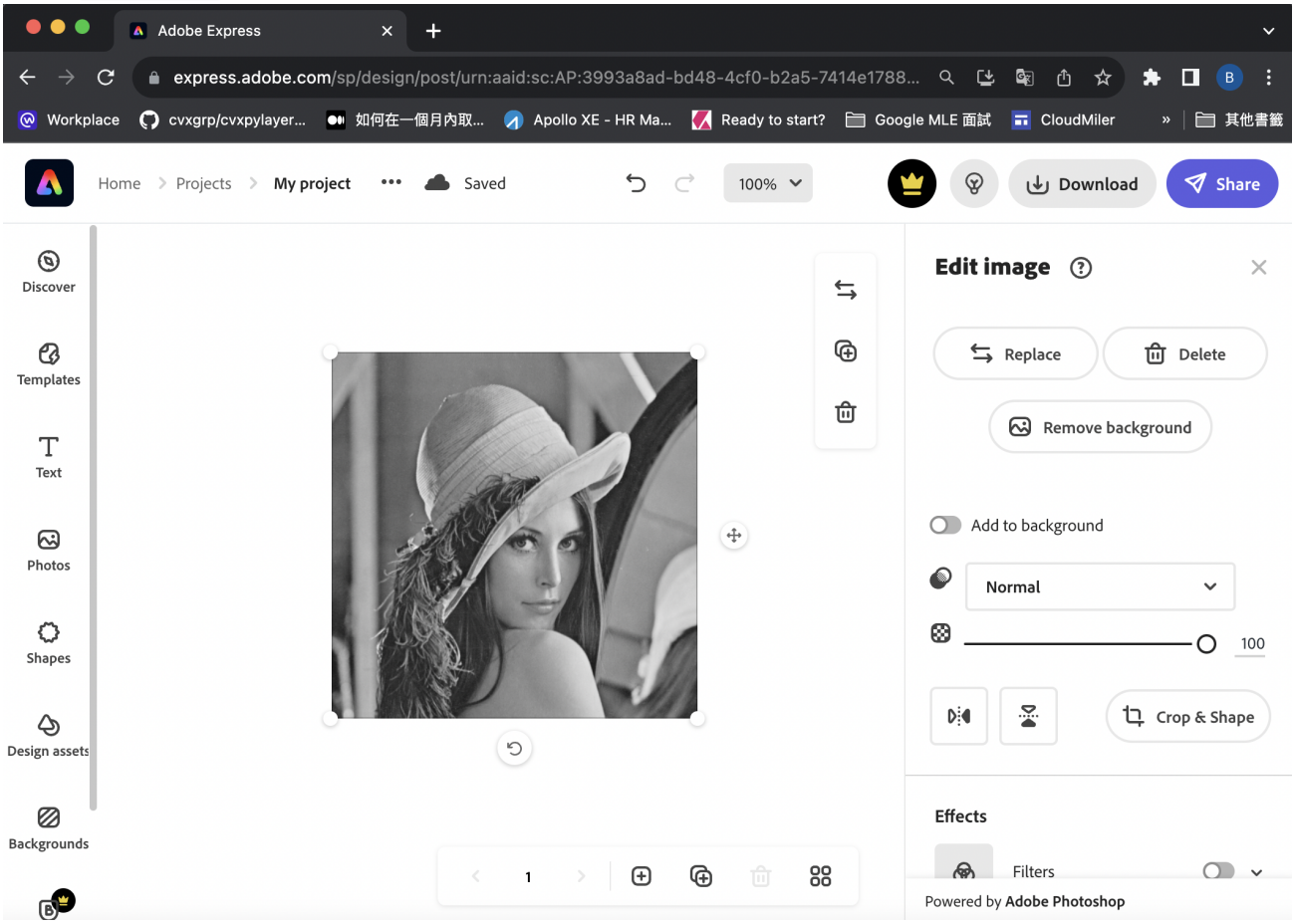
Part 2

d. rotate 45 degree clockwise

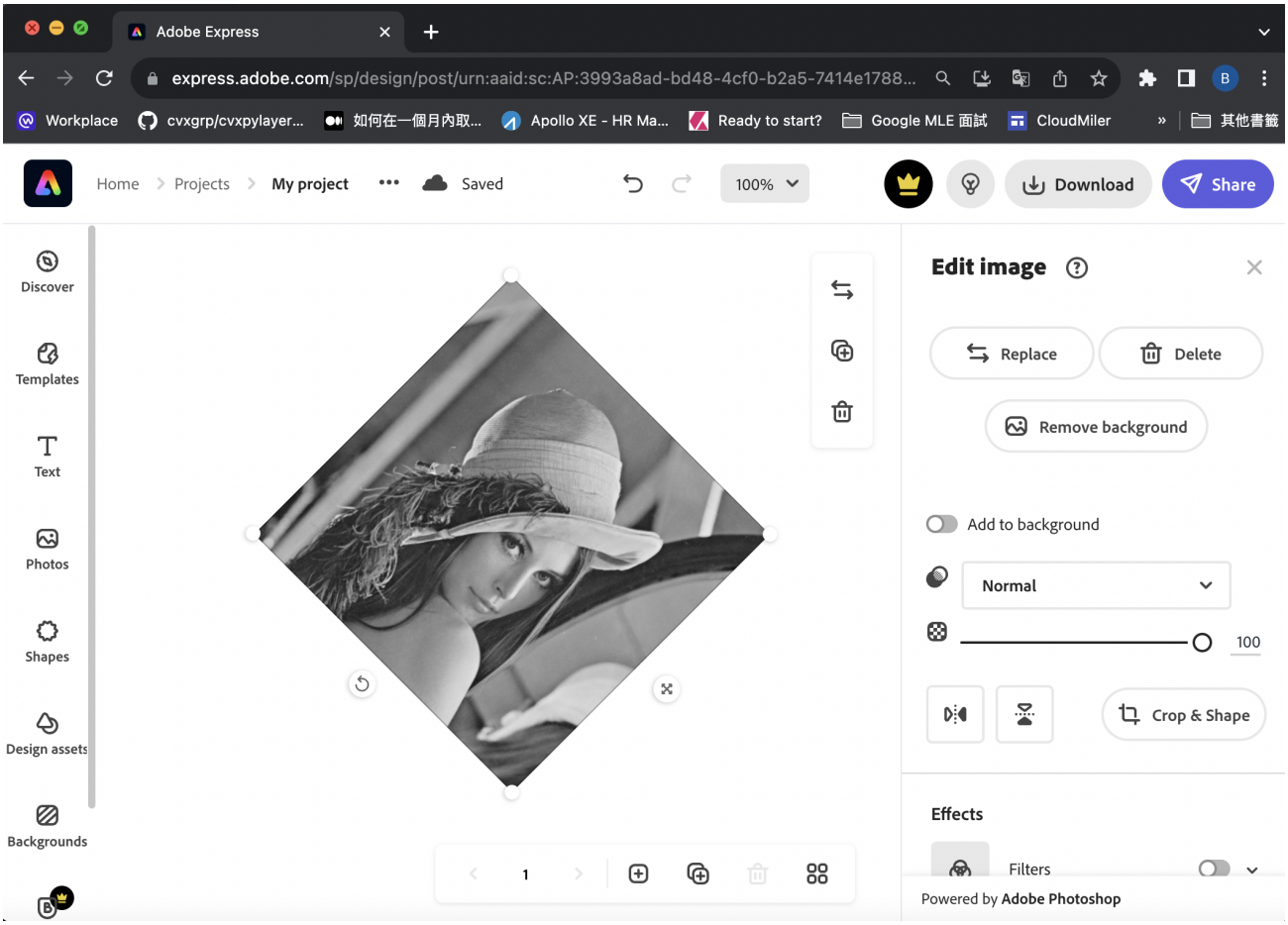
use adobe photoshop online
https://www.adobe.com/tw/express/feature/image/editor

1. open a new project on adobe photoshop online
2. change `lena.bmp` to `lena.png` (it only accepts png, jpeg, and jpg)

3. drag `lena.png` to adobe photoshop online



4. click the `rotate` icon, and rotate 45° clockwise



e. shrink in half

```
def shrink(img, height=height, width=width, scale=2):
        for y in range(0, height, 2):
                for x in range(0, width, 2):
                        elm = img[y][x]
                        img[y//scale][x//scale] = elm
                        img = [ [img[y][x] for x in range(0, width//scale)]
for y in range(0, height//scale)]
        return img

result = copy.deepcopy(img_list) # remove mutability of nested list
save_image(shrink(result), './lena_shrink.bmp')
```

1. the function loop through every 2 row
2. in each even row `y`, loop through every 2 column of the row
3. for each column `x` (of the row `y`), save the content to row `y//2`, column `x//2`. which will make the left upper corner a 256 x 256 shrunk image

4. finally return just the top left 256 x 256 shrunk image

/home/jupyter/ntu/csie-cv/hw6/main.ipynb

f. binarize at 128

```
def binarize(img, height=height, width=width):
        for y in range(height):
                for x in range(width):
                        img[y][x] = 255 if img[y][x] >= 128 else 0
        return img

result = copy.deepcopy(img_list) # remove mutability of nested list
save_image(binarize(result), './lena_binarize.bmp')
```

1. the function loop through every row
2. in each even row `y`, loop through every column of the row
3. for each column `x` (of the row `y`),
   + set 255 if pixel value >= 128

+ set 0 if pixel value < 128