ps1-3

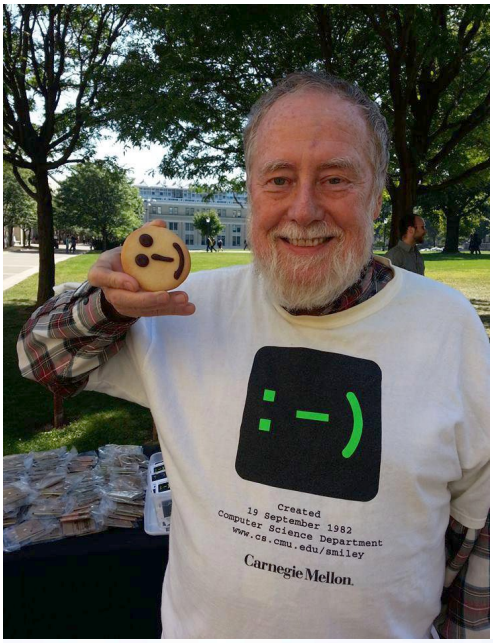
Shaobo Wang

Terminal printout:

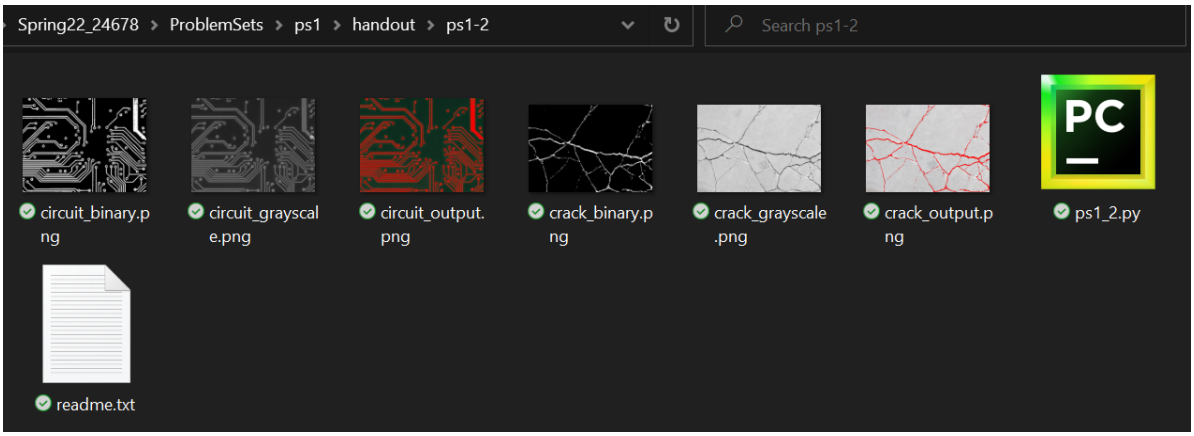
```
PS C:\Users\Brad\iCloudDrive\CMU_Spring2022_Files\Spring22_24678\ProblemSets\ps1\ps1-3> python ps1_3.py smiley.jpg
Press 'S' to save and exit
Adjusted gamma value is: 0.66
PS C:\Users\Brad\iCloudDrive\CMU_Spring2022_Files\Spring22_24678\ProblemSets\ps1\ps1-3> python ps1_3.py carnival.jpg
Press 'S' to save and exit
Adjusted gamma value is: 2.69
```

```
python ps1_3.py smiley.jpg
python ps1_3.py carnival.jpg
```

Image output:



File list in the ps1-2 folder:



Python script:

```python
import cv2
import sys
import numpy as np

MAXVALUE = 255
K = 100  # scale factor for trackbar
gamma = 1.0  # init gamma value
GAMMA_MAX = int(6 * K)  # max gamma value 4.0

win_original_name = "Original Image"
win_corrected_name = "Gamma-corrected Image"


#  insert string before
def insert_name(name, str2add):
    dot_idx = name.find(".")
    new_name = name[:dot_idx] + str2add + name[dot_idx:]
    return new_name


#  gamma correction
def gamma_correct(img_in, gm):
    lut = np.zeros((1, MAXVALUE+1), dtype=np.uint8)  # init look-up table
    for i in range(256):
        lut[0, i] = np.clip(pow(i/MAXVALUE, gamma)*MAXVALUE, 0, MAXVALUE)
    img_out = cv2.LUT(img_in, lut)  # look-up table transform
    return img_out


#  trackbar helper function
def on_gamma_trackbar(val):
    global gamma
    gamma = val/K
    cv2.setTrackbarPos("Gamma", win_corrected_name, int(gamma*K))


if __name__ == "__main__":
    # get input arguments
    args = sys.argv
```

```
    assert (len(args) == 2)  # make sure two arguments input
    img_name = args[1]  # input image path

    cv2.namedWindow(win_original_name)
    cv2.namedWindow(win_corrected_name)

    # read original image
    img = cv2.imread(img_name)
    if img is None:
        sys.exit("Could not read the image.")
    cv2.imshow(win_original_name, img)

    gamma_init = int(gamma*K)  # init position in trackbar
    cv2.createTrackbar("Gamma", win_corrected_name, gamma_init, GAMMA_MAX,
on_gamma_trackbar)

    print("Press 'S' to save and exit")
    while True:
        # gamma correct image wrt trackbar val
        img_output = gamma_correct(img, gamma)
        cv2.imshow(win_corrected_name, img_output)

        key = cv2.waitKey(30)  # wait 30ms

        if key == ord("s"):
            #  press 'S' to save
            img_output_name = insert_name(img_name, "_gcorrected")
            cv2.imwrite(img_output_name, img_output)
            print("Adjusted gamma value is: " + str(gamma))
            break
        elif key == ord("q") or key == ord("x") or key == 27:
            # press 'q', 'x', 'ESC' to quit
            break

    cv2.destroyAllWindows()
```