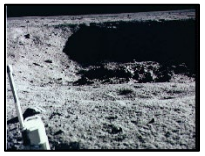
Rayan Baker Boudissa

Completed

Originals



```
# Rayan Baker Boudissa, 2/28/2026
from mediaComp import *

def collage():
    setMediaFolder()
    initialize()
    surface = drawBackground()
    for start in range(0, 2):
        colorStep = 1 - (start * 0.5) #color modification
        scaleStep = 0.8 + (start * 0.1) #size modification
        blur(chromaReverse(walk1, surface, 540 - (start * 100), 50 + (start*20), colorStep, scaleStep))
        blur(chromaReverse(walk2, surface, 490 - (start * 100), 60 + (start*20), colorStep-0.25, scaleStep+0.05))
    copyPicture(walk1, surface, 340, 90) #plain image
    for start in range(3, 5):
        colorStep = 1 - (start * 0.5) #color modification
        scaleStep = 0.8 + (start * 0.1) #size modification
        chromaReverse(walk2, surface, 590 - (start * 100), 40 + (start * 20), colorStep + 0.25, scaleStep)
        chromaReverse(walk1, surface, 540 - (start * 100), 50 + (start * 20), colorStep, scaleStep + 0.05)
    surface = chromaReverse(signature, surface, 10, 10, 0, 1) #draws signature
    pictureTool(surface)

def chromaReverse(source, background, target_x, target_y, strength, size): #copies an image onto another image
    scaledPic = scale(source, size) #resizes picture before chromakeying it
    for source_pixel in getPixels(scaledPic):
        x = getX(source_pixel)
        y = getY(source_pixel)
        red = getRed(source_pixel)
        blue = getBlue(source_pixel)
        green = getGreen(source_pixel)
        valBlueness = (blue / (1 + (red + green) / 2))
        if valBlueness < 1.5 or blue < 60: #excludes blue background
            background_pixel = getPixelAt(background, target_x + x, target_y + y)
            setColor(background_pixel,pixelMod(source_pixel,strength)) #tints original image and copies to new one
    return background
```

```
def pixelMod(pixel,step_check): #posterizes first image copy, uses strength variable to determine copy's iteration
    if step_check == 1 or step_check == -1:
        color = posterizeGray(pixel, step_check)
    else:
        color = tint(pixel, step_check)
    return color

def tint(pixel, strength): #modification: gives new color for each pixel
    red = getRed(pixel)
    green = getGreen(pixel)
    blue = getBlue(pixel)
    intensity = ((red + green + blue) // 3)
    redMod = intensity - red
    greenMod = intensity - green
    blueMod = intensity - blue
    color = makeColor(red + (redMod * strength), green + (greenMod * strength), blue + (blueMod * strength))
    #strength determines how close an RGB color value is to the pixel's color average
    return color

def posterizeGray(pixel, step_check):
    red_value = getRed(pixel)
    green_value = getGreen(pixel)
    blue_value = getBlue(pixel)
    luminance = (red_value + green_value + blue_value) // 3
    if step_check == 1:
        if luminance < 50:
            color = black
        elif 50 <= luminance <= 150:
            color = darkGray
        elif 151 <= luminance <= 200:
            color = lightGray
        else:
            color = white
    if step_check == -1:
        if luminance < 25:
            color = makeColor(10, 0, 65)
        elif  25<= luminance <= 50:
            color = makeColor(35, 10, 40)
        elif  51<= luminance <= 100:
            color = makeColor(100, 115, 85)
        elif 101 <= luminance <= 180:
            color = makeColor(230, 145, 165)
        else:
            color = makeColor(180, 210, 255)
```

```
    return color

def scale(source_picture, size): #modification: changes image size
    new_width = getWidth(source_picture) * size
    new_height = getHeight(source_picture) * size
    new_picture = makeEmptyPicture(new_width, new_height)
    source_x = 0
    for new_x in range(0,getWidth(new_picture)):
        source_y = 0
        for new_y in range(0,getHeight(new_picture)):
            color = getColor(getPixelAt(source_picture, int(source_x), int(source_y)))
            setColor(getPixelAt(new_picture, new_x, new_y), color)
            source_y = source_y + (1 / size)
        source_x = source_x + (1 / size)
    return new_picture

def copyPicture(source, destination, target_x, target_y): #copies an unedited image onto a background
    for source_pixel in getPixels(source):
        x = getX(source_pixel)
        y = getY(source_pixel)
        source_color = getColor(source_pixel)
        destPixel = getPixelAt(destination, x + target_x, y + target_y)
        setColor(destPixel, source_color)

def mergeBackground(startPoint): #merges image with horizontally-flipped copy of itself in a checkerboard pattern
    width = getWidth(background)
    height = getHeight(background)
#     for x in range(startPoint, width, 2):
#         for y in range(startPoint, height, 2):
#             sourcePixel = getPixelAt(surface, x, y)
#             destPixel = getPixelAt(background, x, y)
#             setColor(destPixel, getColor(sourcePixel))
    for x in range(startPoint, width, 2):
        for y in range(startPoint, height, 2):
            sourcePixel = getPixelAt(surface, x, y)
            destPixel = getPixelAt(background, width - x - 1, y)
            setColor(destPixel, getColor(sourcePixel))
    return background

def drawBackground():
    copyPicture(surface, background, 0, 0)
    merged = mergeBackground(0)
    merged = mergeBackground(1)
    return merged
```

```
def blur(picture):
    blur_picture = duplicatePicture(picture)
    for x in range(1, getWidth(picture) - 1):
        for y in range(1, getHeight(picture) - 1):
            center = getPixelAt(blur_picture, x, y)
            left = getPixelAt(picture, x - 1, y)
            right = getPixelAt(picture, x + 1, y)
            top = getPixelAt(picture, x, y - 1)
            bottom = getPixelAt(picture, x, y + 1)
            #takes the average color values between a pixel and four adjacent pixels
            new_red = (getRed(center) + getRed(left) + getRed(right) + getRed(top) + getRed(bottom)) // 5
            new_green = (getGreen(center) + getGreen(left) + getGreen(right) + getGreen(top)+getGreen(bottom))//5
            new_blue = (getBlue(center) + getBlue(left) + getBlue(right) + getBlue(top) + getBlue(bottom)) // 5
            setColor(center, makeColor(new_red, new_green, new_blue))
    return blur_picture

def initialize():
    global walk1
    walk1 = makePicture(getMediaFolder("mLeft1.jpg"))
    global walk2
    walk2 = makePicture(getMediaFolder("mLeft2.jpg"))
    global surface
    surface = makePicture(getMediaFolder("moon-surface.jpg"))
    global background
    background = makeEmptyPicture(getWidth(surface), getHeight(surface), white)
    global signature
    signature = makePicture(getMediaFolder("signature_edit.jpg"))
```