# Amos Taylor

Completed



Original



```
from mediaComp import *
from colorsys import *

## Amos Taylor
## 10/23/26
## Eye image taken by project creator

def collage():
    og_eye = makePicture(getMediaFolder("eye.jpg"))
    signature = makePicture(getMediaFolder("signature.jpg"))
    eye = cropToNew(og_eye, 140, 920, 40, 410)
    small_eye = scale(eye, 1/4)
    edge_eye = freshCopy(small_eye)
    edgeDetect(edge_eye, 5)
    poster_eye = freshCopy(small_eye)
    posterize4color(poster_eye)
    cyano_eye = freshCopy(small_eye)
    cyanotype(cyano_eye)
    iris = cropToNew(og_eye, 360, 680, 70, 325)
    rainbowiris = rainbowfy(iris, 1, 0.5)
    finalpicture = makeEmptyPicture(getWidth(og_eye),getHeight(og_eye))
    finalpicture = copyTo(og_eye, finalpicture, 0, 0)
    finalpicture = copyTo(rainbowiris, finalpicture, 360, 70)
    finalpicture = copyTo(edge_eye, finalpicture, 165, 450)
    finalpicture = copyTo(poster_eye, finalpicture, 360, 475)
    finalpicture = copyTo(cyano_eye, finalpicture, 555, 465)
    finalpicture = reverseCK(signature, finalpicture, 1000, 550)
    pictureTool(finalpicture)

def rainbowfy(image, intensity, satboost):
    for pixel in getPixels(image):
        x = getX(pixel)
        y = getY(pixel)
        r, g, b = getRed(pixel),getGreen(pixel),getBlue(pixel)
        h, s, v = rgb_to_hsv(r, g, b)
        h = (x+y)*intensity/(getWidth(image)+getHeight(image))
        s = s+satboost
        r, g, b = hsv_to_rgb(h, s, v)
        setColor(pixel, makeColor(int(r), int(g), int(b)))
    return image

def copyTo(source_picture, dest_picture, dest_x, dest_y):
    target_x = dest_x
    for source_x in range(0, getWidth(source_picture)):
        target_y = dest_y
        for source_y in range(0, getHeight(source_picture)):
            source_pixel = getPixelAt(source_picture, source_x, source_y)
            color = getColor(source_pixel)
            target_pixel = getPixelAt(dest_picture, target_x, target_y)
            setColor(target_pixel, color)
```

```
            target_y = target_y + 1
        target_x = target_x + 1
    return dest_picture

def cropToNew(picture, start_x, end_x, start_y, end_y):
    new_picture = makeEmptyPicture(end_x-start_x, end_y-start_y)
    target_x = 0
    for source_x in range(start_x, end_x):
        target_y = 0
        for source_y in range(start_y, end_y):
            source_pixel = getPixelAt(picture, source_x, source_y)
            color = getColor(source_pixel)
            target_pixel = getPixelAt(new_picture, target_x, target_y)
            setColor(target_pixel, color)
            target_y = target_y + 1
        target_x = target_x + 1
    return new_picture

def freshCopy(picture):
    new_picture = cropToNew(picture, 0, getWidth(picture), 0, getHeight(picture))
    return new_picture

def scale(source_picture, factor):
    new_width = int(getWidth(source_picture) * factor)
    new_height = int(getHeight(source_picture) * factor)
    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/factor
        source_x = source_x + 1/factor
    return new_picture

def luminance(pixel):
    red_value = getRed(pixel)
    green_value = getGreen(pixel)
    blue_value = getBlue(pixel)
    return (red_value + green_value + blue_value) // 3

def edgeDetect(picture, threshold):
    for pixel in getPixels(picture):
        x = getX(pixel)
        y = getY(pixel)
        if y < getHeight(picture) - 1 and x < getWidth(picture) - 1:
            bottom_right_pixel = getPixelAt(picture, x+1, y+1)
            this_luminance = luminance(pixel)
            bottom_right_luminance = luminance(bottom_right_pixel)
            if abs(bottom_right_luminance - this_luminance) > threshold:
                setColor(pixel, red)
            else:
                setColor(pixel, black)

def reverseCK(source_picture, dest_picture, dest_x, dest_y):
    target_x = dest_x
    for source_x in range(0, getWidth(source_picture)):
        target_y = dest_y
```

```
        for source_y in range(0, getHeight(source_picture)):
            source_pixel = getPixelAt(source_picture, source_x, source_y)
            color = getColor(source_pixel)
            if luminance(source_pixel) < 200:
                target_pixel = getPixelAt(dest_picture, target_x, target_y)
                setColor(target_pixel, color)
            target_y = target_y + 1
        target_x = target_x + 1
    return dest_picture

def posterize4color(picture):
    for pixel in getPixels(picture):
        lumi = luminance(pixel)
        if lumi < 50:
            setColor(pixel, black)
        elif lumi < 151:
            setColor(pixel, darkGray)
        elif lumi < 201:
            setColor(pixel, lightGray)
        else:
            setColor(pixel, white)

def cyanotype(picture):
    grayScaleImproved(picture)
    for pixel in getPixels(picture):
        r, g, b = getRed(pixel), getGreen(pixel), getBlue(pixel)
        if b < 63:
            setBlue(pixel, b*2)
        elif b <= 191:
            setBlue(pixel, b*1.3)
        else:
            setBlue(pixel, b*1.2)
        setGreen(pixel, g*0.75)
        setRed(pixel, r*0.75)

def grayScaleImproved(picture):
    for pixel in getPixels(picture):
        new_red = getRed(pixel) * 0.299
        new_green = getGreen(pixel) * 0.587
        new_blue = getBlue(pixel) * 0.114
        intensity = int(new_red + new_green + new_blue)
        setColor(pixel, makeColor(intensity, intensity, intensity))
```