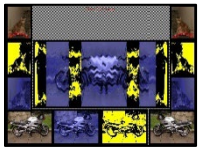
Aiden McKenzie

Completed



Originals

```
# Aiden McKenzie 10/24/2024

from jes4py import *

# Be aware it may take some time to load the collage

def collage():
    canvas = makeEmptyPicture(1000,736)
    checkerboard(canvas) # sets background pattern

    #### MOTORCYCLE SECTION
    bike = makePicture(getMediaPath("grayMotorcycle.jpg"))
    scaledBike = scaleDown(bike,.395) # used for supporting images
    centerBike = scaleDown(bike,.75) # used for centerpiece

    posterizedBike = posterize2color(scaledBike)
    addBorder(posterizedBike)
    copyInto(posterizedBike,canvas,getWidth(canvas)-getWidth(posterizedBike)*2+1,getHeight(canvas)-
getHeight(posterizedBike)+1)

    addBorder(scaledBike)
    #every copyInto is copying an image to the canvas
    copyInto(scaledBike,canvas,0,getHeight(canvas)-getHeight(scaledBike)+1)

    sepiaTint(scaledBike)
    copyInto(scaledBike,canvas,getWidth(canvas)-getWidth(scaledBike)*3+1,getHeight(canvas)-
getHeight(scaledBike)+1)

    #### CAT SECTION
    cat = makePicture(getMediaPath("mycat.jpg"))
    # used for centerpiece scaled specifically to match grayMotorcycle.jpg height
    centerCat = scaleDown(cat,.288231)
    cat=scaleDown(cat,.15) # used for supportings images
```

```
    posterizedCat = posterize2color(cat)
    addBorder(posterizedCat)
    copyInto(posterizedCat,canvas,0,0+getHeight(cat))

    addBorder(cat)
    copyInto(cat,canvas,0,0)

    sepiaTint(cat)
    copyInto(cat,canvas,0,0+getHeight(cat)*2)

    #### CENTER SECTION
    centerCat = posterize2color(centerCat) # applying effects to images before merge
    sepiaTint(centerBike)

    centerPic = merge(centerCat,centerBike,4)

    mirrorHorizontal(centerPic) # applying effects to images after merge
    mirrorVert2(centerPic)
    centerPic = wavyEffect(centerPic,5,.2)
    addBorder(centerPic)

    copyInto(centerPic,canvas,getWidth(canvas)//7,getHeight(canvas)//4)
    mirrorVert(canvas)

    sig = makePicture(getMediaPath("signature.jpg"))
    reverseChromakey(sig,canvas) # applying signature to canvas

    explore(canvas)

def mirrorVert(pic): #mirrors image on a vertical axis
    width = getWidth(pic)
    mirror_point = width // 4
    for x in range(0, mirror_point):
        for y in range(0, getHeight(pic)):
            left_pixel = getPixel(pic, x, y)
            right_pixel = getPixel(pic, width - x - 1, y)
            color = getColor(left_pixel)
            setColor(right_pixel, color)

def mirrorVert2(pic):
    #same as first except it reads the pixels from right to left instead of left to right
    #width = getWidth(pic) #also puts the mirror point at halfway instead of 1/4
    mirror_point = width // 2
    for x in range(0, mirror_point):
        for y in range(0, getHeight(pic)):
```

```
            right_pixel = getPixel(pic, width - x - 1, y)
            left_pixel = getPixel(pic, x, y)
            color = getColor(right_pixel)
            setColor(left_pixel, color)

def mirrorHorizontal(pic): #mirrors image on a horizontal axis
    height = getHeight(pic)
    mirror_point = height // 4
    for x in range(0, getWidth(pic)):
        for y in range(0, mirror_point):
            bottom_pixel = getPixel(pic, x, height - y - 1)
            top_pixel = getPixel(pic, x, y)
            color = getColor(bottom_pixel)
            setColor(top_pixel, color)

def reverseChromakey(source, background): #used to apply signature to canvas
    for source_pixel in getPixels(source):
        x = getX(source_pixel)
        y = getY(source_pixel)
        #checks if color of the pixel is not white
        if getRed(source_pixel) + getGreen(source_pixel) + getBlue(source_pixel) < 255:
            setColor(getPixel(background,x,y-280), red) #-280 determined by exploring signature.jpg

def wavyEffect(image,amplitude,frequency): #applies a sine wave pattern to pixels in an image
    width = getWidth(image)
    height = getHeight(image)
    newImage = makeEmptyPicture(width,height)
    for x in range(width):
        for y in range(height):
            #formula based on the function y = a *sin(b*x) + c
            newY = int(y + amplitude * math.sin(frequency*x))
            #checks to make sure y isn't out of bounds
            if 0 < newY < height:
                initialPixelLocation = getPixel(image,x,y)
                newPixelLocation = getPixel(newImage,x,newY)
                #moves pixel up or down depending on the wave
                setColor(newPixelLocation,getColor(initialPixelLocation))
    return newImage

def scaleDown(source_picture, scale_factor): #scales down an image
    new_width = getWidth(source_picture) *scale_factor
    new_height = getHeight(source_picture) *scale_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(getPixel(source_picture, source_x, source_y))
            setColor(getPixel(new_picture, new_x, new_y), color)
            source_y = source_y + 1/scale_factor
        source_x = source_x + 1/scale_factor
    return new_picture

def posterize2color(picture): #posterizes an image into two colors
    new_picture = makeEmptyPicture(getWidth(picture), getHeight(picture))
    for pixel in getPixels(picture):
        x = getX(pixel)
        y = getY(pixel)
        red_value = getRed(pixel)
        green_value = getGreen(pixel)
        blue_value = getBlue(pixel)
        luminance = (red_value + green_value + blue_value) // 3
        if luminance < 70:
            setColor(getPixel(new_picture,x,y), black)
        else:
            setColor(getPixel(new_picture,x,y), yellow)
    return new_picture

def addBorder(picture): #adds a border around all sides of an image
    border_thickness = 10
    bottom = getHeight(picture) - border_thickness
    right = getWidth(picture) - border_thickness
    for pixel in getPixels(picture):
        y = getY(pixel)
        x = getX(pixel)
        if y < border_thickness:
            setColor(pixel, black)
        if x < border_thickness:
            setColor(pixel,black)
        elif x >= right:
            setColor(pixel,black)
        elif y >= bottom:
            setColor(pixel, black)

def blackWaffle(src,startPoint): #creates a waffle pattern with black pixels
    for x in range(startPoint, getWidth(src),2):
        for y in range(startPoint, getHeight(src),2):
            setColor(getPixel(src, x, y), black)
```

```
def checkerboard(image):#uses blackWaffle function to create a checkerboard pattern
    blackWaffle(image,0)
    blackWaffle(image,1)

def grayScale(picture): #converts an image to gray scale
    for pixel in getPixels(picture):
        intensity = (getRed(pixel) + getGreen(pixel) + getBlue(pixel)) / 3
        setRed(pixel, intensity * 0.75)
        setGreen(pixel, intensity * 0.75)
        setBlue(pixel, intensity)

def sepiaTint(picture): #tints a picture blue
    grayScale(picture)
    for pixel in getPixels(picture):
        red_value = getRed(pixel)
        blue_value = getBlue(pixel)

        if red_value < 63:
            blue_value = blue_value * 2
        elif red_value < 192:
            blue_value = blue_value * 1.3
        else:
            if red_value > 255:
                red_value = 255
                blue_value = blue_value * 1.2

        setRed(pixel, red_value)
        setBlue(pixel, blue_value)

#creates empty box with a width that is the combined width of pic 1 and 2
# calls makeBar 1 and 2 to fill in the empty box
def merge(pic1, pic2, sections):
    width1 = getWidth(pic1)
    width2 = getWidth(pic2)
    height = getHeight(pic1)
    mergedPic = makeEmptyPicture(width1+width2,height,black)
    makeBar1(pic1,pic2,mergedPic,sections)
    makeBar2(pic1,pic2,mergedPic,sections)
    return mergedPic

def makeBar1(pic1,pic2,dstPic,sections):
    # start is where the loop will begin to copy pictures from
    start = 0
    # end is where it will stop copying pixels
    end = getWidth(pic1)//sections
```

```
    # offset is how far over the x will be pasted in the empty box
    offset = getWidth(pic2)//sections
    for section in range(sections):
        for x in range(start,end):
            for y in range(getHeight(pic1)):
                pixel = getPixel(pic1,x,y)
                dstPixel = getPixel(dstPic, x+offset, y)
                color = getColor(pixel)
                setColor(dstPixel,color)
        # sets everything to it's next value for the loop
        start = start+getWidth(pic1)//sections
        end = end+ getWidth(pic1)//sections
        offset = offset+getWidth(pic2)//sections

# same as makeBar1 but for pic2 with certain values changed to adjust where the bars are placed
def makeBar2(pic1,pic2,dstPic,sections):
    start = 0
    end = getWidth(pic2)//sections
    offset = 0
    for section in range(sections):
        for x in range(start, end):
            for y in range(getHeight(pic2)):
                pixel = getPixel(pic2,x,y)
                dstPixel = getPixel(dstPic, x+offset, y)
                color = getColor(pixel)
                setColor(dstPixel,color)
        start = start+getWidth(pic2)//sections
        end = end+ getWidth(pic2)//sections
        offset = offset + getWidth(pic1)//sections
```