```
"""
*****************************************************************************
 FILE:   threeD.py

 AUTHOR:     Peter Campbell

 ASSIGNMENT:  Lab 4: Image Processing

 DATE:  11/01/12

 DESCRIPTION: This is a program with one top level function 'makeAnaglyph'
 that uses a number of sub-programs to create a 3D image that can be
 viewed with 3D glasses. It uses two captured and imported images, one for the
 left eye and one for the right, and makes one 3D image.

*****************************************************************************
"""
from cs110graphics import *

def transparentPixel(pixel, trans = 100):
    """This program takes a 4-tuple representing a pixel and an int representin
  a transparency value and returns a new pixel with the given transparency"""

    red, green, blue, _ = pixel
    return (red, green, blue, trans)

def transparentImage(image, trans = 100):
    """Takes an image and a transparency and sets all the pixels in the image to
  the given transparency."""

    pixels = image.getPixels()
    for i in range(len(pixels)):
        pixels[i] = transparentPixel(pixels[i], trans)
    image.setPixels(pixels)
    return image

def threeDLeftImage(image):
    """Changes the given image by retaining red and alpha components of each
  pixel (makes the green and blue components zero)."""

    pixels = image.getPixels()
    for i in range(len(pixels)):
        red, _, _, alpha = pixels[i]
        pixels[i] = (red, 0, 0, alpha)
    image.setPixels(pixels)
    return image


def threeDRightImage(image):
    """Changes the given image by retaining green, blue, andalpha components of
  each pixel (makes the red component zero)."""

    pixels = image.getPixels()
    for i in range(len(pixels)):
        _, green, blue, alpha = pixels[i]
        pixels[i] = (0, green, blue, alpha)
    image.setPixels(pixels)
    return image

def combinedImages(imageA, imageB):
    """takes two identically-sized images and returns a new image in which each
  pixel is constructed by adding the color components of the corresponding
  pixels in imageA and imageB. The alpha component of each pixel comes from
```

Good!

```
64    imageA. Returns a new image. The original images are not modified."""
65
66        #indexes the colors of each pixel and then makes a new pixel by adding the
67        #two values and using the alpha value from the pixel from imageA.
68        newPixels = []
69        pixelsA = imageA.getPixels()
70        pixelsB = imageB.getPixels()
71        for i in range(len(pixelsA)):
72            redA = pixelsA[i][0]
73            greenA = pixelsA[i][1]
74            blueA = pixelsA[i][2]
75            alphaA = pixelsA[i][3]
76            redB = pixelsB[i][0]
77            greenB = pixelsB[i][1]
78            blueB = pixelsB[i][2]
79            newPixel = (redA + redB, greenA + greenB, blueA + blueB, alphaA)
80            newPixels.append(newPixel)
81        newImage = imageA.clone()
82        newImage.setPixels(newPixels)
83        return newImage
84
85    def cropImage(image, loc, size):
86        """Creates a smaller version (size x size) of the image with center at loc
87      in the original image. This is done by cropping. The original image is not
88      modified."""
89
90        #this makes sure that the point where the user clicked to crop the images is
91        #within a border of 300 so that the image can be cropped properly. If the
92        #image is within the border instead of asking the user for a new point it
93        #just adds or subtracts from the value until its inside the border
94        #CITE: TA Carson
95        #DETAILS: Carson helped be understand that I needed while loops for each
96        #possibility of botht the x and y values.
97        cx, cy = loc.get()
98        width, height = image.size()
99        while cx < 300:
100           cx += 1
101       while cx > width - 300:
102           cx -= 1
103       while cy < 300:
104           cy += 1
105       while cy > height - 300:
106           cy -= 1
107       left = cx - size / 2
108       right = width - (cx + size / 2)
109       top = cy - size / 2
110       bottom = height - (cy + size / 2)
111       cropImg = image.clone()
112       cropImg.crop(top, bottom, left, right)
113       return cropImg
114
115   def getMouse(win, prompt):
116       """A program that makes sure that the user is giving a mouse click when one
117     is desired and prompts the user if they are giving a different event."""
118
119       ev = win.wait(prompt)
120       while ev.getDescription() != 'mouse release':
121           print "Wrong event type " + ev.getDescription()
122           sleep(1)
123           ev = win.wait(prompt)
124       return ev.getMouseLocation()
125
126   def isInside(pt, ul, lr):
```

Nice! Good comments

```
127         """A program that tests to see if a point is inside a box defined by an
128     upper left and lower right point. It returns true if it is and false if it
129     is not."""
130
131         x, y = pt.get()
132         xu, yu = ul.get()
133         xl, yl = lr.get()
134         return x >= xu and x <= xl and y >= yu and y <= yl
135
136
137 def clickInside(win, ul, lr, prompt, errmess):
138         """A program that tests if a mouse click from the user is inside a desired
139     box region. It uses the 'isInside' program and prompts the user if the are
140     not clicking in the desired region."""
141
142         pt = getMouse(win, prompt)
143         while not isInside(pt, ul, lr):
144             print errmess
145             sleep(1)
146             pt = getMouse(win, prompt)
147         return pt
148
149
150 def cropOverLayedImages(pt1, pt2, cropPt, leftImg, rightImg):
151         """This program takes two images that the user wants to be put on top of
152     each other and crops them so that no part of either image sticks out and
153     that they are then perfectly aligned. It does this by taking the two points
154     from either image that the user wants to align and calculates their real
155     values in the original images. Then if finds the difference between the two
156     and crops them accordingly with two if-statements."""
157
158         x1, y1 = pt1.get()
159         x2, y2 = pt2.get()
160         cx, cy = cropPt.get()
161         rx1 = cx + x1 - 300
162         ry1 = cy + y1 - 300
163         rx2 = cx + x2 - 900
164         ry2 = cy + y2 - 300
165         dx = rx2 - rx1
166         dy = ry2 - ry1
167
168         #these if-statements make sure the the images are being cropped on the
169         #correct sides. It does this by looking at the change in the aligned points
170         #and seeing if they are positive or negative.
171         #CITE:TA Carson
172         #DETAILS: Helped me understand that the image may have to be cropped
173         #differently depending on the aligned points and suggested if-statements and
174         #absolute values.
175         if dx < 0:
176             leftImg.crop(0, 0, abs(dx), 0)
177             rightImg.crop(0, 0, 0, abs(dx))
178         else:
179             leftImg.crop(0, 0, 0, dx)
180             rightImg.crop(0, 0, dx, 0)
181         if dy < 0:
182             leftImg.crop(0, abs(dy), 0, 0)
183             rightImg.crop(abs(dy), 0, 0, 0)
184         else:
185             leftImg.crop(dy, 0, 0, 0)
186             rightImg.crop(0, dy, 0, 0)
187         return [leftImg, rightImg]
188
189
```

Nice!

```
def makeAnaglyph():
    """This is the main function that uses the above sub-programs to formulate
  the threeD image. It accepts the files for the left and right image from the
  user, converts them to images and then adds the left image to the user. It
  crops both of them after asking the user for a focal point. It then displays
  the new cropped imaged next to each other. The user clicks a point in each
  image that they want to align and then crops them so when the images are put
  on top of each other, aligned at those points, they make a box and no part
  of either image sticks out. It then uses the sub programs to combine the two
  images and make one threeD image."""

    leftImg = Image(OpenFileDialog("Select the left image"))
    rightImg = Image(OpenFileDialog("Select the right image"))
    width, height = leftImg.size()
    win = Window(width, height)
    win.add(leftImg)
    leftImg.moveTo(Point(width / 2, height / 2))
    cropPoint = getMouse(win, "Select a focal point")
    win.close()
    newLeft = cropImage(leftImg, cropPoint, 600)
    newRight = cropImage(rightImg, cropPoint, 600)
    width2, height2 = newLeft.size()
    win2 = Window(2 * width2, height2)
    win2.add(newLeft)
    newLeft.moveTo(Point(width2 / 2, height2 / 2))
    win2.add(newRight)
    newRight.moveTo(Point(width2 / 2 * 3, height2 /2))
    tp1 = clickInside(win2, Point(0, 0), Point(width2, height2), "Click the " +
                      "pixel in the left image that you would like to align",
                      "No, the left image silly.")
    tp2 = clickInside(win2, Point(width2, 0), Point(2 *  width2, height2),
                      "Click the pixel in the right image that you would " +
                      "like to align", "No, the right image silly.")
    win2.remove(newLeft)
    win2.remove(newRight)
    win2.close()
    finalLeft, finalRight = cropOverLayedImages(tp1, tp2, cropPoint, leftImg,
                                                rightImg)
    finalLeft = threeDLeftImage(leftImg)
    finalRight = threeDRightImage(rightImg)
    finalImage = combinedImages(finalLeft, finalRight)
    widthf, heightf = finalImage.size()
    win3 = Window(widthf, heightf)
    win3.add(finalImage)
    finalImage.moveTo(Point(widthf / 2, heightf / 2))
    finalImage.save(SaveFileDialog("Save your image by naming it"))
    win3.wait("Click to close the window")
    win3.close()


if __name__ == "__main__":
    StartGraphicsSystem(makeAnaglyph)
```

A little excessive but in description

white space?

"