1. Suppose I TRULY wanted to make a "black and white" photo (nothing but black pixels and white pixels).

a) What would I need to do in order to make that image? Light colors to white and dark to black.

This is an example of *posterizing* -- reducing the number colors to a much smaller number (e.g., 2).

```
def blackAndWhite(picture):
  for y in range(0 , getHeight(picture)):
    for x in range(0 , getWidth(picture)):
      pixel = getPixel(picture,x,y)

      total=getRed(pixel)+getGreen(pixel)+getBlue(pixel)

      luminance=total/3

      if luminance < 128:
        setColor(pixel,black)
      else:
        setColor(pixel,white)
```

2. You might recognize a recent classic posterization example and a more recently "Obamified" President Ruud.

a) Besides luminance, how might you select a color for posterization?

Select color that has minimum distance to pixel's color

```
def obamifyMe(picture):
  ltBlue = makeColor(113,150,159)
  myRed = makeColor(215,26,33)
  myBlue = makeColor(0,50,77)
  myTan = makeColor(252,228,168)
  for y in range(0 , getHeight(picture)):
    for x in range(0 , getWidth(picture)):
      pixel = getPixel(picture,x,y)
      total=getRed(pixel)+getGreen(pixel)+getBlue(pixel)
      luminance=total/3
      if luminance<64:
        setColor(pixel,myBlue)
      elif luminance<128:
        setColor(pixel,myRed)
      elif luminance<192:
        setColor(pixel,ltBlue)
      else:
        setColor(pixel,myTan)
```

b) Complete the following code based on this method.

```
def obamifyMe(picture):
  ltBlue = makeColor(113,150,159)
  myRed = makeColor(215,26,33)
  myBlue = makeColor(0,50,77)
  myTan = makeColor(252,228,168)
  for y in range(0 , getHeight(picture)):
    for x in range(0 , getWidth(picture)):
      pixel = getPixel(picture,x,y)
      pixelColor = getColor(pixel)
      minDistance = distance(pixelColor, ltBlue)
      minColor = ltBlue
      if distance(pixelColor, myRed) < minDistance:
        minDistance = distance(pixelColor, myRed)
        minColor = myRed
      if distance(pixelColor, myBlue) < minDistance:
        minDistance = distance(pixelColor, myBlue)
        minColor = myBlue
      if distance(pixelColor, myTan) < minDistance:
        minDistance = distance(pixelColor, myTan)
        minColor = myTan
      setColor(pixel, minColor)
```

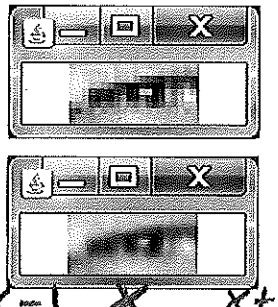
c. How might posterization be used as a picture compression technique?

Instead of 24-bit color RGB value stored in the file we could have a table

Color	RGB	compressed #
1	Blue (113, 150, 150)	0
		1
		2

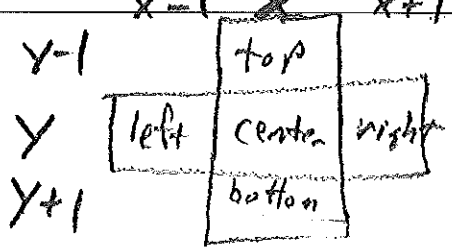
each only needs 2 bits / pixel

3. When we scaled up our picture (1 pixel became a square of 4) we get a grainy (i.e., rough saw-tooth edges, etc.) image due to *pixelation*. For example, after scaling up President Ruud's picture twice, his left eye looks like (top eye):



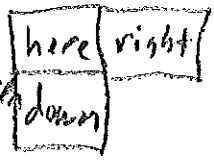
We can reduce pixelation by *blurring* an image. A simple way to blur an image is to set each pixel to a color that is the average of the pixels around it. Blurring after each scaling we get the bottom eye to the right. a) Complete the blur code below:

b) Why do the x and y range parameters stop one short of the borders? Avoids running off edge of picture




```
def blur(pict):
  """ Blurs a pict by setting each pixel's color to the
      average of its four neighbors. """

  for x in range(1,getWidth(pict)-1):
    for y in range(1,getHeight(pict)-1):
      center = getPixel(pict, x, y)
      top = getPixel(pict, x, y-1)

      bottom = getPixel(pict, x, y+1)

      right = getPixel(pict, x+1, y)

      left = getPixel(pict, x-1, y)

      newRed = (getRed(center)+getRed(top)+getRed(bottom)+getRed(right)+getRed(left))/5

      newGreen=(getGreen(center)+getGreen(top)+getGreen(bottom)      \
                + getGreen(right)+getGreen(left))/5

      newBlue = (getBlue(center) + getBlue(top) + getBlue(bottom) + getBlue(right)
                 + getBlue(left))/5

      setColor(center, makeColor(newRed, newGreen, newBlue))
```

4. How does the following edge detection code work?



If there is a "big" change in luminance between "here" and both "down" and "right", then an edge is detected.

```
def edgeDetector(pict):
  """ Draws a black-and-white line drawing using edge detection. """
  for x in range(getWidth(pict)-1):
    for y in range(1,getHeight(pict)-1):
      here = getPixel(pict, x, y)
      down = getPixel(pict, x, y+1)
      right = getPixel(pict, x+1, y)

      hereLum = (getRed(here)+getGreen(here)+getBlue(here))/3
      downLum = (getRed(down)+getGreen(down)+getBlue(down))/3
      rightLum = (getRed(right)+getGreen(right)+getBlue(right))/3

      if abs(hereLum - downLum) > 10 and abs(hereLum - rightLum) > 10:
        setColor(here, black)
      if abs(hereLum - downLum) <= 10 or abs(hereLum - rightLum) <= 10:
        setColor(here, white)
```

b) Could we make the body of the second `if`-statement the `else` of the first `if`-statement without changing the results?

Yes, just make second if-statement "else" of 1st