**1. What does RGBA stand for?**

In Python, RGBA stands for Red, Green, Blue, and Alpha and it is used to specify the color and transparency of a graphic element in the same way as in computer graphics in general. RGBA values in Python can be represented as a tuple of four values, ranging from 0 to 255, which represent the intensity of red, green, and blue light, and the alpha value, respectively. For example, in Python, the color white can be represented as (255, 255, 255, 255), while a partially transparent blue can be represented as (0, 0, 255, 128), where 128 represents 50% transparency. RGBA values can be used with the built-in Python libraries such as matplotlib and pygame for creating and visualizing graphics.

**2. From the Pillow module, how do you get the RGBA value of any images?**

To get the RGBA value of an image using the Python Imaging Library (PIL), which is now known as the Pillow library, you can use the following code:

from PIL import Image

img = Image.open("path/to/image.jpg") # open image file

rgba\_data = img.getdata(mode='RGBA') # get RGBA data

# Iterate through the data and print each pixel's RGBA value

for pixel in rgba\_data:

r, g, b, a = pixel

print(r, g, b, a)

In this code, img.open() opens an image file, and img.getdata(mode='RGBA') returns a sequence of RGBA values for each pixel in the image. The for loop iterates through the sequence of RGBA values and unpacks each pixel's RGBA value into separate variables r, g, b, and a, which represent the intensity of red, green, blue, and alpha, respectively. The print statement outputs each pixel's RGBA value.

Note that the mode argument in img.getdata(mode='RGBA') specifies that you want to get the RGBA values for each pixel in the image. If you want to get the RGB values, you can use mode='RGB'.

**3. What is a box tuple, and how does it work?**

In Python, a box tuple is a tuple that represents the bounding box of an object in an image. A bounding box is a rectangle that surrounds an object in an image and defines its position and size.

A box tuple is a 4-tuple that consists of four values: the x-coordinate and y-coordinate of the top-left corner of the bounding box, and the width and height of the bounding box. For example, a bounding box with top-left corner at (100, 200) and a width of 50 and a height of 80 can be represented as a box tuple (100, 200, 50, 80).

In Python, box tuples are used in various computer vision and image processing tasks to define the location and size of objects in images. For example, you can use a box tuple to crop an object from an image, to draw a rectangle around an object, or to determine the overlap between two bounding boxes.

**4. Use your image and load in notebook then, How can you find out the width and height of an Image object?**

To find out the width and height of an Image object in Python, you can use the following code:

from PIL import Image

img = Image.open("path/to/image.jpg") # open image file

width, height = img.size # get the width and height of the image

print("Width:", width)

print("Height:", height)

In this code, img.open() opens an image file and returns an Image object. The img.size property returns a tuple containing the width and height of the image. The print statements output the width and height of the image.

Note that the Image object is created using the Python Imaging Library (PIL), which is now known as the Pillow library.

**5. What method would you call to get Image object for a 100×100 image, excluding the lower-left quarter of it?**

To get an Image object for a 100x100 image that excludes the lower-left quarter of it in Python, you can use the following code:

from PIL import Image

img = Image.open("path/to/image.jpg") # open image file

width, height = img.size # get the width and height of the image

box = (0, 0, width//2, height//2) # define a box tuple for the upper-right quarter of the image

cropped\_img = img.crop(box) # crop the object using the box tuple

In this code, img.open() opens an image file and returns an Image object. The img.size property returns a tuple containing the width and height of the image. The box tuple defines the bounding box for the upper-right quarter of the image, with the top-left corner located at (0, 0) and the width and height equal to half of the original image's width and height, respectively. The img.crop(box) method crops the object defined by the box tuple from the image and returns a new Image object.

**6. After making changes to an Image object, how could you save it as an image file?**

To save the changes made to an Image object as an image file in Python, you can use the save method of the Image object. For example, if you have an Image object img, you can save it as a JPEG file using the following code:

img.save("new\_image.jpg", "JPEG")

In this code, img.save("new\_image.jpg", "JPEG") saves the Image object img as a JPEG image file with the name "new\_image.jpg". The second argument, "JPEG", is the format of the saved image file.

You can also save the Image object as other image formats, such as PNG, GIF, BMP, and TIFF, by using the corresponding format string as the second argument to the save method. For example, to save the Image object as a PNG file, you can use the following code:

img.save("new\_image.png", "PNG")

Note that the Python Imaging Library (PIL), which is now known as the Pillow library, supports many image formats, and the format string that you use in the save method should be one of the supported formats.

**7. What module contains Pillow’s shape-drawing code?**

Pillow's shape-drawing code is contained in the ImageDraw module in Python. The ImageDraw module provides simple 2D graphics drawing capabilities for use with the Image class. You can use the ImageDraw module to draw shapes, such as lines, rectangles, ellipses, and polyglines, on an Image object.

Here's an example of how to use the ImageDraw module to draw a rectangle on an Image object in Python:

from PIL import Image, ImageDraw

img = Image.new("RGBA", (200, 200), (255, 255, 255, 255)) # create a new image with RGBA mode and white background

draw = ImageDraw.Draw(img) # create a draw object

# draw a red rectangle on the image

draw.rectangle((20, 20, 180, 180), fill=(255, 0, 0, 255))

img.save("rectangle.png") # save the image with the rectangle

In this code, Image.new creates a new image with a size of 200x200 and a white background. The ImageDraw.Draw function creates a draw object that can be used to draw on the image. The draw.rectangle function draws a rectangle on the image with its top-left corner at (20, 20) and its bottom-right corner at (180, 180), and with a red fill color. Finally, the img.save method saves the image with the rectangle to a PNG file with the name "rectangle.png".

**8. Image objects do not have drawing methods. What kind of object does? How do you get this kind of object?**

Image objects in the Pillow library do not have drawing methods. To draw shapes and text on an Image object, you need to create a ImageDraw object. The ImageDraw object is created by calling the ImageDraw.Draw method and passing in the Image object as an argument.

Here's an example of how to create a ImageDraw object in Python:

from PIL import Image, ImageDraw

img = Image.new("RGBA", (200, 200), (255, 255, 255, 255)) # create a new image with RGBA mode and white background

draw = ImageDraw.Draw(img) # create a draw object

In this code, Image.new creates a new image with a size of 200x200 and a white background. The ImageDraw.Draw function creates a draw object that can be used to draw on the image. You can use the draw object to draw shapes and text on the img Image object.