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

RGBA stands for Red Green Blue Alpha.

**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 Pillow module in Python, you can use the `getpixel()` method of the `Image` class. The `getpixel()` method returns a tuple containing the RGBA values of the pixel at the specified location.

Here's an example of how to get the RGBA value of a pixel at position (x, y) in an image:

```python

from PIL import Image

# Open an image file

image = Image.open('image.png')

# Get the RGBA value of the pixel at (x, y)

r, g, b, a = image.getpixel((x, y))

# Print the RGBA values

print('Red: ', r)

print('Green: ', g)

print('Blue: ', b)

print('Alpha: ', a)

```

In this example, we first open an image file using the `Image.open()` method. Then, we use the `getpixel()` method to get the RGBA value of the pixel at the specified location `(x, y)`. The method returns a tuple containing the red, green, blue, and alpha values of the pixel, which we then assign to the variables `r`, `g`, `b`, and `a`, respectively. Finally, we print the RGBA values using the `print()` function.

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

In Pillow, a box tuple is a tuple of four integers that defines a rectangular region in an image. The four integers represent the left, upper, right, and lower pixel coordinates of the rectangle, in that order. The syntax for a box tuple is as follows:

```

(left, upper, right, lower)

```

Here's an example of how to use a box tuple to crop an image:

```python

from PIL import Image

# Open an image file

image = Image.open('image.png')

# Define a box tuple for the region to crop

box = (100, 100, 300, 300)

# Crop the image to the specified region

cropped\_image = image.crop(box)

# Save the cropped image

cropped\_image.save('cropped\_image.png')

```

from PIL import Image

# Open an image file

image = Image.open('image.png')

# Define a box tuple for the region to crop

box = (100, 100, 300, 300)

# Crop the image to the specified region

cropped\_image = image.crop(box)

# Save the cropped image

cropped\_image.save('cropped\_image.png')

In this example, we first open an image file using the `Image.open()` method. Then, we define a box tuple that specifies the rectangular region to crop, with the upper left corner at (100, 100) and the lower right corner at (300, 300). We pass the box tuple to the `crop()` method of the `Image` object to extract the specified region from the image. Finally, we save the cropped image using the `save()` method.

**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 Pillow, you can use the `size` attribute of the image. Here's an example:

```python

from PIL import Image

# Open an image file

image = Image.open('image.png')

# Get the width and height of the image

width, height = image.size

# Print the width and height

print('Width:', width)

print('Height:', height)

```

In this example, we first open an image file using the `Image.open()` method. Then, we get the width and height of the image by accessing the `size` attribute of the `Image` object. Finally, we print the width and height using the `print()` function.

**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, excluding the lower-left quarter, you can first create a new `Image` object with a solid color using the `Image.new()` method, and then use the `Image.paste()` method to paste the original image onto the new image with a box tuple that excludes the lower-left quarter.

Here's an example:

```python

from PIL import Image

# Open an image file

image = Image.open('image.png')

# Create a new image with a solid color

new\_image = Image.new('RGB', (100, 100), (255, 255, 255))

# Define a box tuple for the region to paste

box = (0, 0, 50, 50)

# Paste the original image onto the new image, excluding the lower-left quarter

new\_image.paste(image, box)

# Save the modified image

new\_image.save('modified\_image.png')

```

In this example, we first open an image file using the `Image.open()` method. Then, we create a new image with a solid white color and size 100x100 using the `Image.new()` method. We define a box tuple that specifies the region to paste the original image onto, excluding the lower-left quarter. We then use the `Image.paste()` method to paste the original image onto the new image with the specified box tuple. Finally, we save the modified image using the `save()` method.

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

After making changes to an `Image` object, you can save it as an image file using the `save()` method.

Here's an example:

```python

from PIL import Image

# Open an image file

image = Image.open('image.png')

# Rotate the image 90 degrees

image = image.rotate(90)

# Save the modified image as a new file

image.save('rotated\_image.png')

```

In this example, we first open an image file using the `Image.open()` method. We then rotate the image by 90 degrees using the `rotate()` method and save the modified image as a new file using the `save()` method, which takes the filename to save the image to as an argument. The `save()` method can also take additional options such as file format, compression level, and metadata.

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

Pillow's shape-drawing code is contained in the `ImageDraw` module. The `ImageDraw` module provides simple 2D graphics for Image objects, allowing you to draw lines, rectangles, ellipses, polygons, text, and more. It also provides support for alpha compositing, antialiasing, and arbitrary transformations.

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

`Image` objects in Pillow do not have built-in drawing methods. Instead, you can use the `ImageDraw` module to create a `Draw` object, which has drawing methods that allow you to draw on an `Image` object.

Here's an example:

```python

from PIL import Image, ImageDraw

# Open an image file

image = Image.open('image.png')

# Create a drawing object

draw = ImageDraw.Draw(image)

# Draw a red rectangle

draw.rectangle((50, 50, 100, 100), fill='red')

# Save the modified image as a new file

image.save('rectangle.png')

```

In this example, we first open an image file using the `Image.open()` method. We then create a `Draw` object using the `ImageDraw.Draw()` method and pass in the `Image` object to draw on. We can then use the drawing methods of the `Draw` object to draw on the image. In this case, we draw a red rectangle using the `rectangle()` method. Finally, we save the modified image as a new file using the `save()` method.