

Images

For the next week, we'll focus on an important area of computer science called **image processing**.

What is an image?

You're probably accustomed to working with .jpg, .gif, .png, and other types of images on your computer. How do computers store the data that make up an image? It's represented as a 2D regular grid of pixel values. We'll say that there are X columns and Y rows. Let's draw this out on the board.

Recall that Python 'counts' starting at 0 and goes to $n-1$. So the pixel numbering in the grid go from (0, 0) on the top-left corner to ($X-1$, $Y-1$) on the bottom right corner, where again **X is the image width in pixels** and **Y is the image height in pixels** (X columns = width, Y rows = height). This seemingly strange coordinate system is common to images and computer graphics.

In the case of **grayscale images** (computer just stores a single intensity value at each pixel...you can think of the value as an integer ranging from 0 for black and 255 for white). We'll often be working with **color images**, which store the color at each position of the 2D image grid a **(R, G, B) triplet**, which specifies the amount of red, green, and blue color. So the dimensions of a color image are (X , Y , 3). The values for each RGB value would also be 0 to 255 inclusive. For example, (0, 255, 0) for a green colored pixel.

What would be a white pixel value?

What would be a black pixel value?

The PPM format

Zelle's Graphics module provides a convenient way for us to work this complex type of data. It assumes your image is in the format you may never have heard of called **.ppm**. There are instructions in the project this week how to convert your images from .jpg/.png/etc to this format (if you want to use your own images).

Images in Zelle Graphics

1. Make and show a image using `Image` object in Zelle Graphics library (`lecture_19_show.py`).
2. Make a rainbow RGB image using nested loops and an `Image` object (`lecture_19_rainbow_image_fun`).