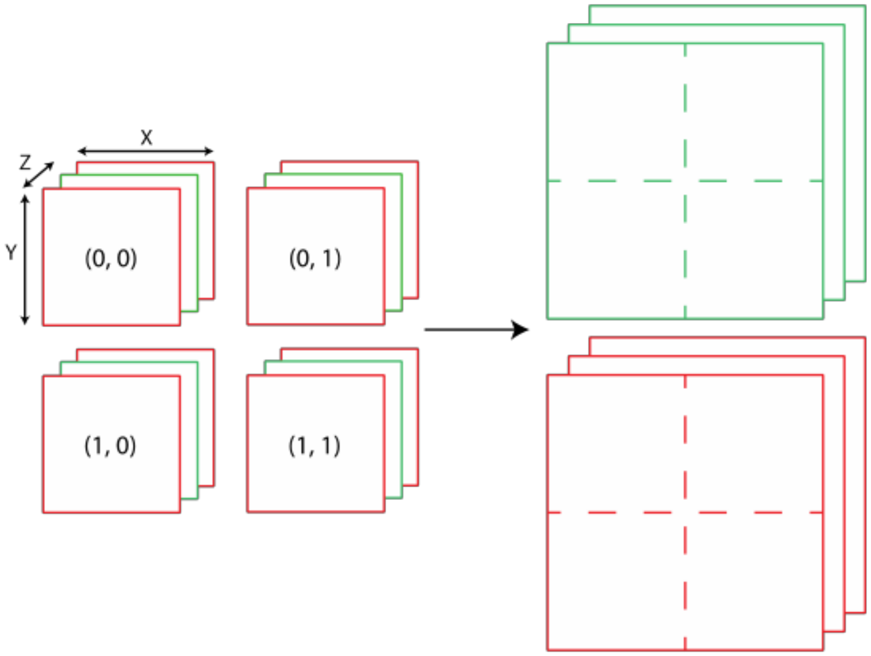
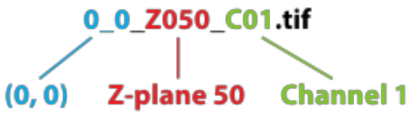
Task:

Our microscope generates 3D images of the brain by acquiring 2D image (x,y axis) slices at regular intervals along the length of the brain (z axis). These images are acquired in two color channels (red and green). Since the sample is often too large to fit within the field of view of the camera, we take multiple 3D image sub-stacks in a tile pattern to cover the brain. We need to merge these sub-stacks into a single 3D image. The folder “test-data” contains multichannel (i.e. red, green) image data that was acquired using a 2x2 grid of tiles. Within the folder you will find a series of images in TIFF format. Each file in this folder represents a single image from a 3D image sub-stack, from a single channel, from a single position along the Z axis. Your task is to write a function that will output 2 files containing 3D image arrays, sorted by channel, that joins all of the tiles in x, y and z. The final output should be a single TIFF file for each channel.



The name of each file corresponds to the tile’s position in xy, position along the z-axis, and the channel that image corresponds to.



Requirements:

- Written in python3.6+
- Accepts a single argument: Directory name

- The function must be able to accept data with an arbitrary number of tiles and channels. The geometry of the tiles can be expected to fill a rectangular grid.
- Data should be loaded using the Tifffile library. This will load the data as a numpy array and therefore any manipulations should be done using numpy functions/formatting.
- This is a downsampled version of a real dataset that is over 500GB. Because these output files are often larger than the amount of RAM available, they should be populated using memory-mapped IO. The Tifffile library provides a method (tifffile.memmap) for creating a memory-mapped output arrays.

URLs:

- Numpy memory-mapping: https://docs.scipy.org/doc/numpy/reference/generated/numpy.memmap.html

- Tifffile library: https://pypi.org/project/tifffile/

Example Usage:

```
Create an empty TIFF file and write to the memory-mapped numpy array:

>>> memmap_image = memmap('temp.tif', shape=(256, 256), dtype='uint16')
>>> print(memmap_image)

[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]

>>> memmap_image[:] = 1
>>> del memmap_image
>>>  rememmap_image = tif.memmap('temp.tif')
>>> print(rememmap_image)

[[1 1 1 ... 1 1 1]
 [1 1 1 ... 1 1 1]
 [1 1 1 ... 1 1 1]
 ...
 [1 1 1 ... 1 1 1]
 [1 1 1 ... 1 1 1]
 [1 1 1 ... 1 1 1]]

>>> rememmap_image.shape, rememmap_image.dtype

((256, 256), dtype('uint16'))
```