

Saving and loading arrays

INTRODUCTION TO NUMPY

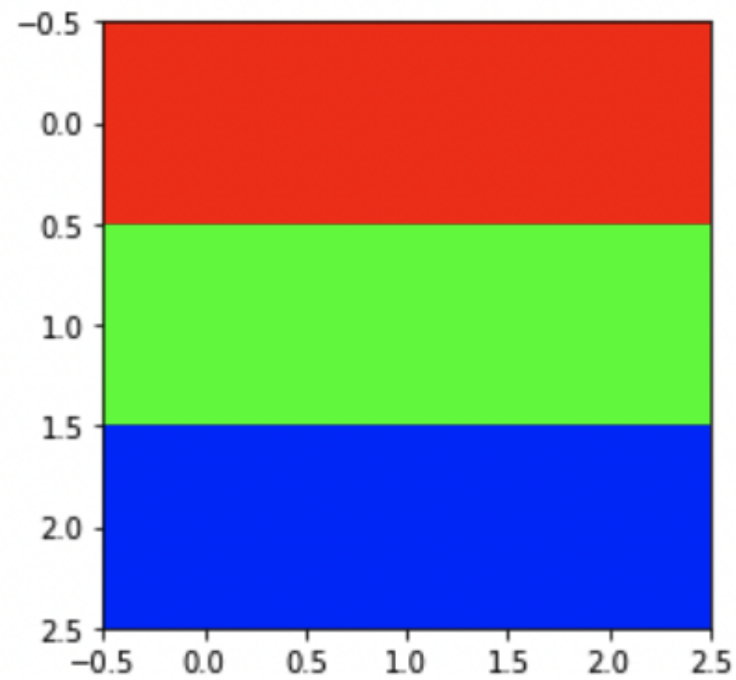


Izzy Weber

Curriculum Manager, DataCamp

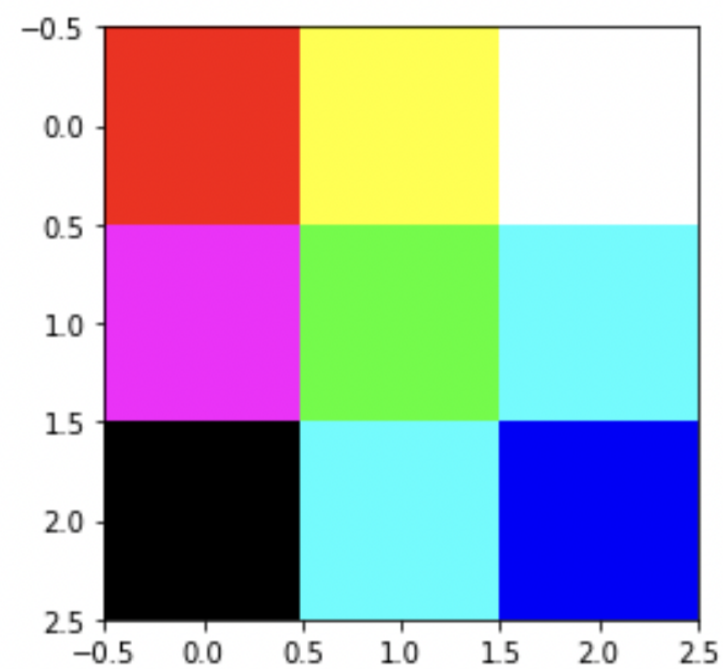
RGB arrays

```
rgb = np.array([[[255, 0, 0], [255, 0, 0], [255, 0, 0]],  
               [[0, 255, 0], [0, 255, 0], [0, 255, 0]],  
               [[0, 0, 255], [0, 0, 255], [0, 0, 255]]])  
  
plt.imshow(rgb)  
plt.show()
```



RGB arrays

255	0	0	255	255	0	255	255	255
255	0	255	0	255	0	0	255	255
0	0	0	0	255	255	0	0	255

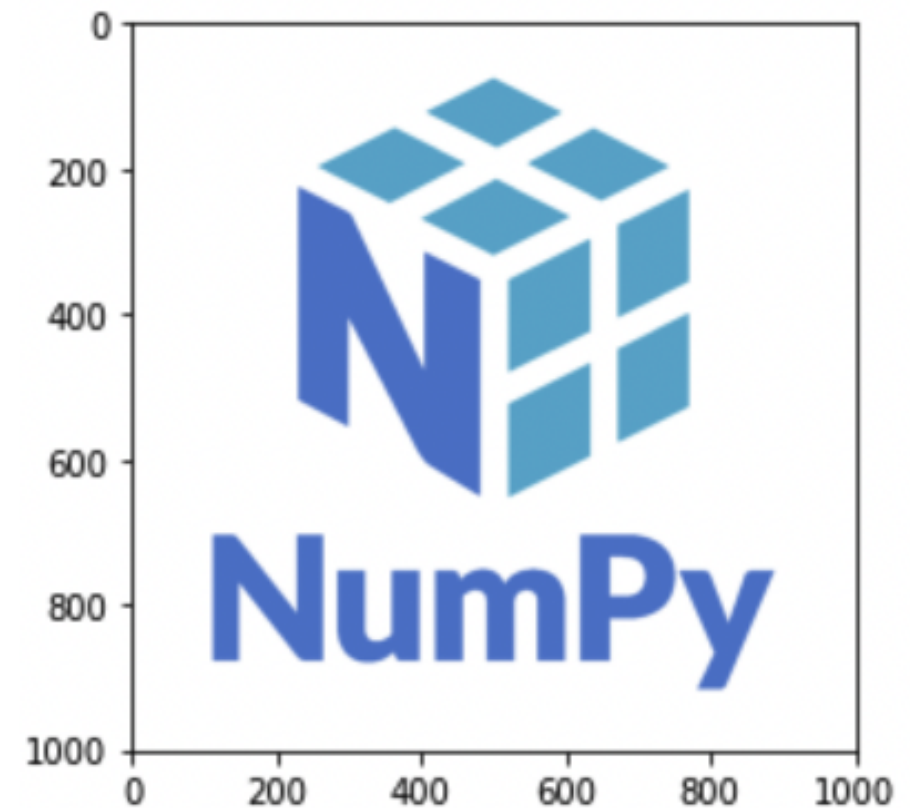


Loading .npy files

Save arrays in many formats:

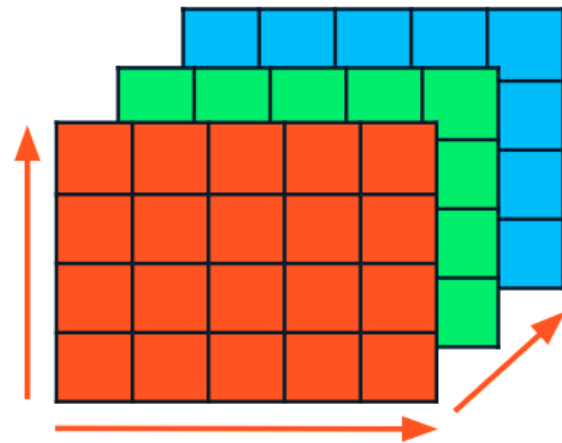
- .csv
- .txt
- .pkl
- .npy

```
with open("logo.npy", "rb") as f:  
    logo_rgb_array = np.load(f)  
plt.imshow(logo_rgb_array)  
plt.show()
```



Examining RGB data

```
red_array = logo_rgb_array[:, :, 0]  
blue_array = logo_rgb_array[:, :, 1]  
green_array = logo_rgb_array[:, :, 2]
```



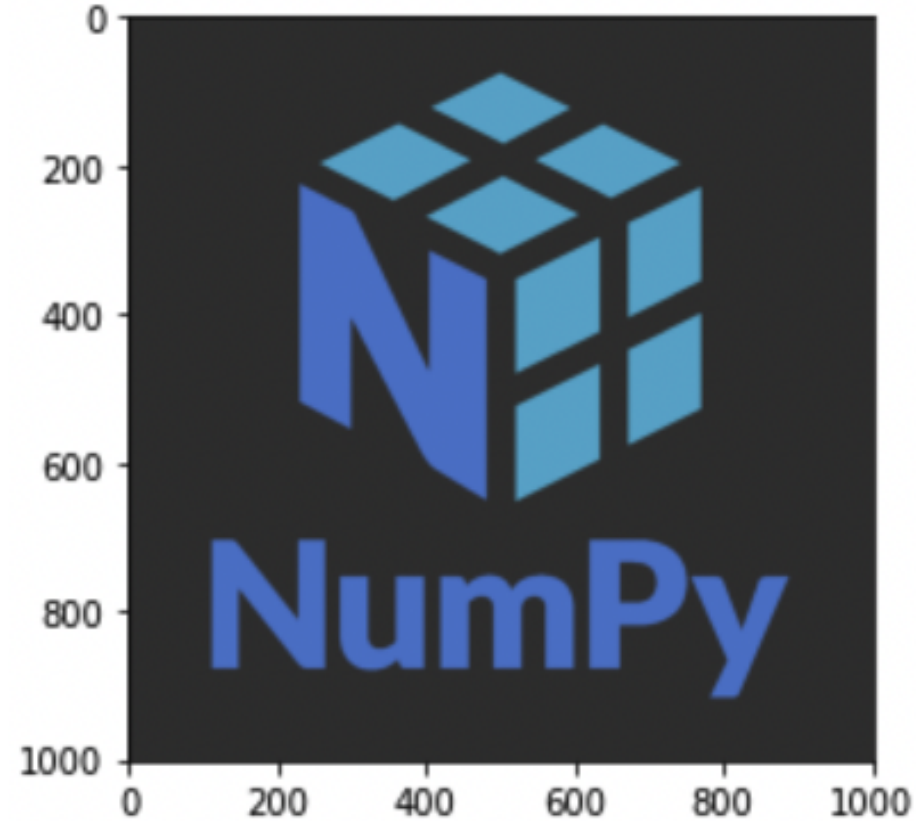
Examining RGB data

```
red_array[1], green_array[1], blue_array[1]
```

```
(array([255, 255, 255, ..., 255, 255, 255]),  
 array([255, 255, 255, ..., 255, 255, 255]),  
 array([255, 255, 255, ..., 255, 255, 255]))
```

Updating RGB data

```
dark_logo_array = np.where(logo_rgb_array == 255, 50, logo_rgb_array)
plt.imshow(dark_logo_array)
plt.show()
```



Saving arrays as .npy files

```
with open("dark_logo.npy", "wb") as f:  
    np.save(f, dark_logo_array)
```


If we need help()...

```
help(np.unique)
```

Help on function unique in module numpy:

```
unique(ar, return_index=False, return_inverse=False, return_counts=False,  
       axis=None)
```

Find the unique elements of an array.

Returns the sorted unique elements of an array. There are three optional outputs in addition to the unique elements:

- * the indices of the input array that give the unique values...

numpy.unique

`numpy.unique(ar, return_index=False, return_inverse=False,
return_counts=False, axis=None)` [\[source\]](#)

Find the unique elements of an array.

Returns the sorted unique elements of an array. There are three optional outputs in addition to the unique elements:

- the indices of the input array that give the unique values
- the indices of the unique array that reconstruct the input array
- the number of times each unique value comes up in the input array

Parameters: *ar* : *array_like*

Input array. Unless *axis* is specified, this will be flattened if it is not already 1-D.

help() with methods

```
help(np.ndarray.flatten)
```

```
Help on method_descriptor: flatten(...)
```

```
a.flatten(order='C')
```

Return a copy of the array collapsed into one dimension.

Parameters

<hr />-----

order : {'C', 'F', 'A', 'K'}, optional

 'C' means to flatten in row-major (C-style) order.

 'F' means to flatten in column-major (Fortran- ...

Let's practice!

INTRODUCTION TO NUMPY

Array acrobatics

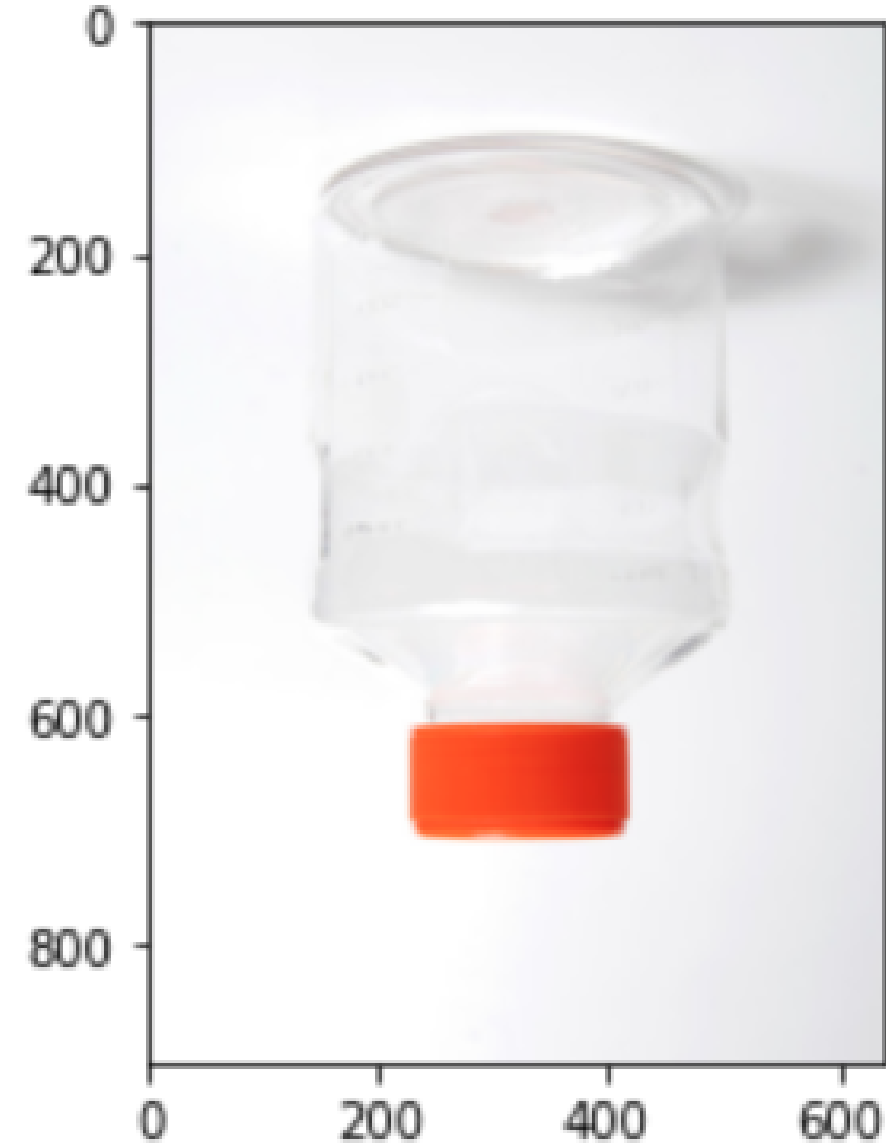
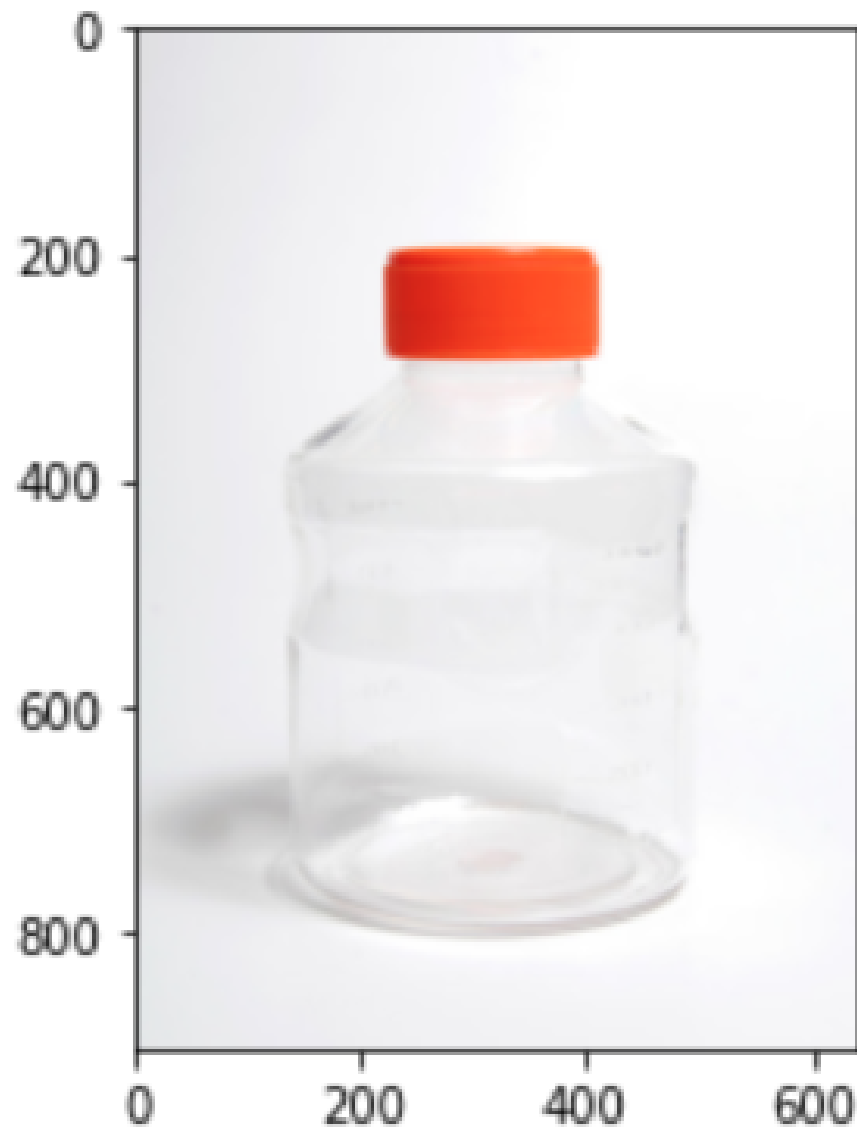
INTRODUCTION TO NUMPY



Izzy Weber

Core Curriculum Manager, DataCamp

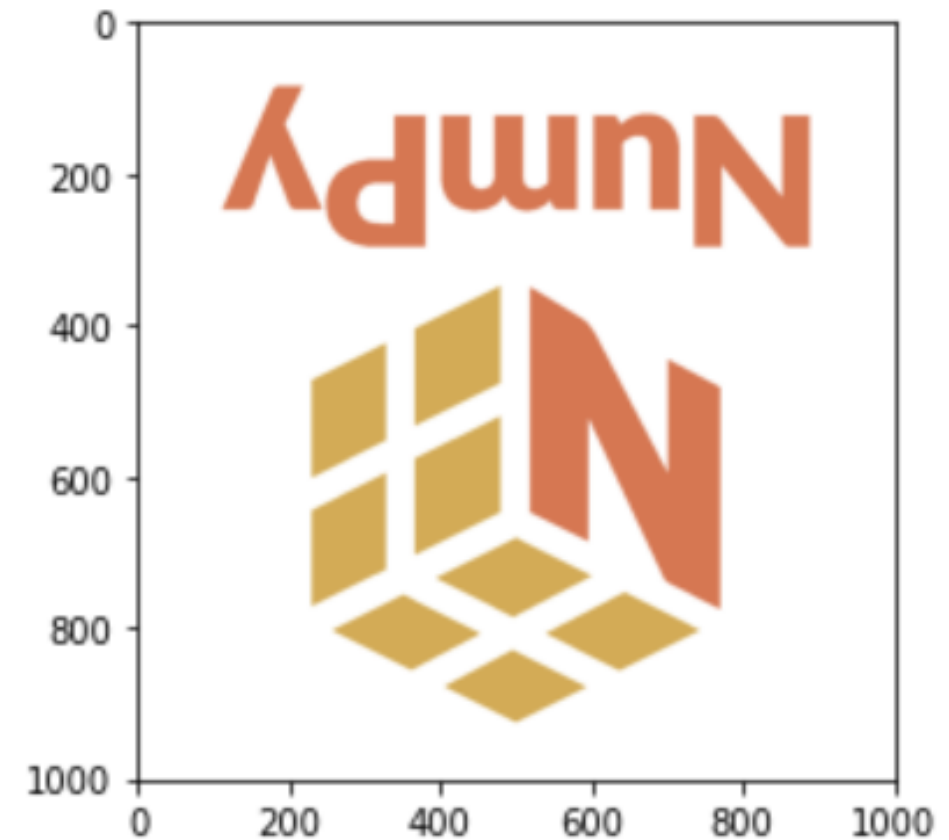
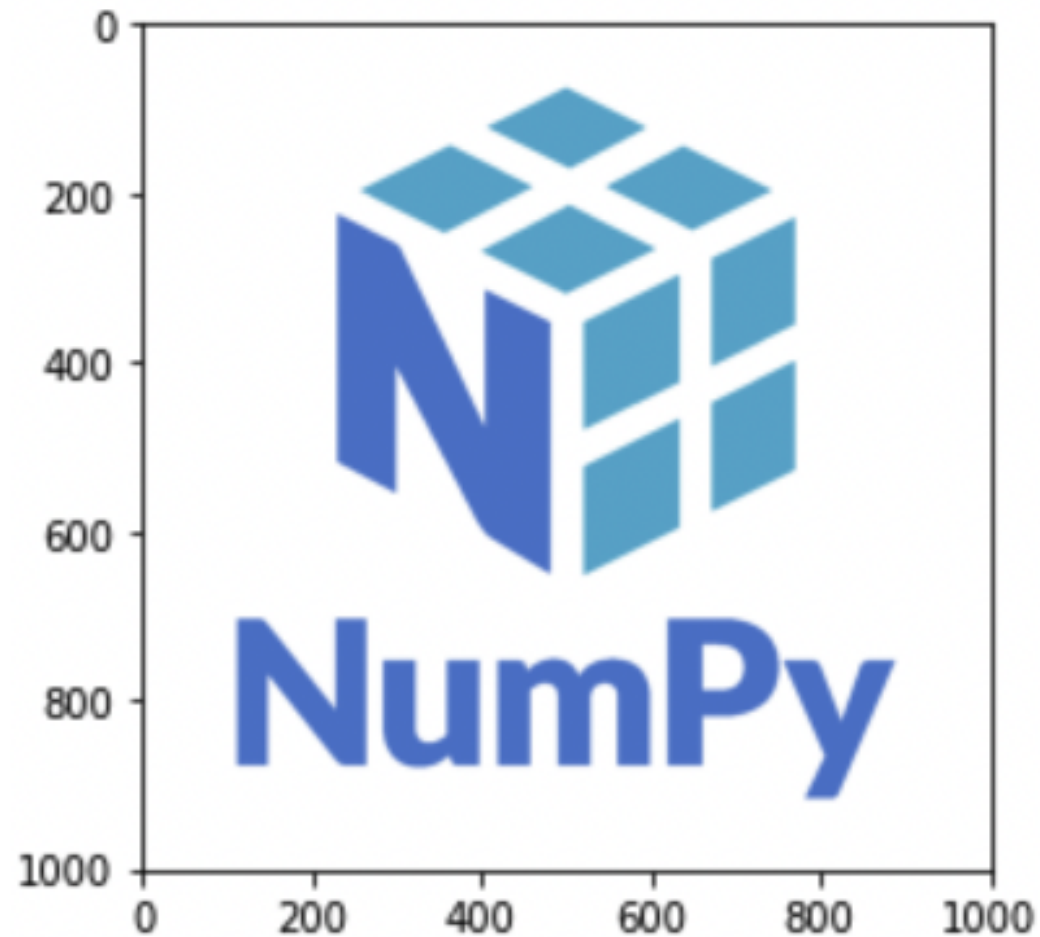
Data augmentation



¹ Plastic bottle photo by Lilly_M via Wikimedia Commons

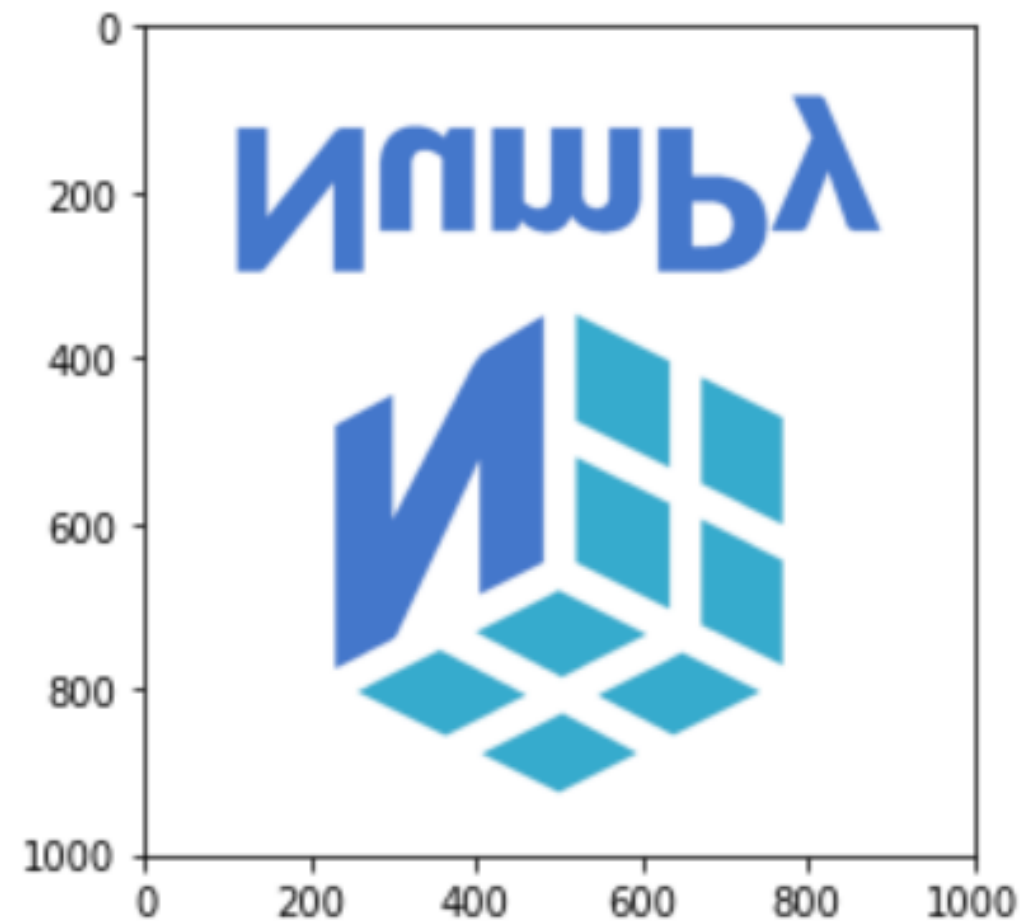
Flipping an array

```
flipped_logo = np.flip(logo_rgb_array)  
plt.imshow(flipped_logo)  
plt.show()
```



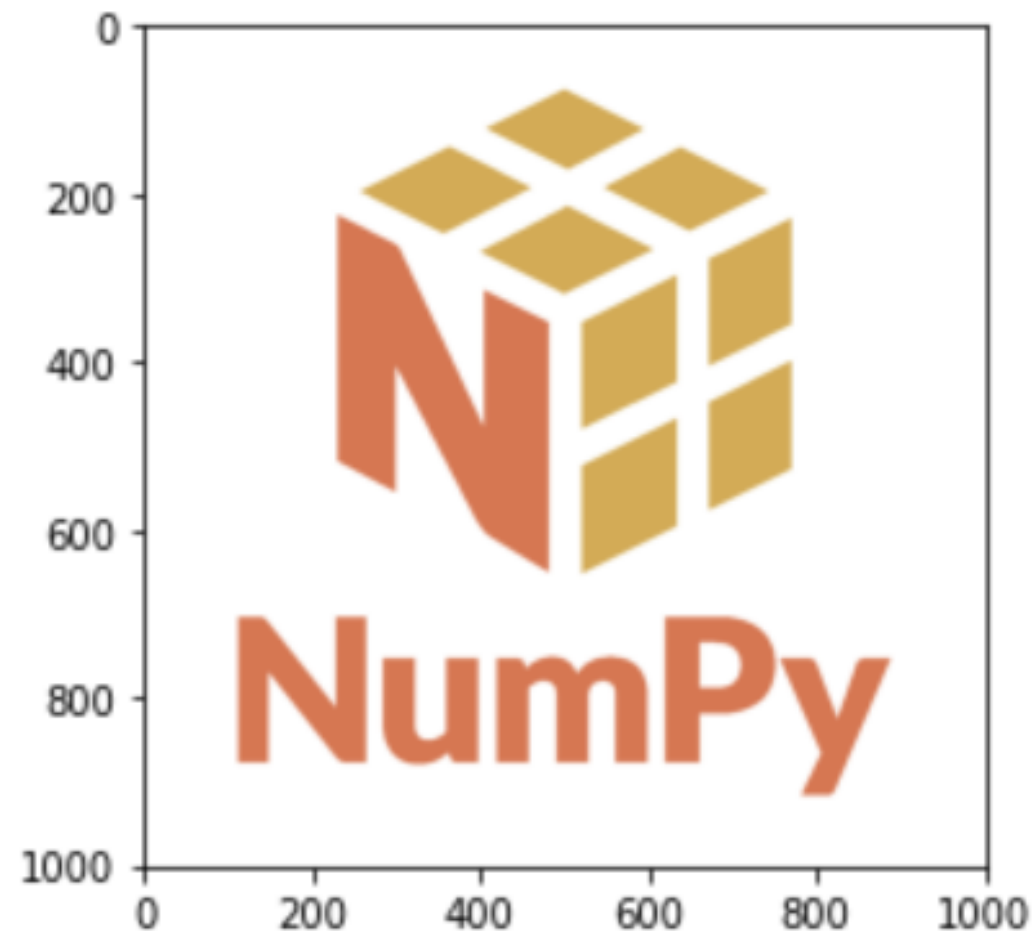
Flipping along an axis

```
flipped_rows_logo = np.flip(logo_rgb_array, axis=0)  
plt.imshow(flipped_rows_logo)  
plt.show()
```



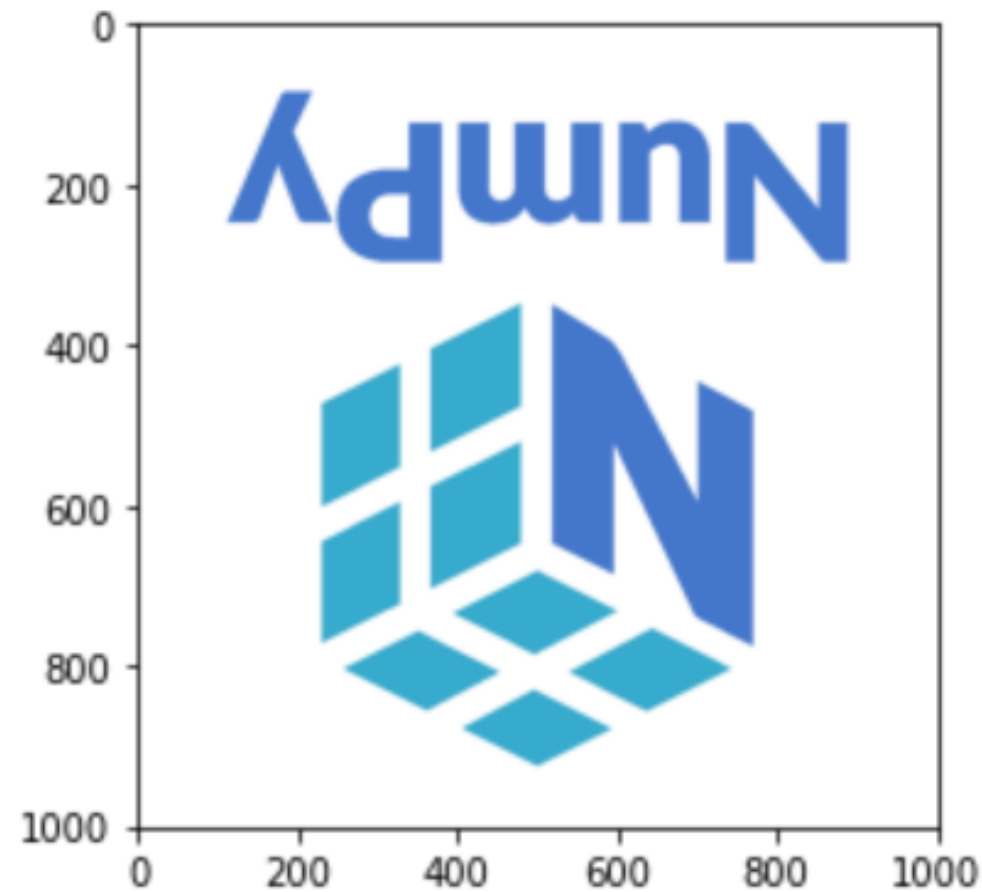
Flipping along an axis

```
flipped_colors_logo = np.flip(logo_rgb_array, axis=2)  
plt.imshow(flipped_colors_logo)  
plt.show()
```



Flipping multiple axes

```
flipped_except_colors_logo = np.flip(logo_rgb_array, axis=(0, 1))  
plt.imshow(flipped_except_colors_logo)  
plt.show()
```



Transposing an array

```
array = np.array([[1.1, 1.2, 1.3],  
                  [2.1, 2.2, 2.3],  
                  [3.1, 3.2, 3.3],  
                  [4.1, 4.2, 4.3]])  
  
np.flip(array)
```

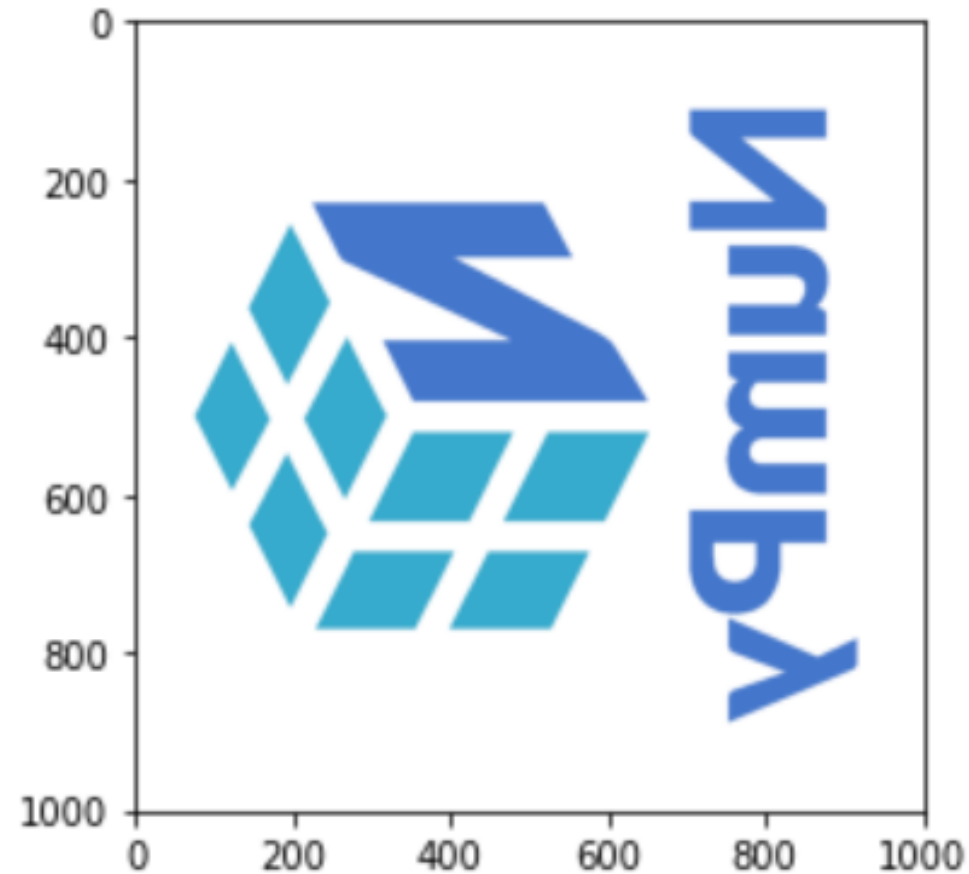
```
array([[4.3, 4.2, 4.1],  
       [3.3, 3.2, 3.1],  
       [2.3, 2.2, 2.1],  
       [1.3, 1.2, 1.1]])
```

```
array = np.array([[1.1, 1.2, 1.3],  
                  [2.1, 2.2, 2.3],  
                  [3.1, 3.2, 3.3],  
                  [4.1, 4.2, 4.3]])  
  
np.transpose(array)
```

```
array([[1.1, 2.1, 3.1, 4.1],  
       [1.2, 2.2, 3.2, 4.2],  
       [1.3, 2.3, 3.3, 4.3]])
```

Setting transposed axis order

```
transposed_logo = np.transpose(logo_rgb_array, axes=(1, 0, 2))  
plt.imshow(transposed_logo)  
plt.show()
```



Let's practice!

INTRODUCTION TO NUMPY

Stacking and splitting

INTRODUCTION TO NUMPY



Izzy Weber

Core Curriculum Manager, DataCamp

Slicing dimensions

```
rgb = np.array([[[255, 0, 0], [255, 255, 0], [255, 255, 255]],  
                [[255, 0, 255], [0, 255, 0], [0, 255, 255]],  
                [[0, 0, 0], [0, 255, 255], [0, 0, 255]]])  
  
red_array = rgb[:, :, 0]  
green_array = rgb[:, :, 1]  
blue_array = rgb[:, :, 2]  
red_array
```

```
array([[255, 255, 255],  
       [255,  0,  0],  
       [0,  0,  0]])
```

Splitting arrays

```
red_array, green_array, blue_array = np.split(rgb, 3, axis=2)  
red_array
```

```
array([[[255], [255], [255]],  
       [[255], [ 0], [ 0]],  
       [[ 0], [ 0], [ 0]]])
```

```
red_array.shape
```

```
(3, 3, 1)
```


Trailing dimensions

```
red_array_2D = red_array.reshape((3, 3))  
red_array_2D
```

```
array([[255, 255, 255],  
       [255,   0,   0],  
       [  0,   0,   0]])
```

```
red_array_2D.shape
```

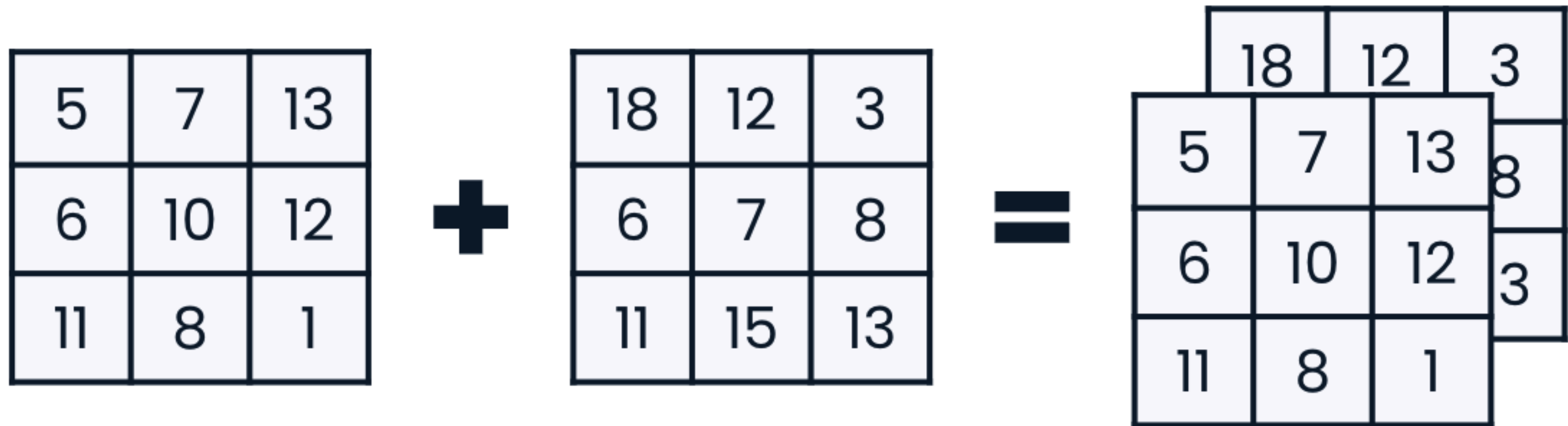
```
(3, 3)
```

Array division rules

```
red_array, green_array, blue_array = np.split(rgb, 5, axis=2)
```

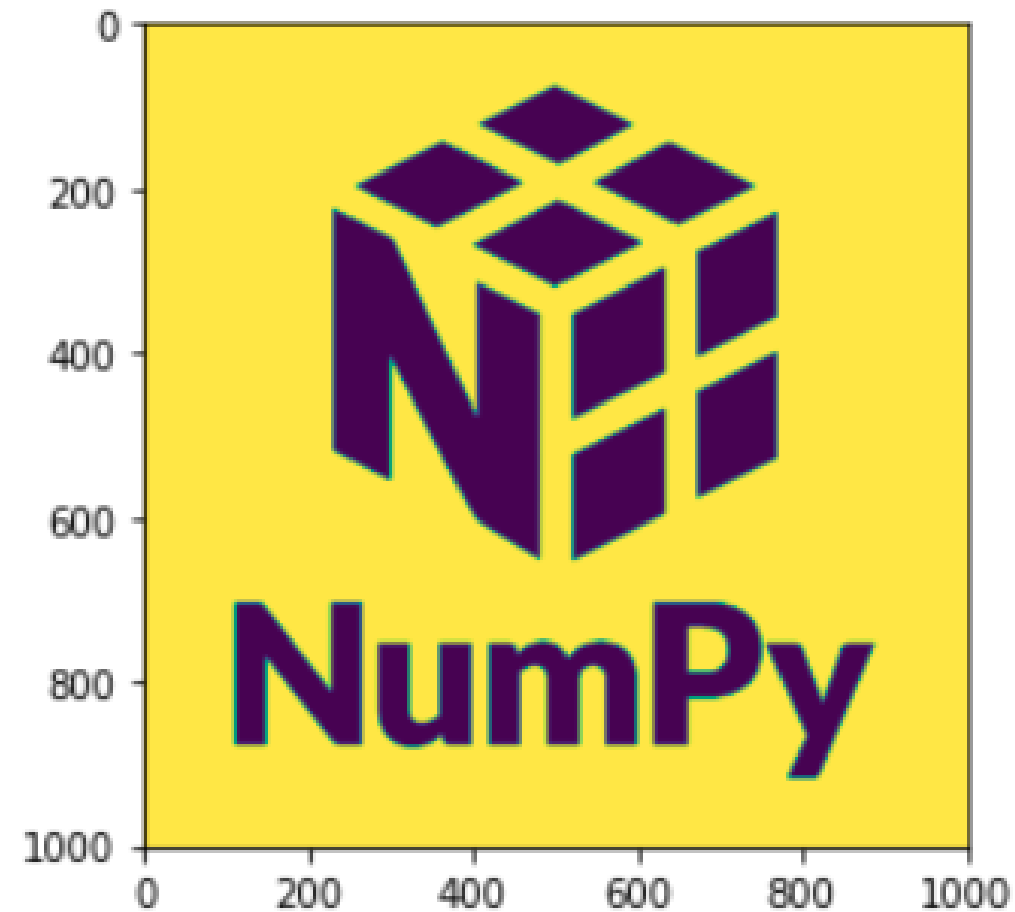
```
ValueError: array split does not result in an equal division
```

Stacking arrays



Plotting 2D image data

```
red_array, green_array, blue_array = np.split(logo_rgb_array, 3, axis=2)  
plt.imshow(red_array)  
plt.show()
```

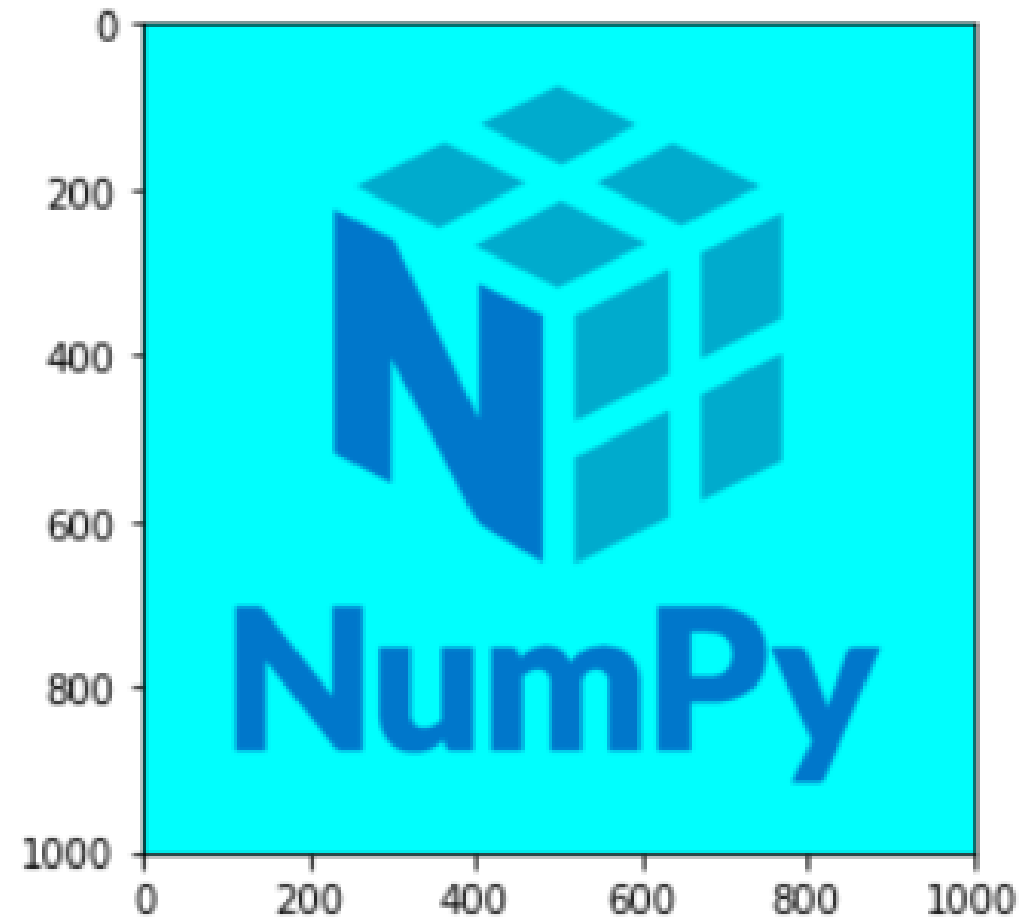


Stacking 2D arrays

```
red_array = np.zeros((1001, 1001)).astype(np.int32)  
green_array = green_array.reshape((1001, 1001))  
blue_array = blue_array.reshape((1001, 1001))
```

Stacking 2D arrays

```
stacked_rgb = np.stack([red_array, green_array, blue_array], axis=2)  
plt.imshow(stacked_rgb)  
plt.show()
```



Let's practice!

INTRODUCTION TO NUMPY

Congratulations!

INTRODUCTION TO NUMPY



Izzy Weber

Core Curriculum Manager, DataCamp

Welcome to team NumPy



NumPy is amazing!



What's next?



DataCamp courses:

- Data Manipulation with pandas
- Introduction to Data Visualization with Seaborn
- Parallel Programming with Dask in Python
- Introduction to TensorFlow in Python

$\begin{bmatrix} \begin{bmatrix} \text{'T'} & \text{'H'} & \text{'A'} & \text{'N'} & \text{'K'} & \text{'S'} & \text{' '} & \text{'\&'} \end{bmatrix} \\ \begin{bmatrix} \text{'C'} & \text{'O'} & \text{'N'} & \text{'G'} & \text{'R'} & \text{'A'} & \text{'T'} & \text{'S'} \end{bmatrix} \end{bmatrix}$

Thank you!
INTRODUCTION TO NUMPY