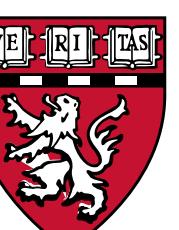


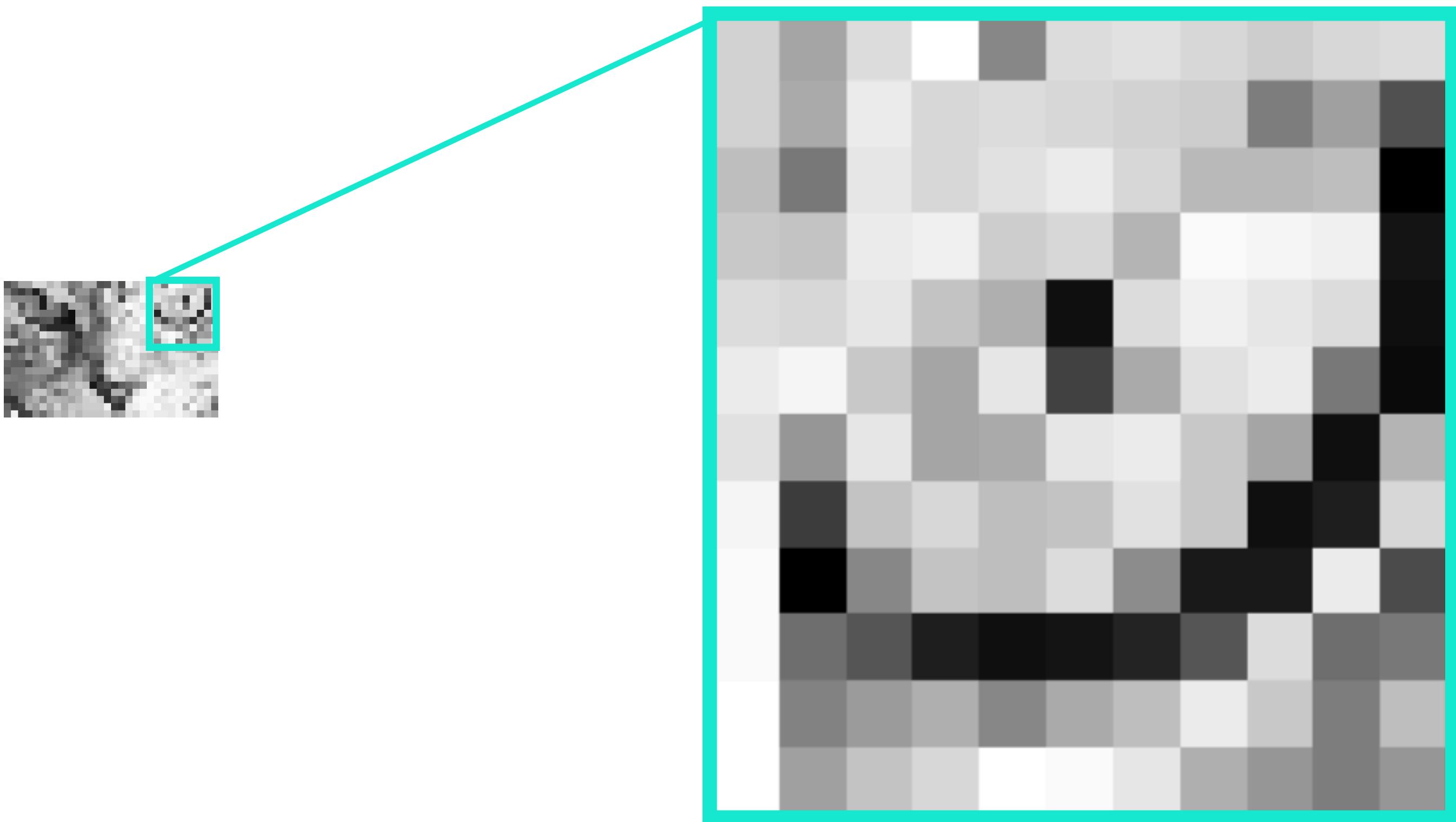


Python for bioimage analysis





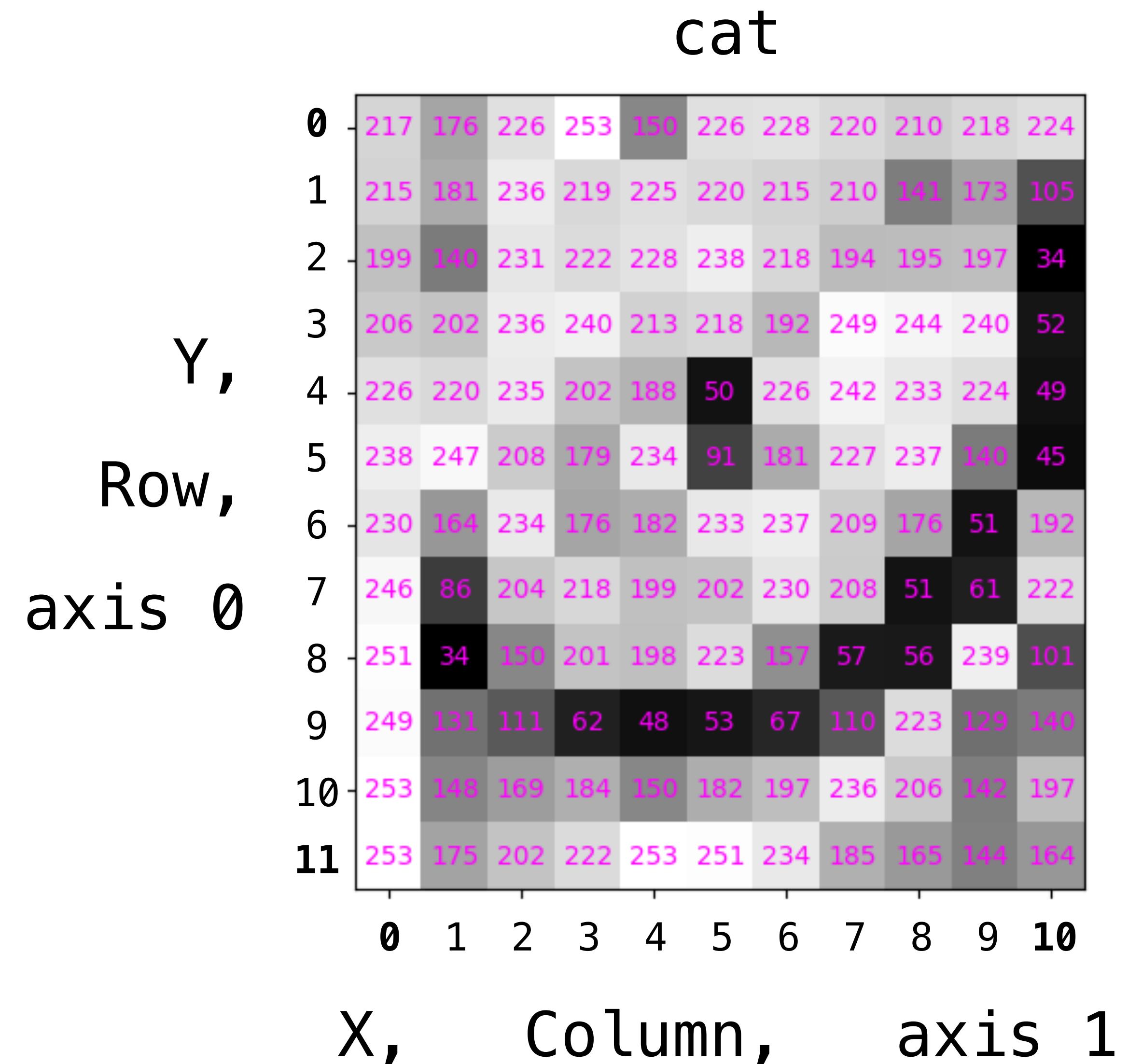
The data





The data

Type of the image: <class 'numpy.ndarray'>
Datatype of the image: uint8
Shape of the image: (12, 11)
Minimum pixel value: 34
Maximum pixel value: 253
Mean pixel value: 184.17





Plot a Histogram

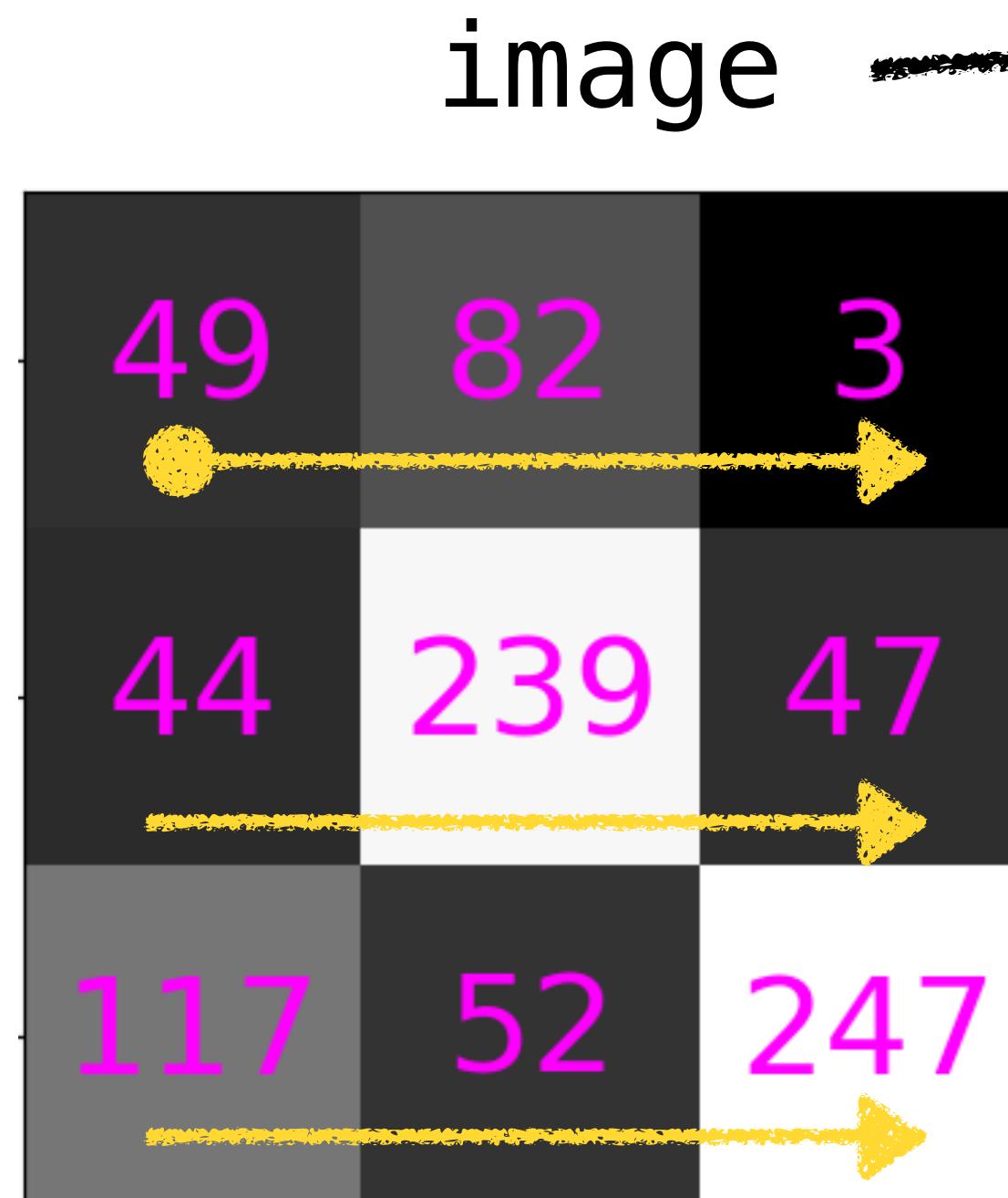
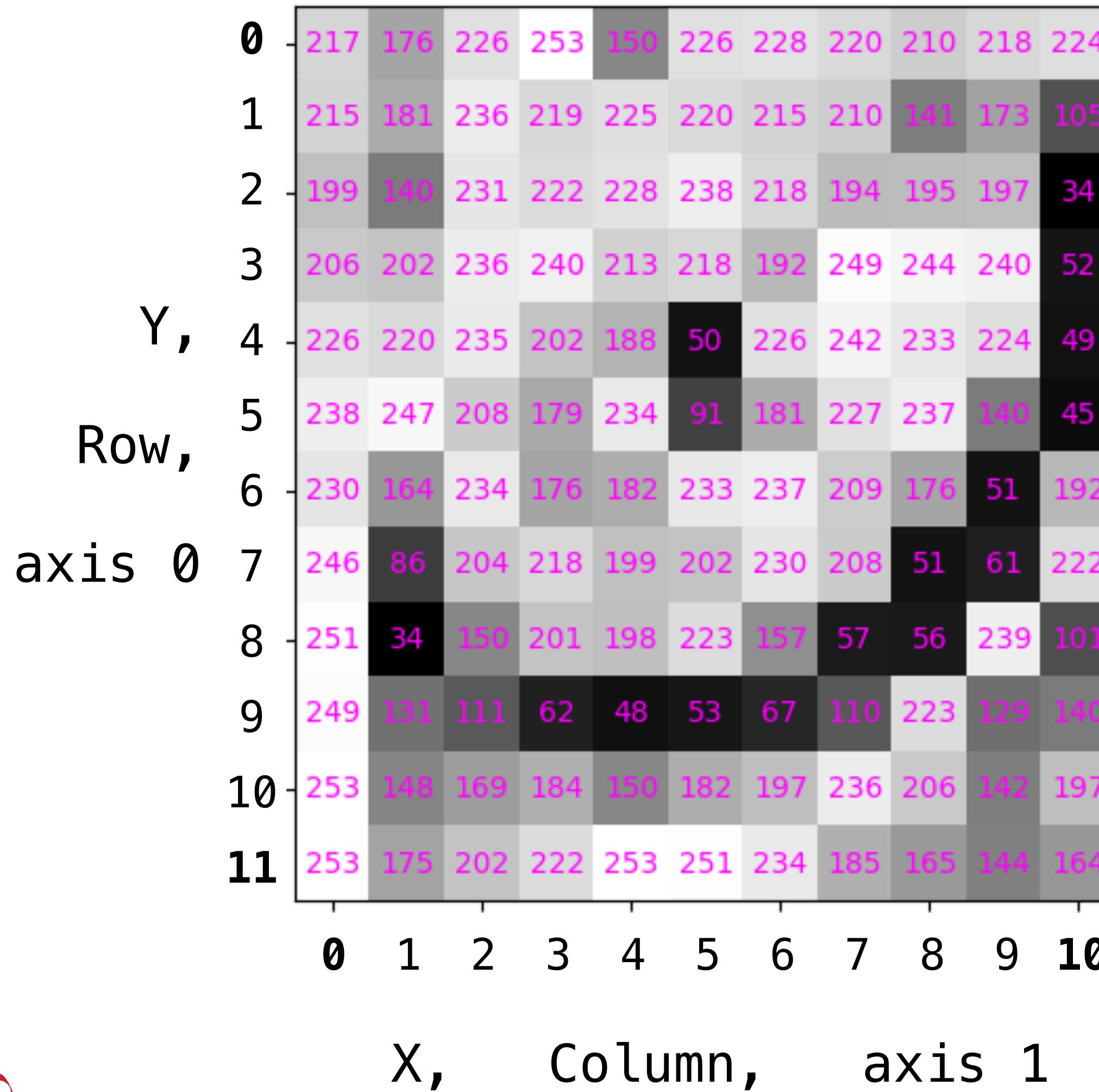


image.ravel()





Indexing: individual entries



Row Column
axis 0 axis 1
↓ ↓
cat [, ])





Indexing: individual entries

0	217	176	226	253	150	226	228	220	210	218	224
1	215	181	236	219	225	220	215	210	141	173	105
2	199	140	231	222	228	238	218	194	195	197	34
3	206	202	236	240	213	218	192	249	244	240	52
4	226	220	235	202	188	50	226	242	233	224	49
5	238	247	208	179	234	91	181	227	237	140	45
6	230	164	234	176	182	233	237	209	176	51	192
7	246	86	204	218	199	202	230	208	51	61	222
8	251	34	150	201	198	223	157	57	56	239	101
9	249	131	111	62	48	53	67	110	223	129	140
10	253	148	169	184	150	182	197	236	206	142	197
11	253	175	202	222	253	251	234	185	165	144	164

X, Column, axis 1

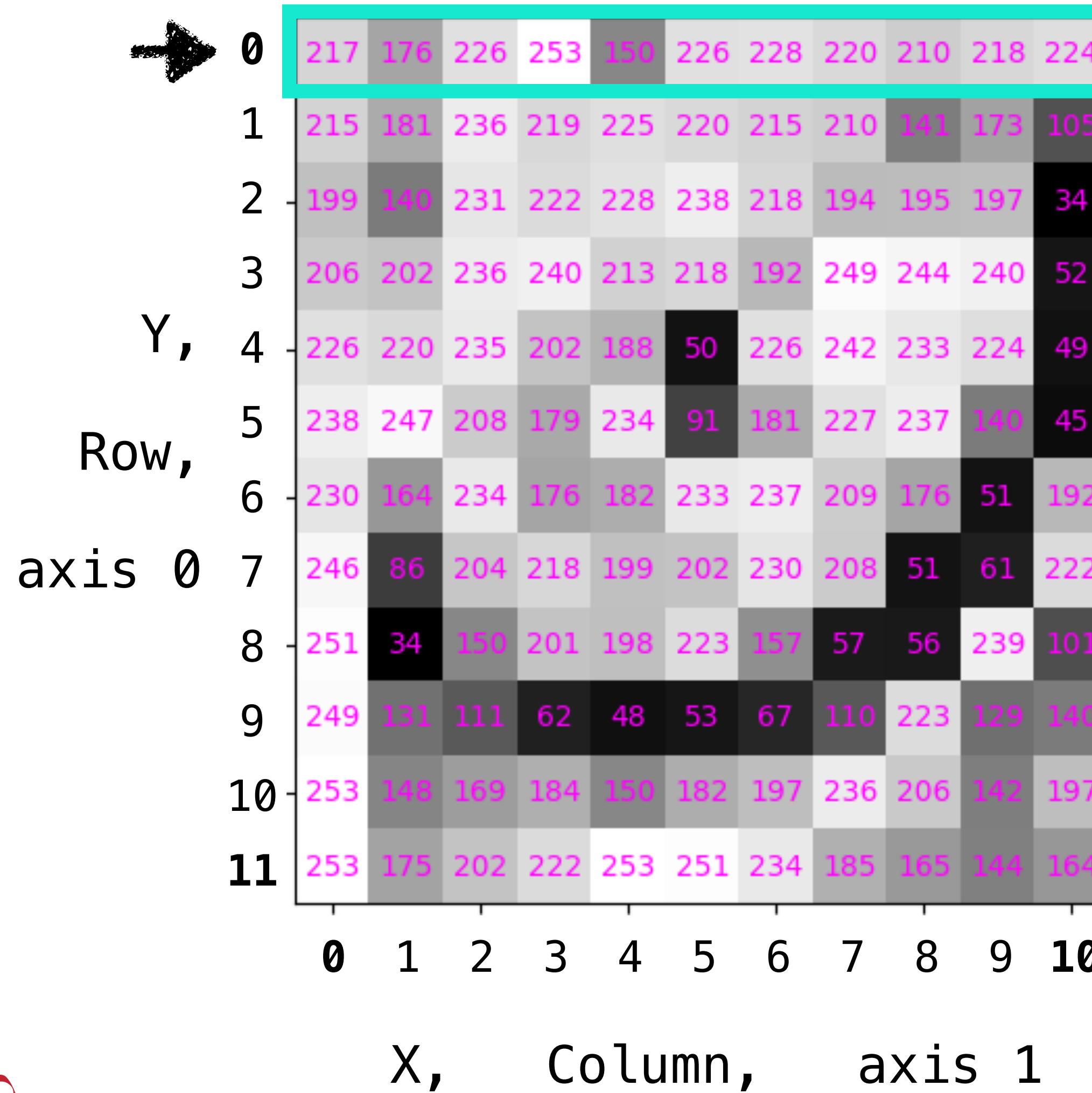
0th row 1st column

```
print(cat[0, 1])
```





Indexing: individual entries



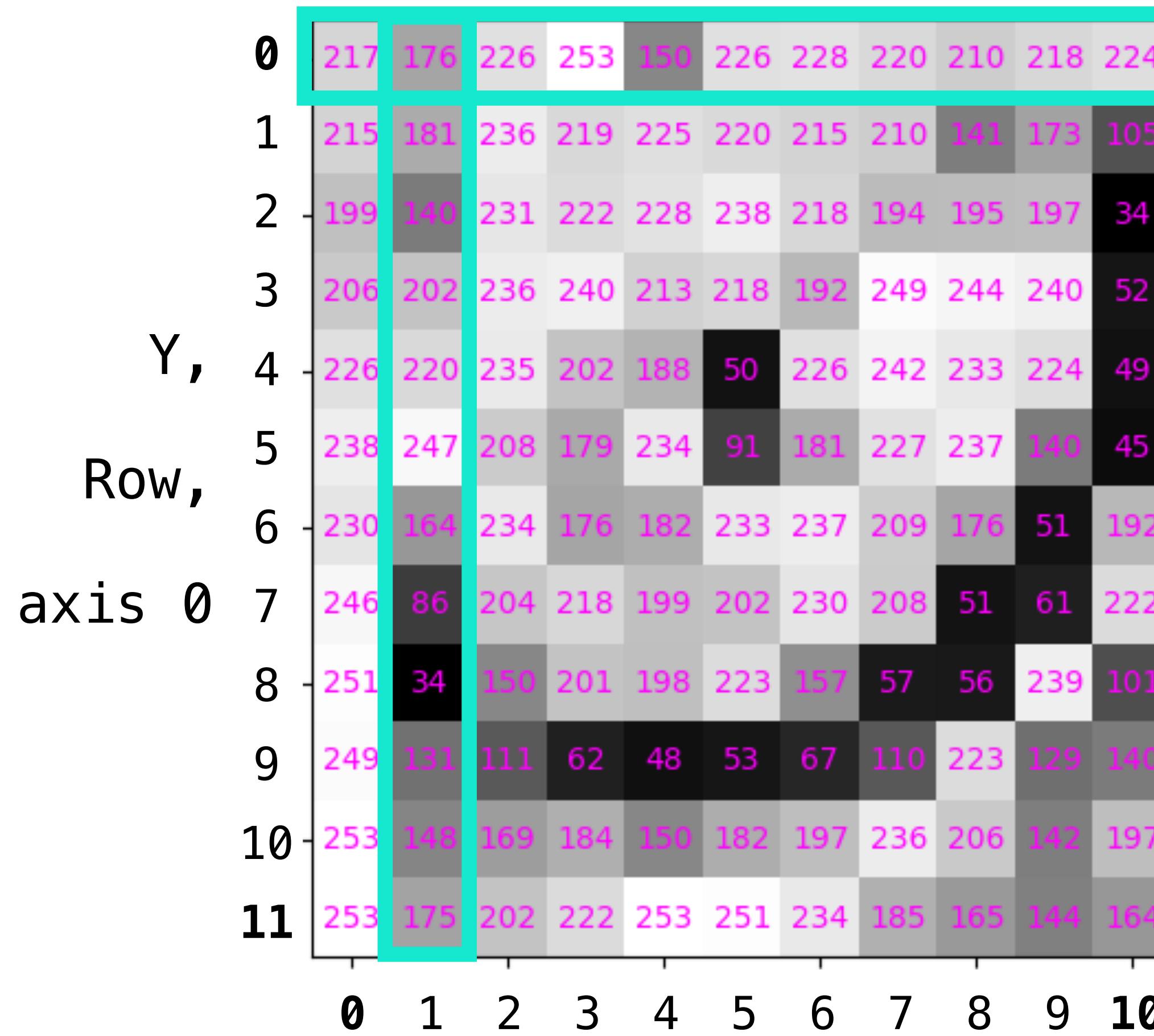
0th row 1st column

```
print(cat[0, 1])
```

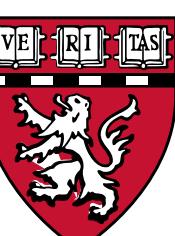
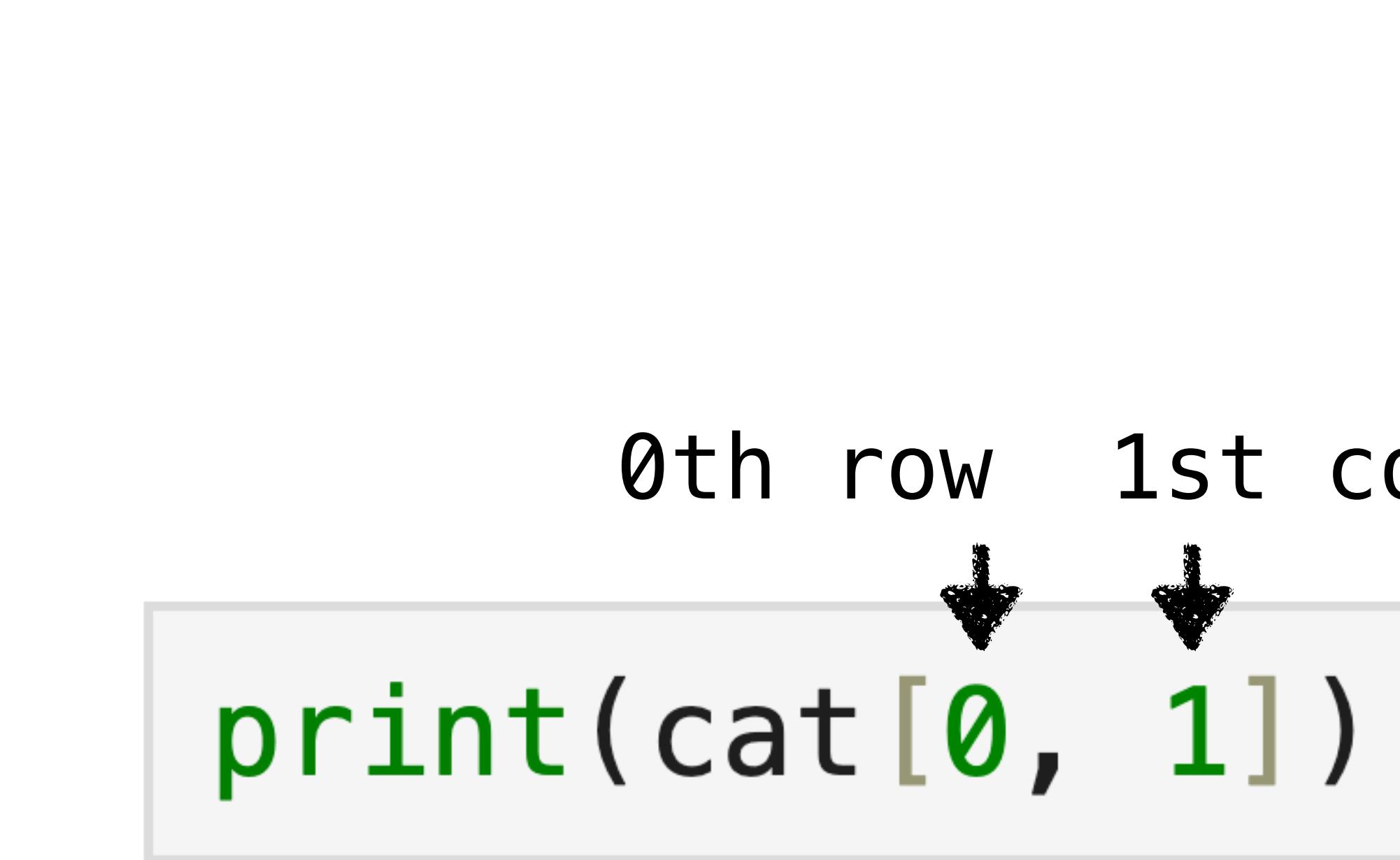




Indexing: individual entries

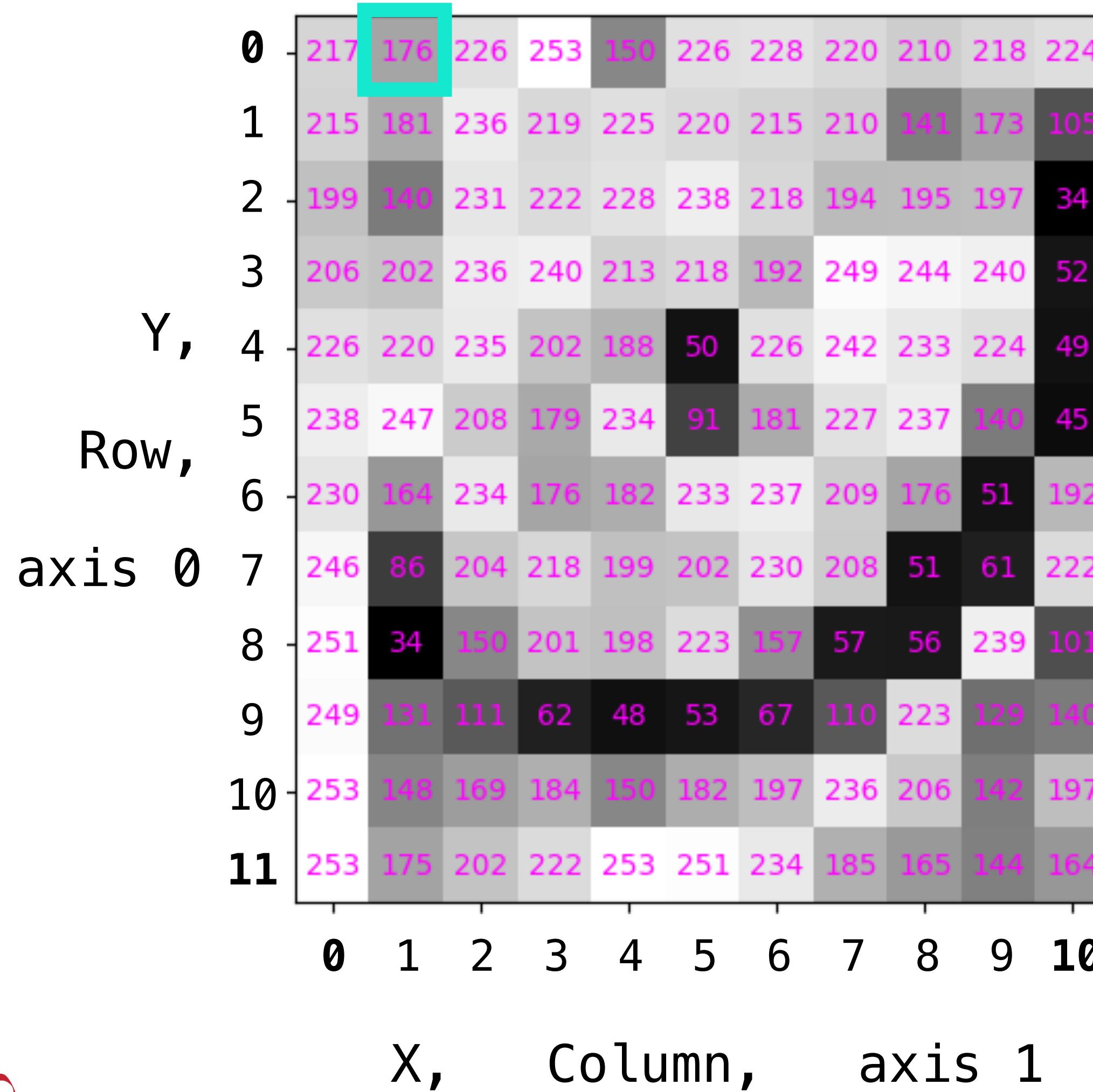


X, Column, axis 1
↑





Indexing: individual entries



0th row 1st column

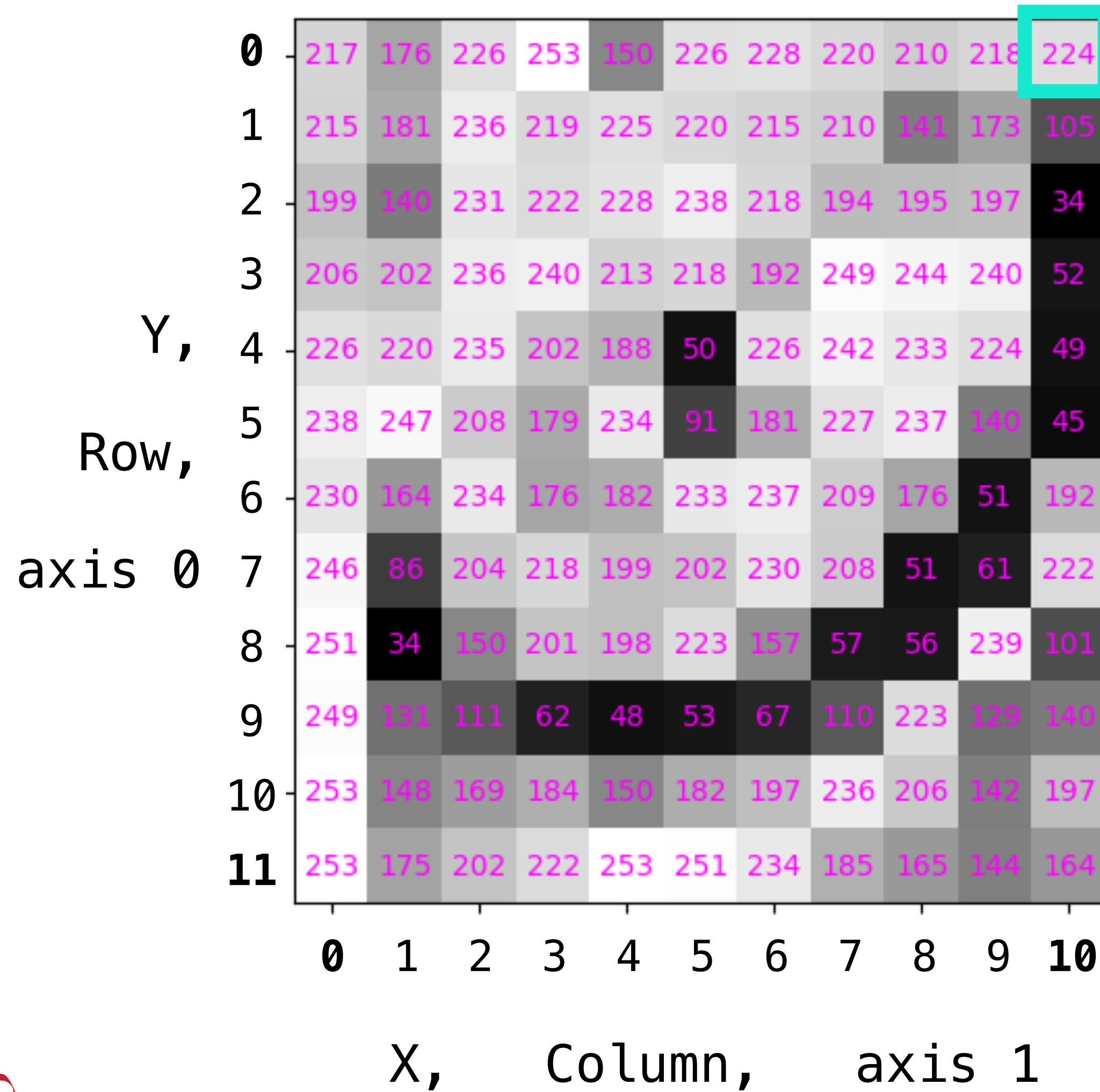
```
print(cat[0, 1])
```

176



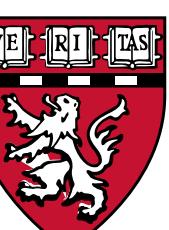


Indexing: individual entries



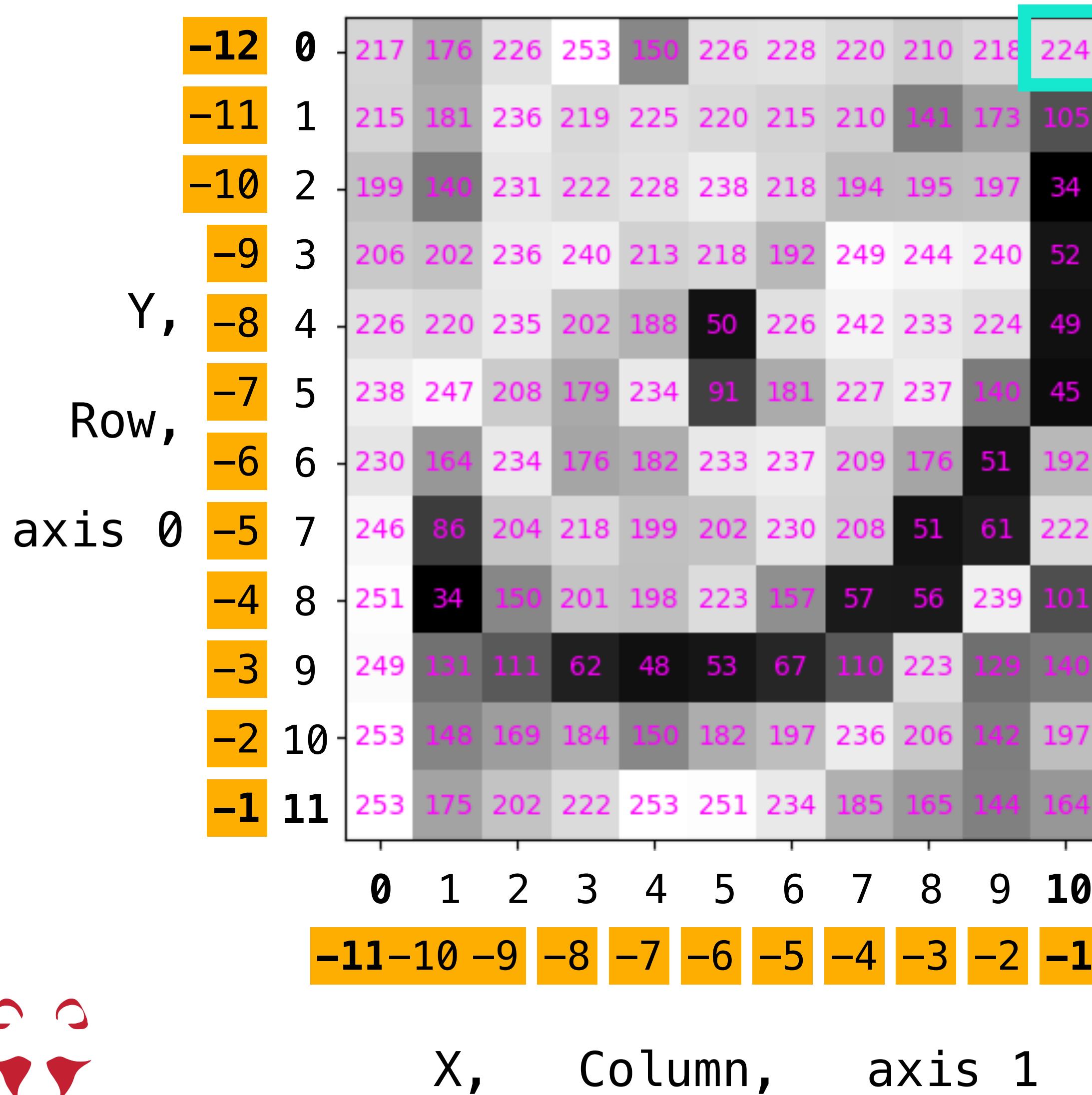
```
print(cat[0, 10])
```

224





Indexing: individual entries



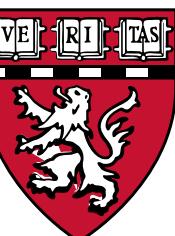
```
print(cat[0, 10])
```

224

Negative indexing

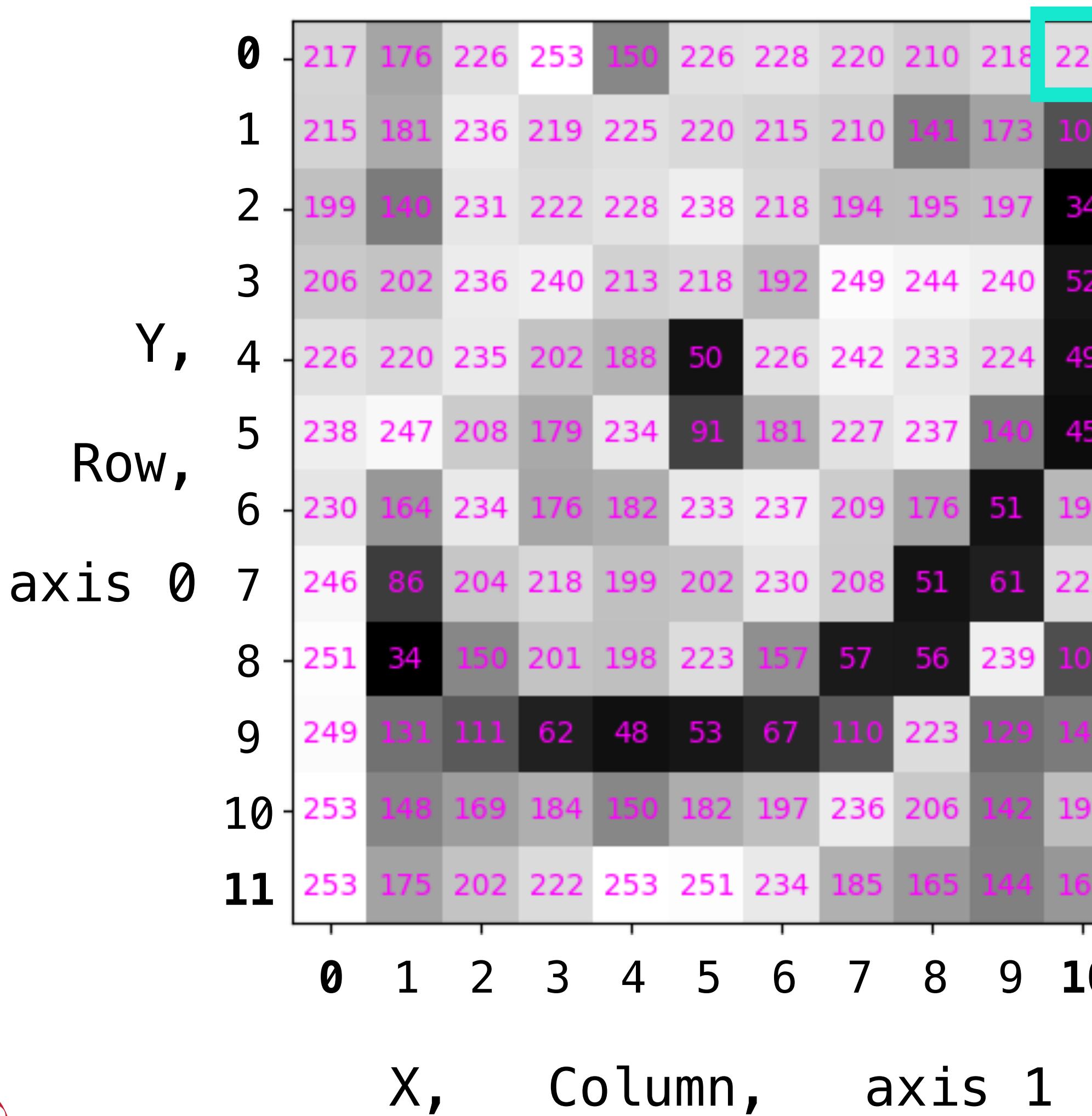
```
print(cat[0, -1])
```

224





Indexing: individual entries

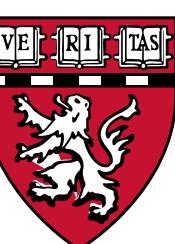


Exercise:

Explore indexing individual entries

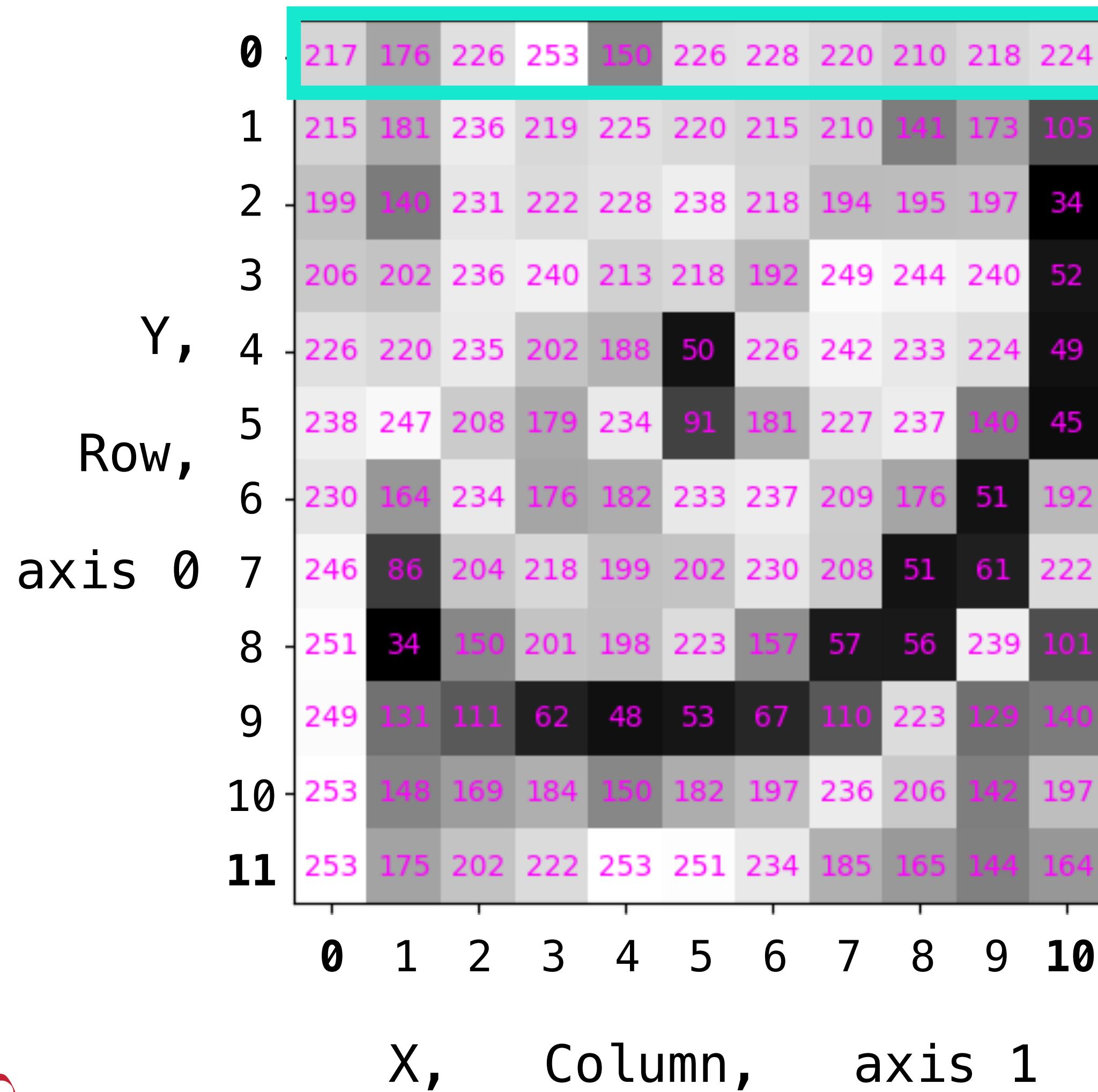


```
row, col = 0, 10  
valueplot(cat, indices=str([row, col]))
```





Indexing: rows



0th row all column

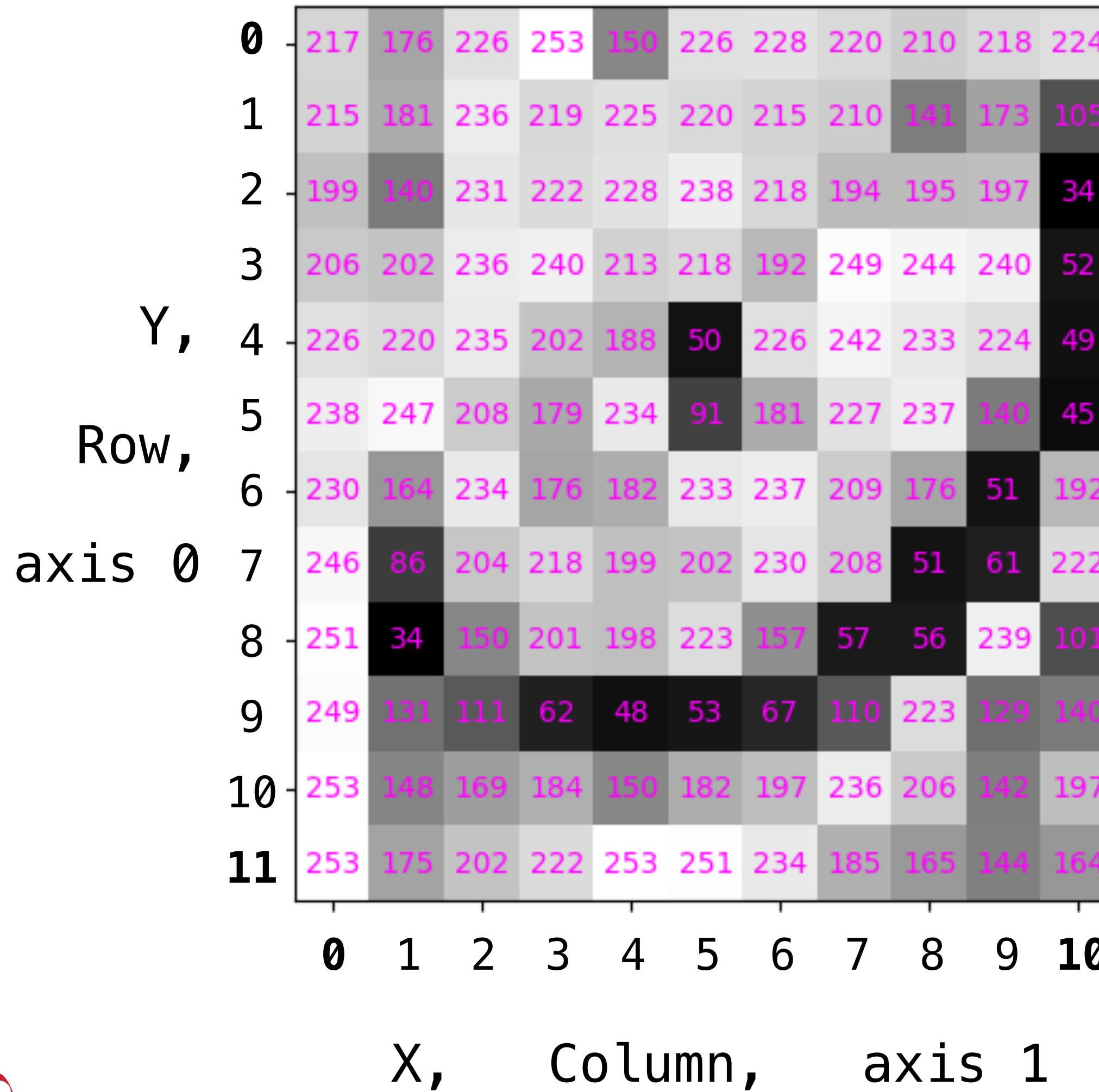
```
print(cat[0, :])
```

[217 176 226 253 150 226 228 220 210 218 224]





Indexing: rows



```
row = 1  
print(cat[row, :])  
print(cat[row, ])  
print(cat[row]) ← axis 0
```

✓ 0.0s

[215 181 236 219 225 220 215 210 141 173 105]

[215 181 236 219 225 220 215 210 141 173 105]

[215 181 236 219 225 220 215 210 141 173 105]





Indexing: rows

Y, Row, axis 0	0	1	2	3	4	5	6	7	8	9	10
	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
-12	217	176	226	253	150	226	228	220	210	218	224
-11	215	181	236	219	225	220	215	210	141	173	105
-10	199	140	231	222	228	238	218	194	195	197	34
-9	206	202	236	240	213	218	192	249	244	240	52
-8	226	220	235	202	188	50	226	242	233	224	49
-7	238	247	208	179	234	91	181	227	237	140	45
-6	230	164	234	176	182	233	237	209	176	51	192
-5	246	86	204	218	199	202	230	208	51	61	222
-4	251	34	150	201	198	223	157	57	56	239	101
-3	249	131	111	62	48	53	67	110	223	129	140
-2	253	148	169	184	150	182	197	236	206	142	197
-1	253	175	202	222	253	251	234	185	165	144	164

```
row = -1  
print(cat[row, :])  
print(cat[row, ])  
print(cat[row])
```

✓ 0.0s

```
[253 175 202 222 253 251 234 185 165 144 164]
```

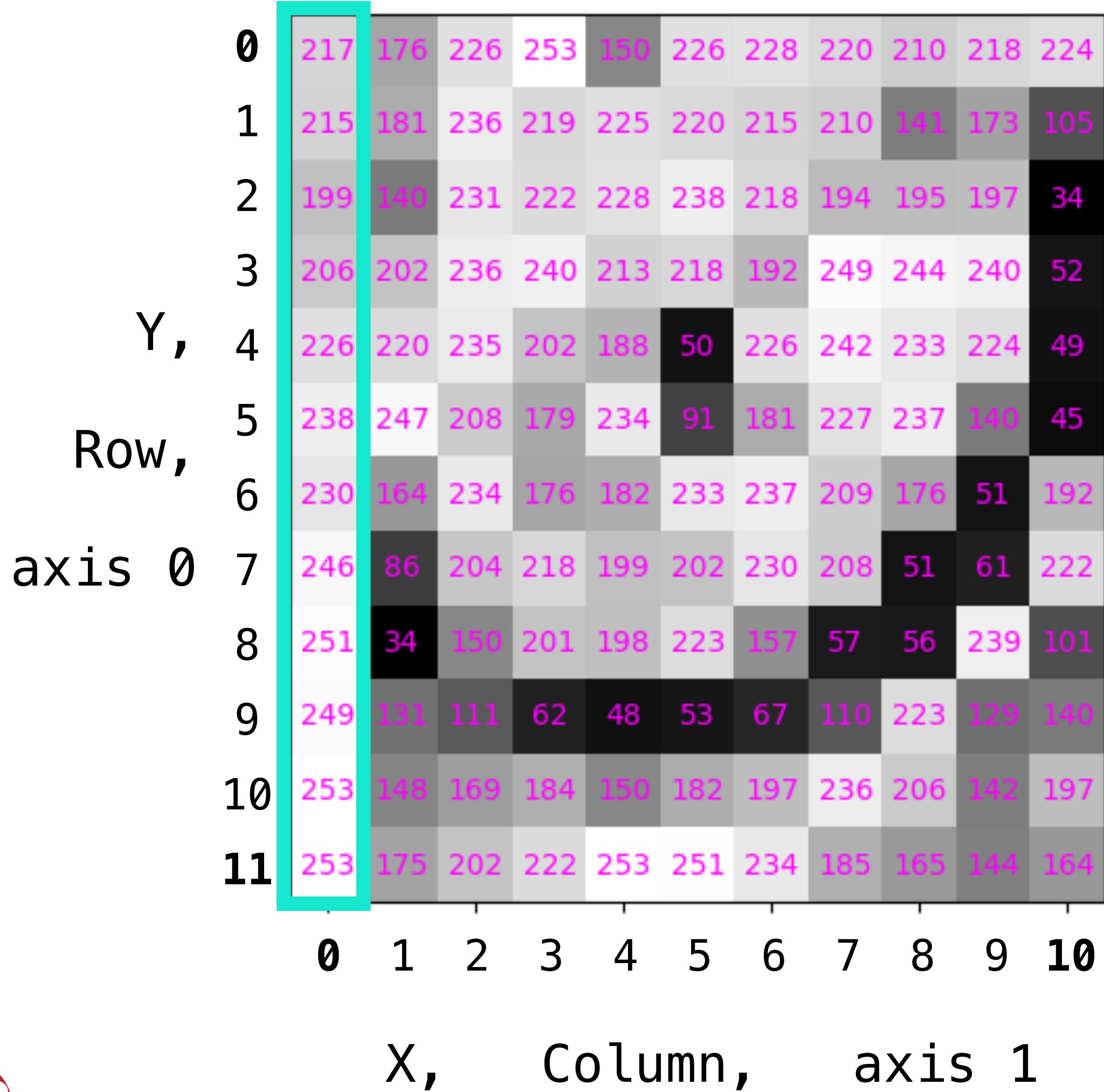
```
[253 175 202 222 253 251 234 185 165 144 164]
```

```
[253 175 202 222 253 251 234 185 165 144 164]
```





Indexing: columns



all rows column 0



```
print(cat[:, 0])
```

```
[217 215 199 206 226 238 230 246 251 249 253 253]
```





Indexing: columns

Y, Row, axis 0	0	1	2	3	4	5	6	7	8	9	10
0	217	176	226	253	150	226	228	220	210	218	224
1	215	181	236	219	225	220	215	210	141	173	105
2	199	140	231	222	228	238	218	194	195	197	34
3	206	202	236	240	213	218	192	249	244	240	52
4	226	220	235	202	188	50	226	242	233	224	49
5	238	247	208	179	234	91	181	227	237	140	45
6	230	164	234	176	182	233	237	209	176	51	192
7	246	86	204	218	199	202	230	208	51	61	222
8	251	34	150	201	198	223	157	57	56	239	101
9	249	131	111	62	48	53	67	110	223	129	140
10	253	148	169	184	150	182	197	236	206	142	197
11	253	175	202	222	253	251	234	185	165	144	164

axis 0 axis 1



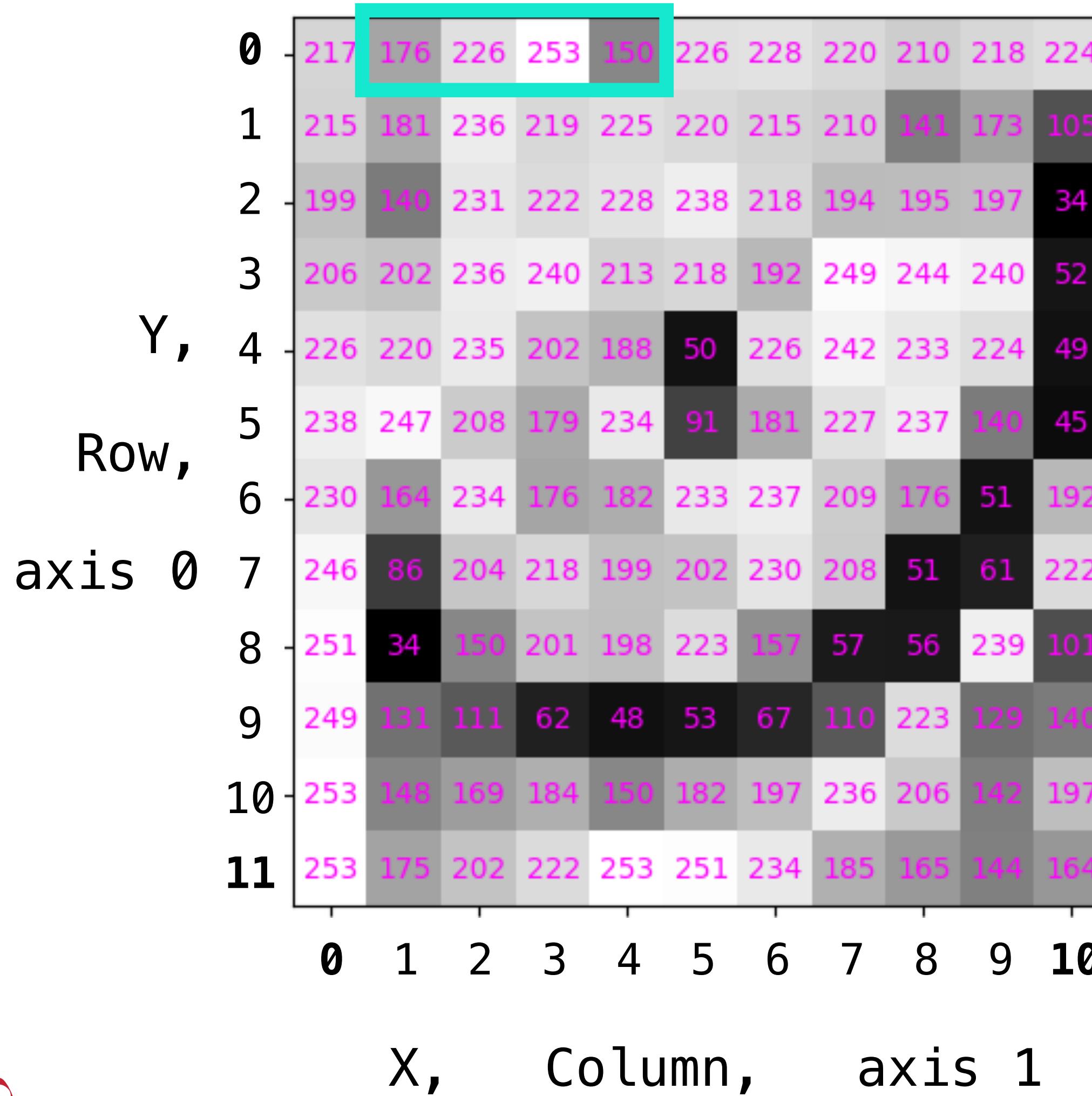
```
print(cat[, 0])
```

-> Exercises





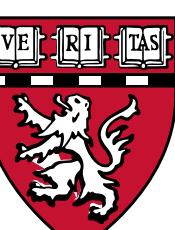
Indexing: rows and columns



“Column 1 (inclusive)
to 5 (exclusive)”

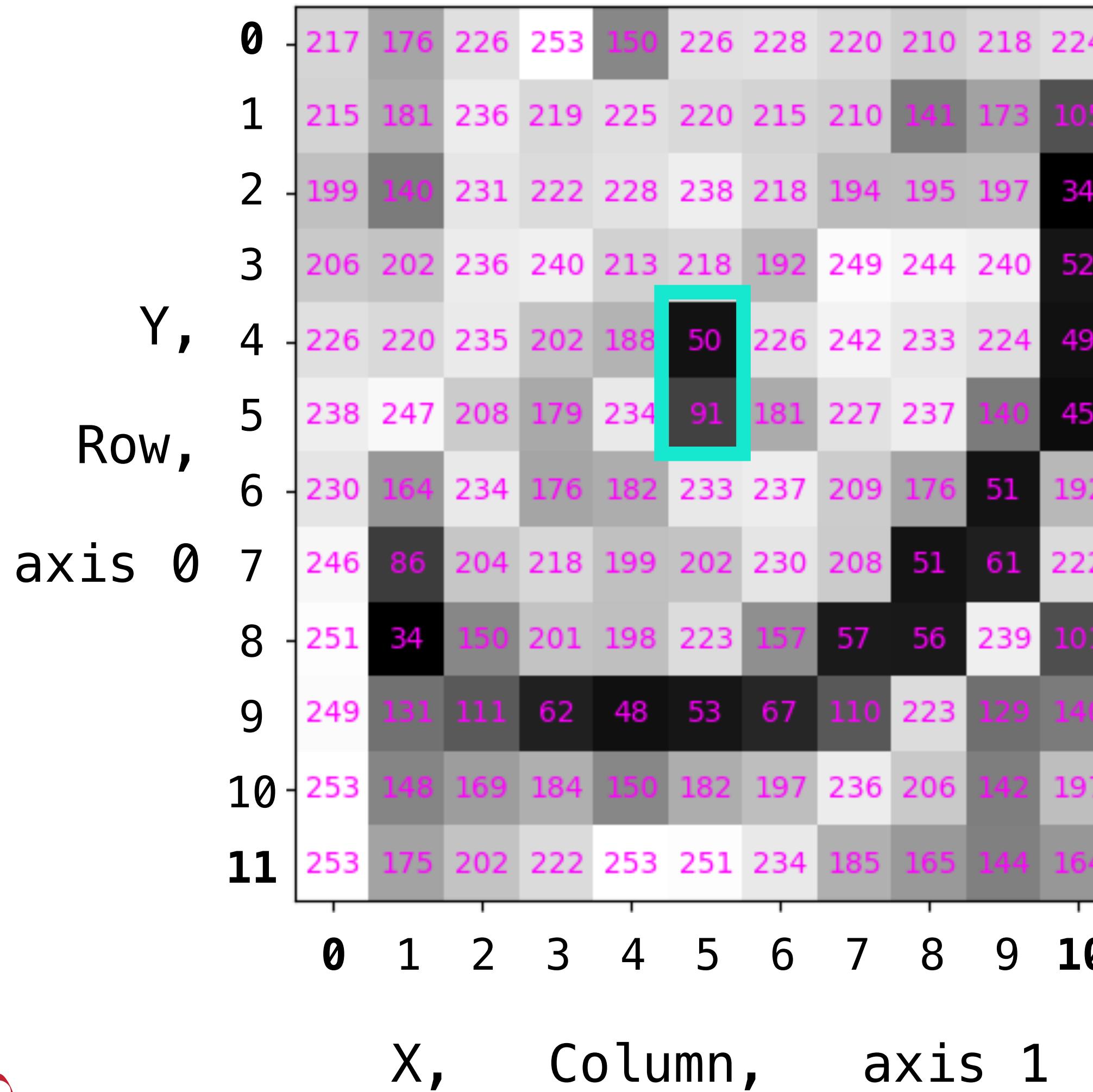
`print(cat[0, 1:5])`

[176 226 253 150]



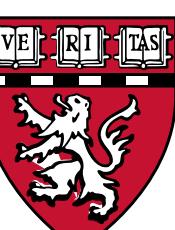


Indexing: rows and columns



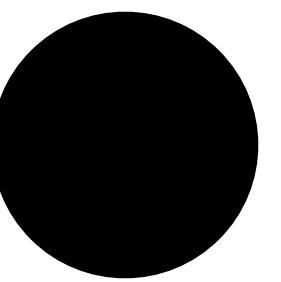
Exercise:

highlight these values using function
valueplot()





array_a





array_b
= 
array_a

array_b = array_a





```
array_b  
= array_a
```

```
array_b  
array_a
```

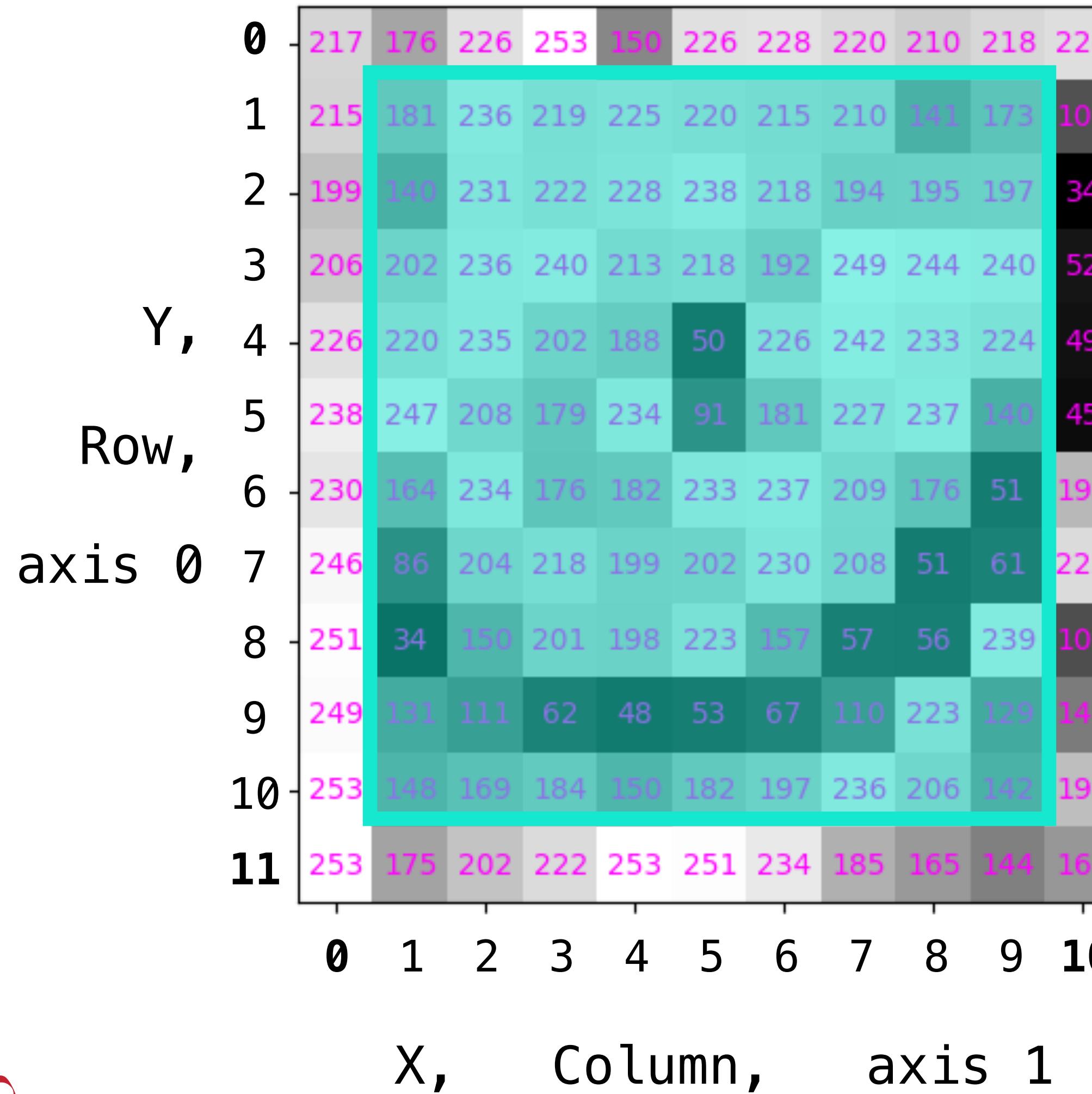
```
array_b = array_a
```

```
array_b = array_a.copy()
```





Indexing: rows and columns

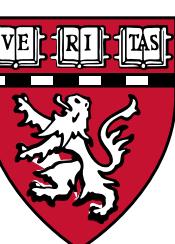


Exercise:

Make a copy of cat and name it "pirate".

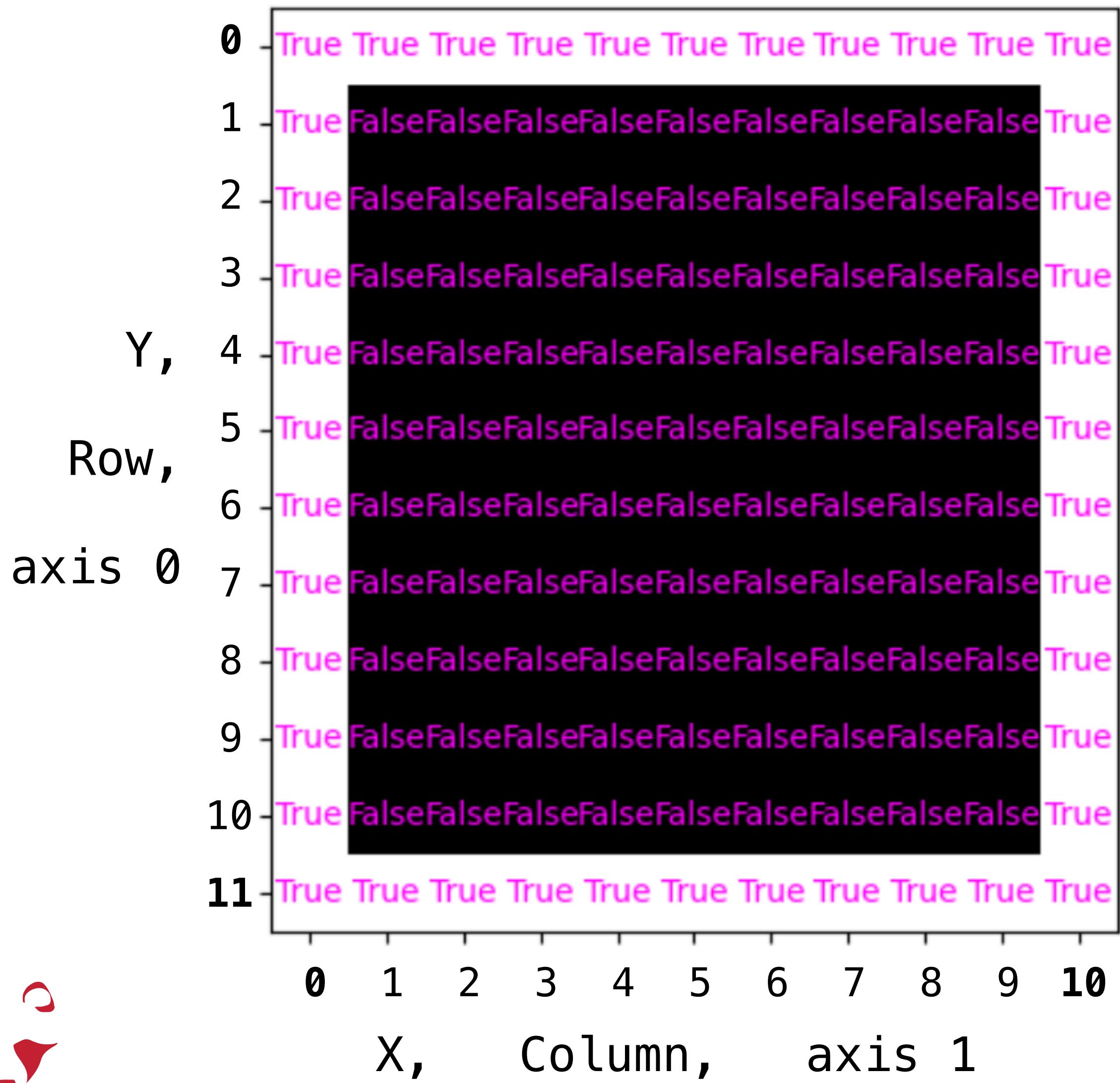
Assign to all pixels but the rim-pixels a value of 0.

Plot to verify





Indexing: Boolean



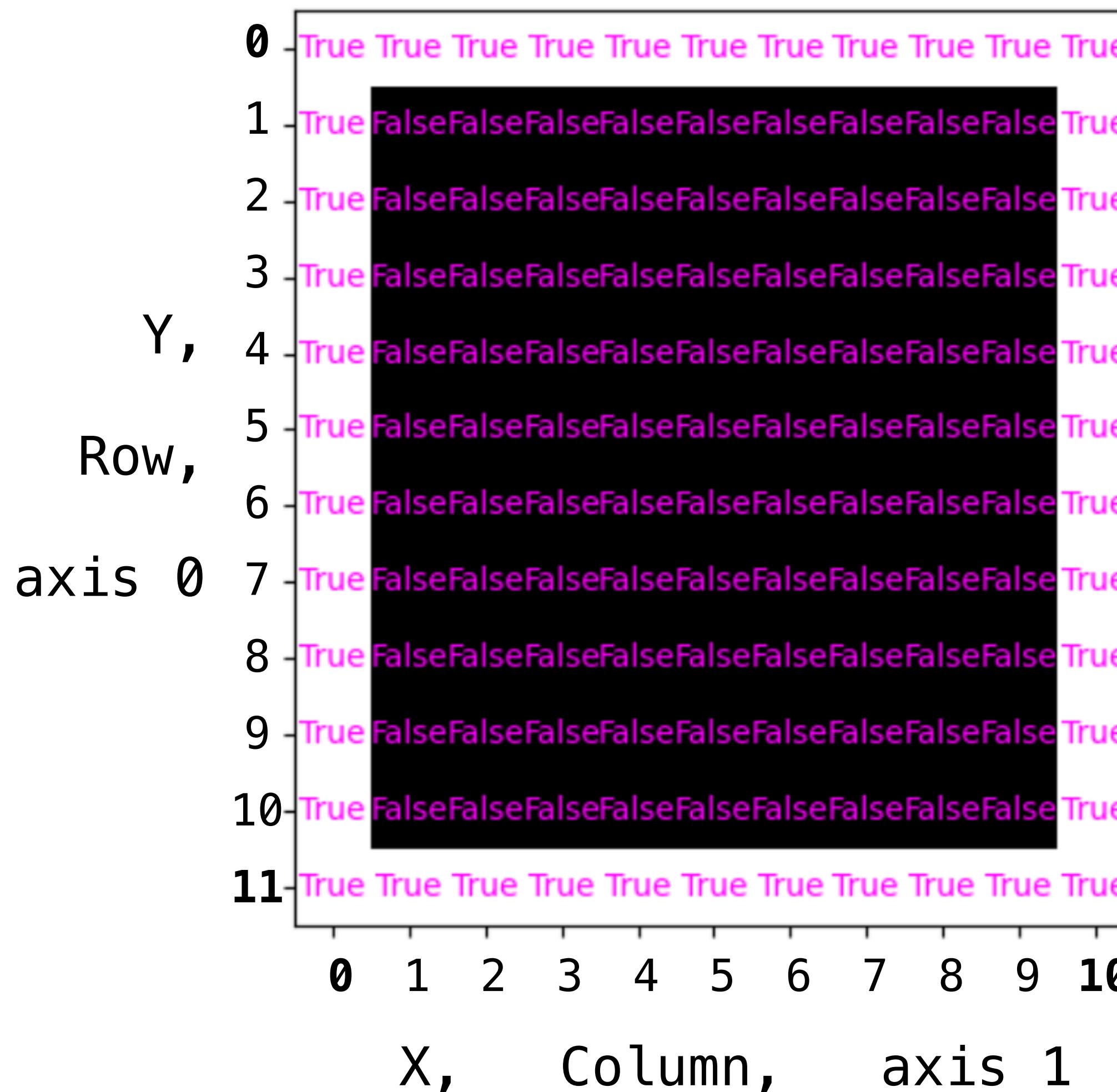
numpy arrays can contain boolean entries



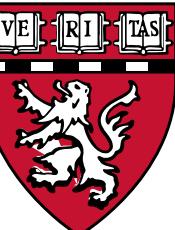
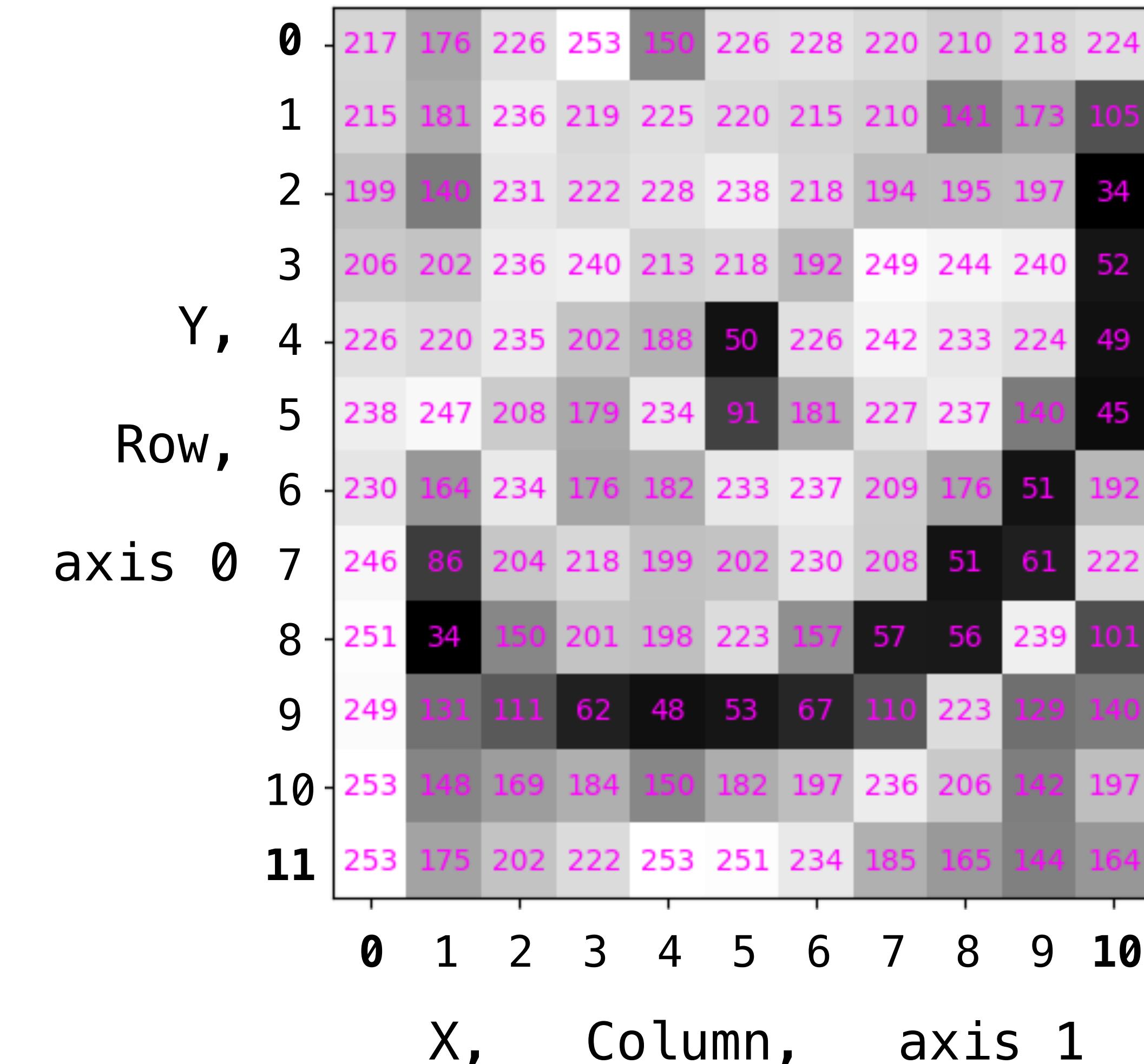


Indexing: Boolean

monocle_bool

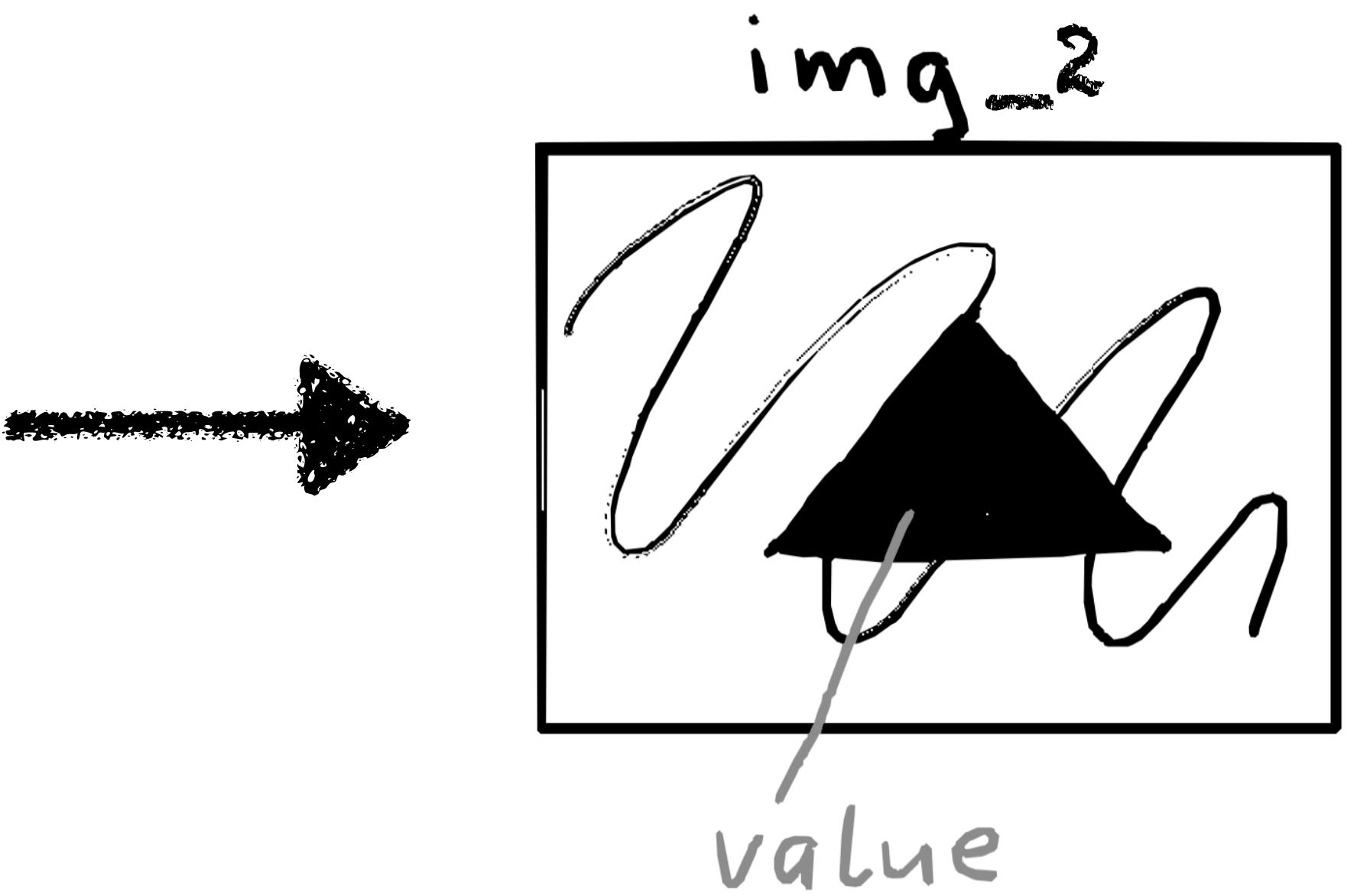
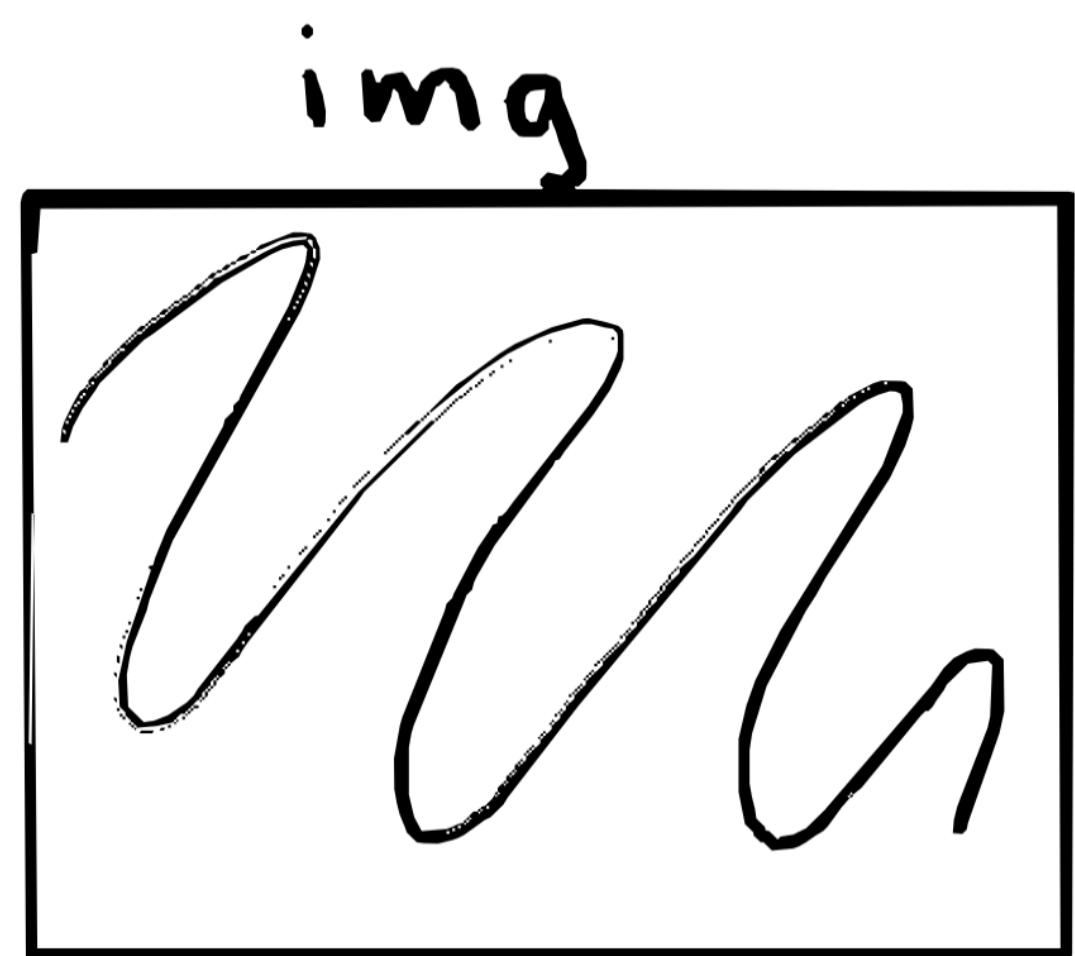


cat



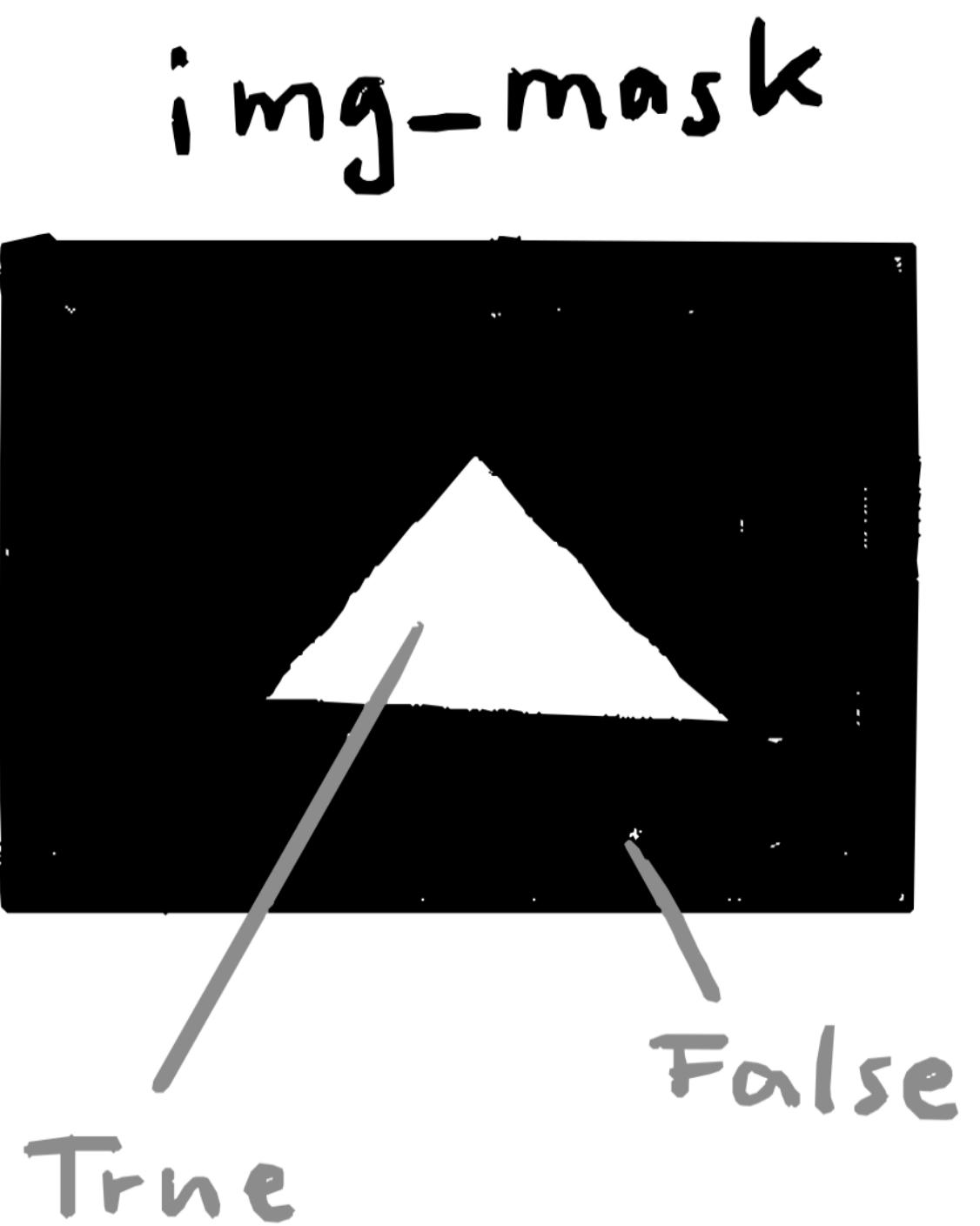
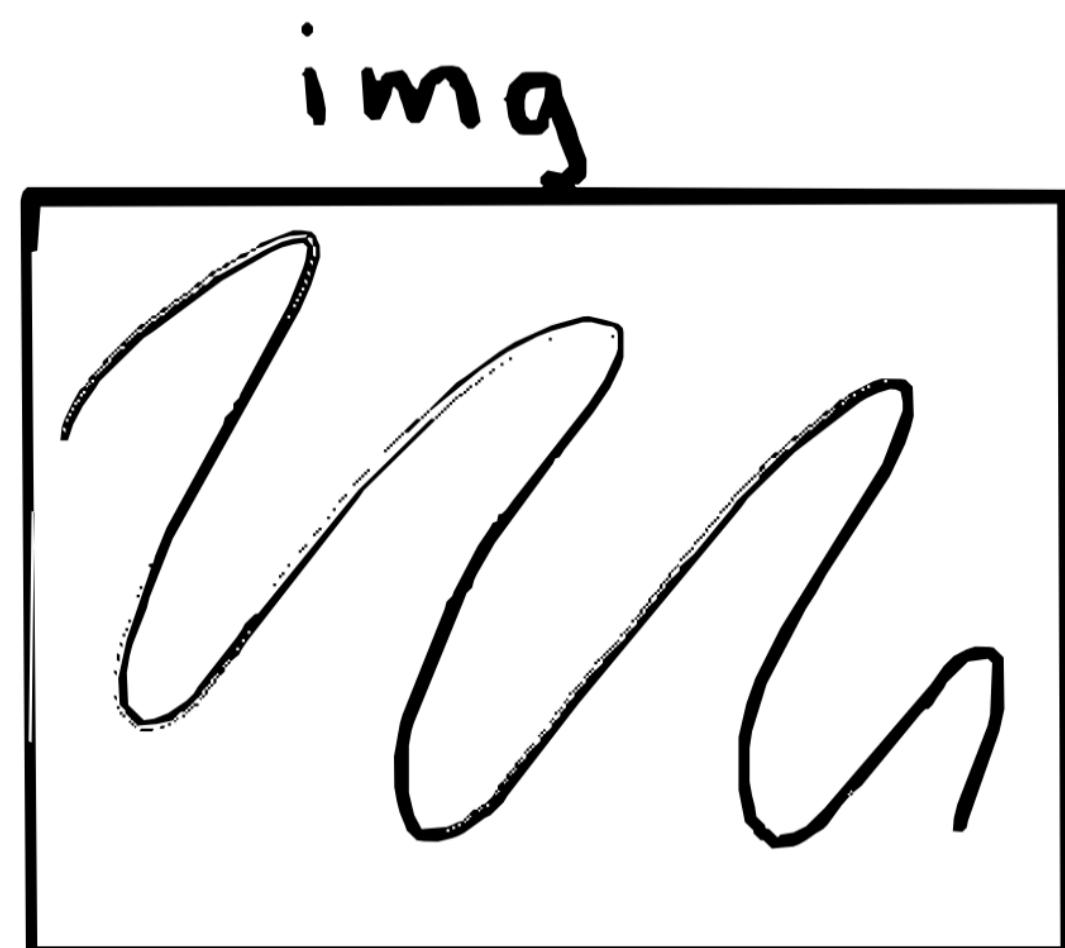


Indexing: Boolean



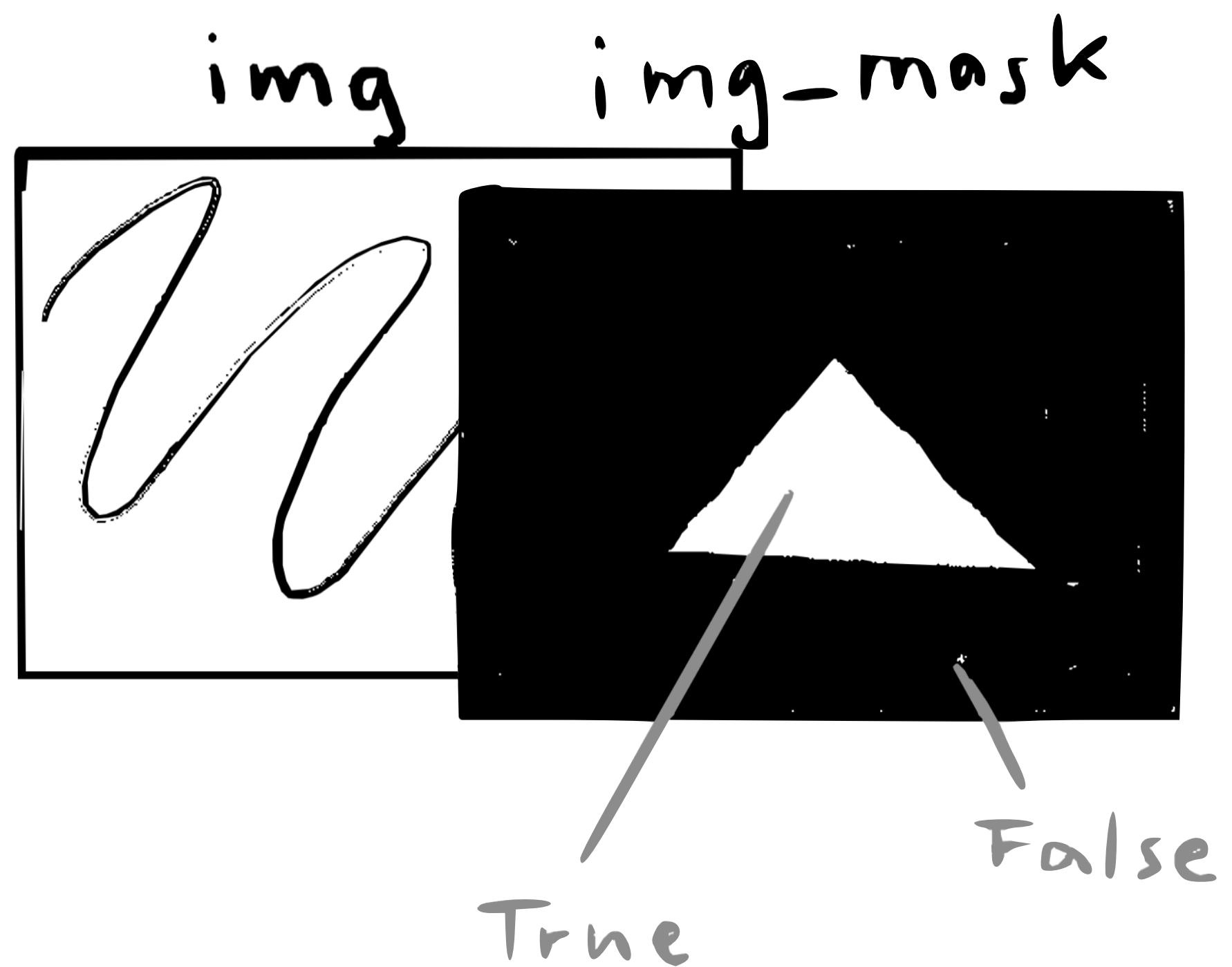


Indexing: Boolean





Indexing: Boolean



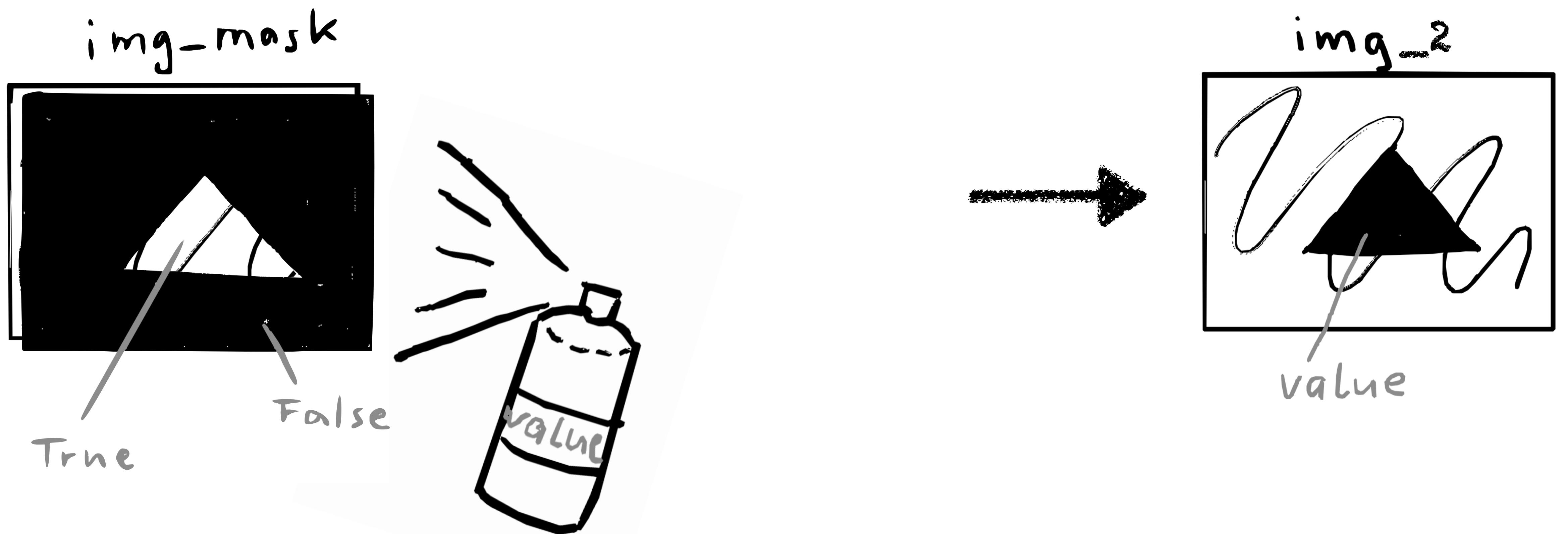


Indexing: Boolean



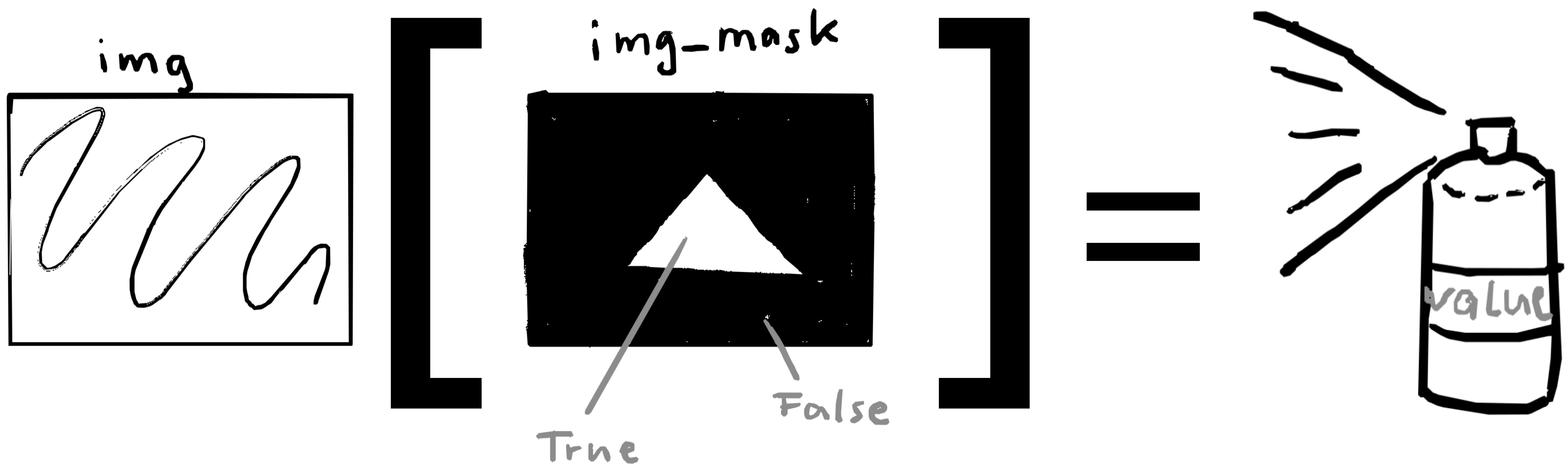


Indexing: Boolean





Indexing: Boolean





Projections

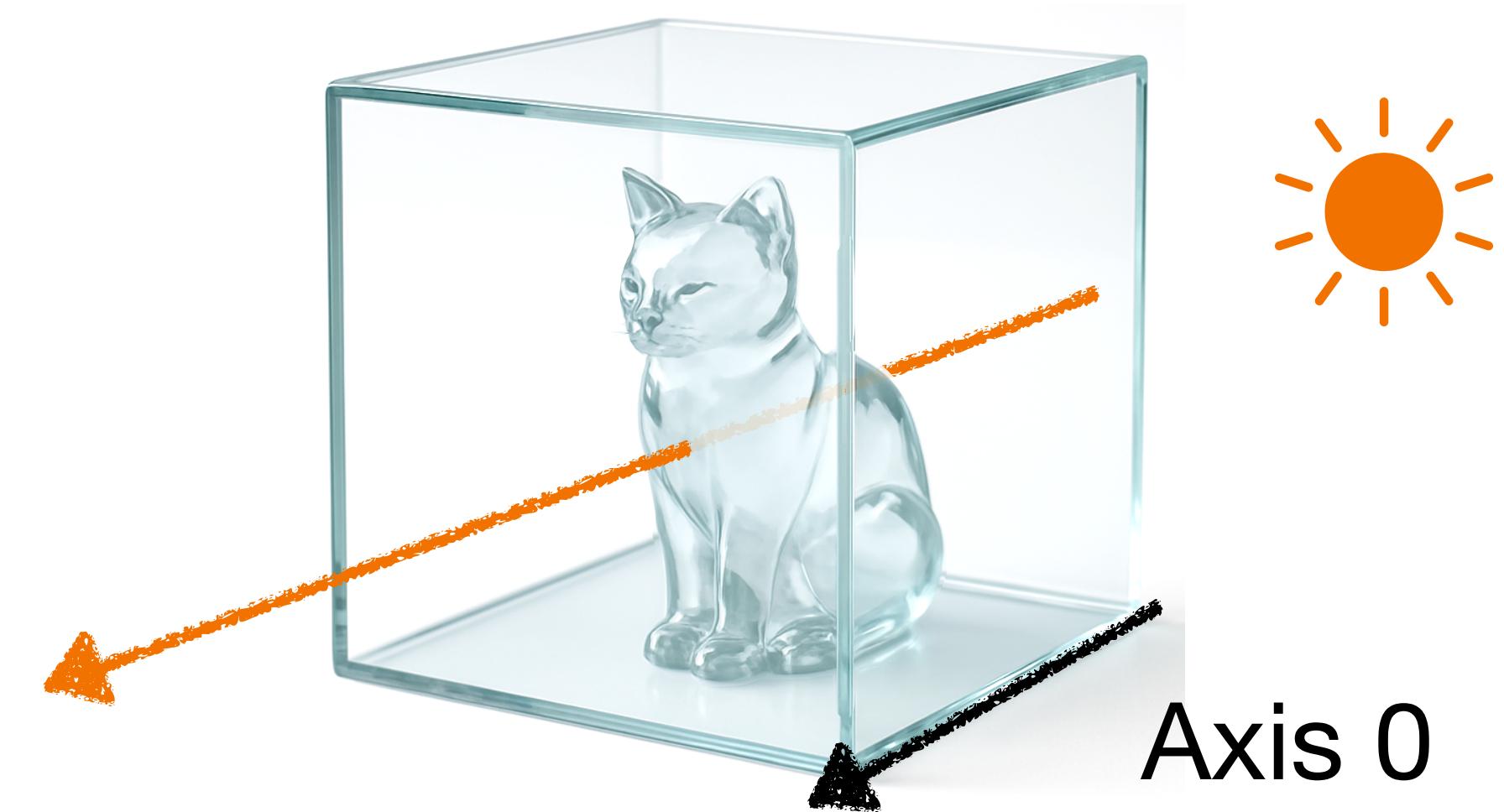


Axis 0



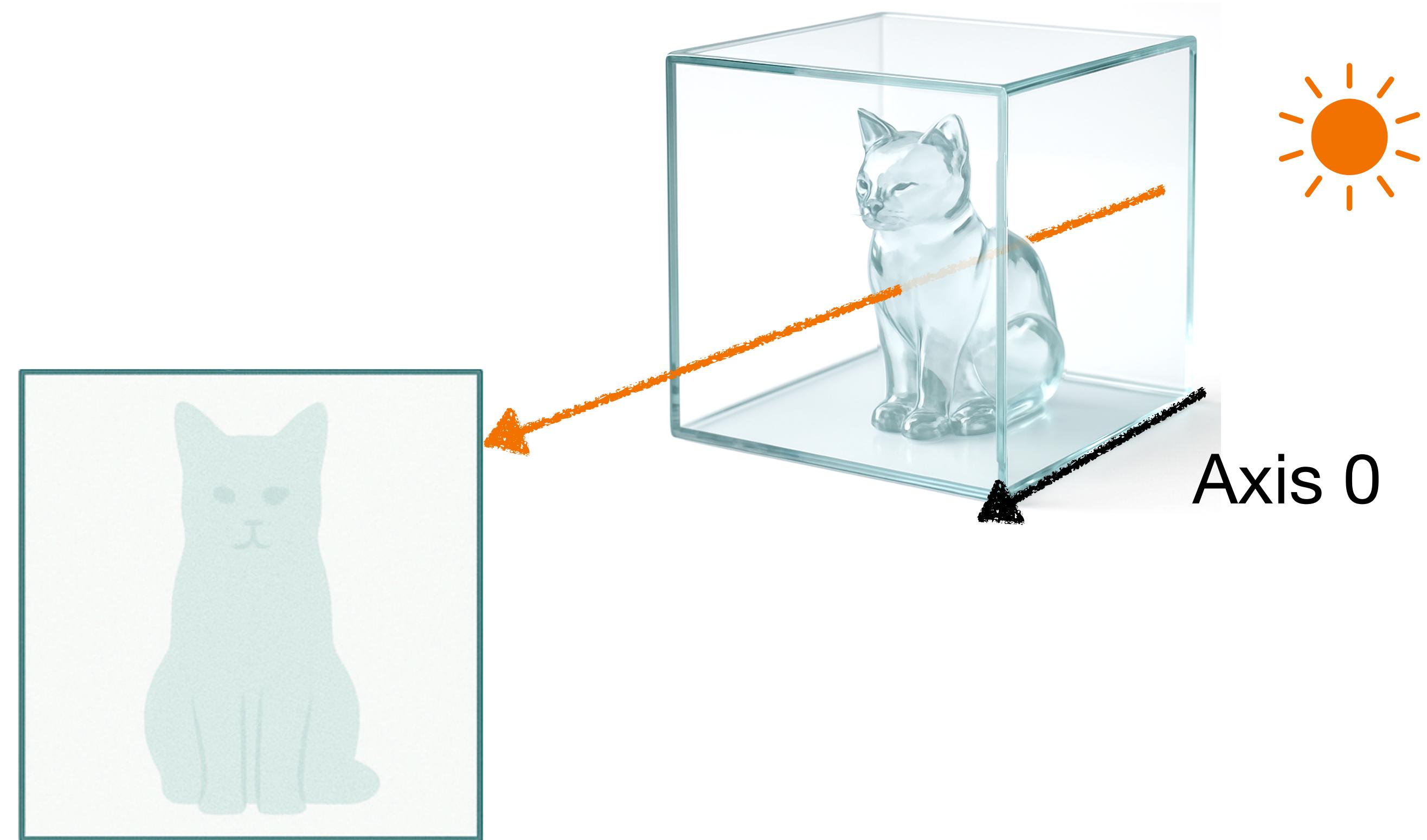


Projections



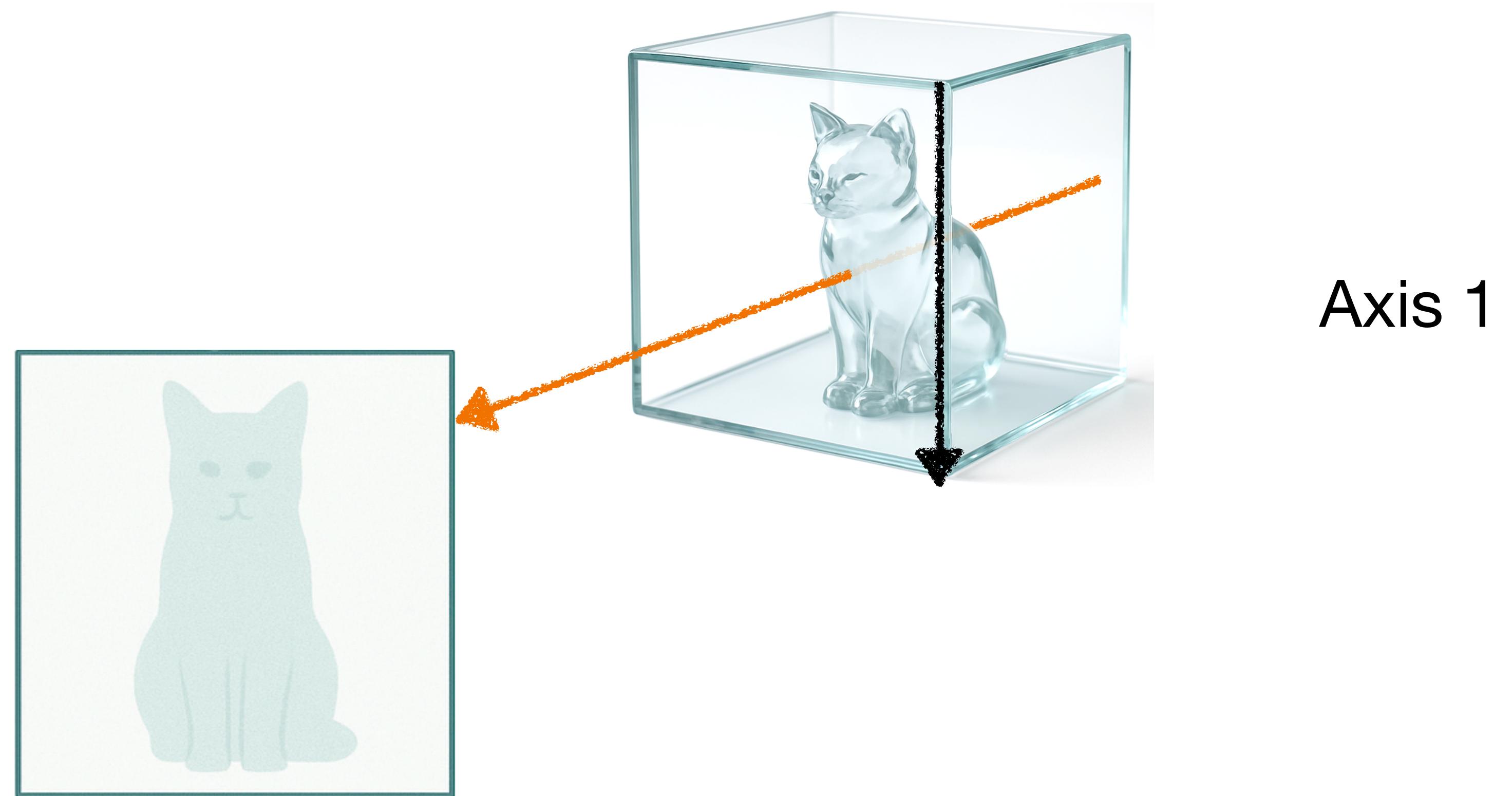


Projections



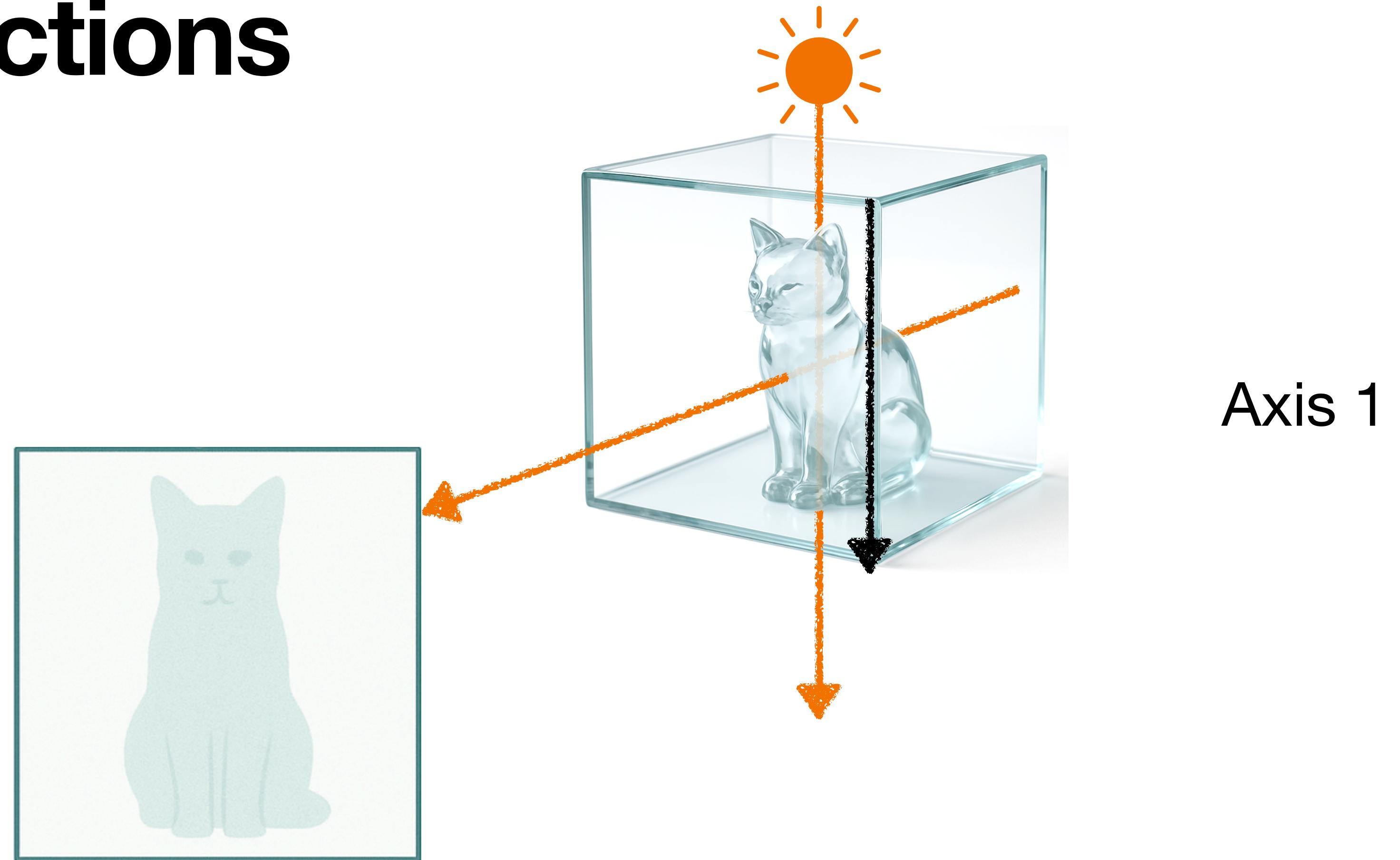


Projections



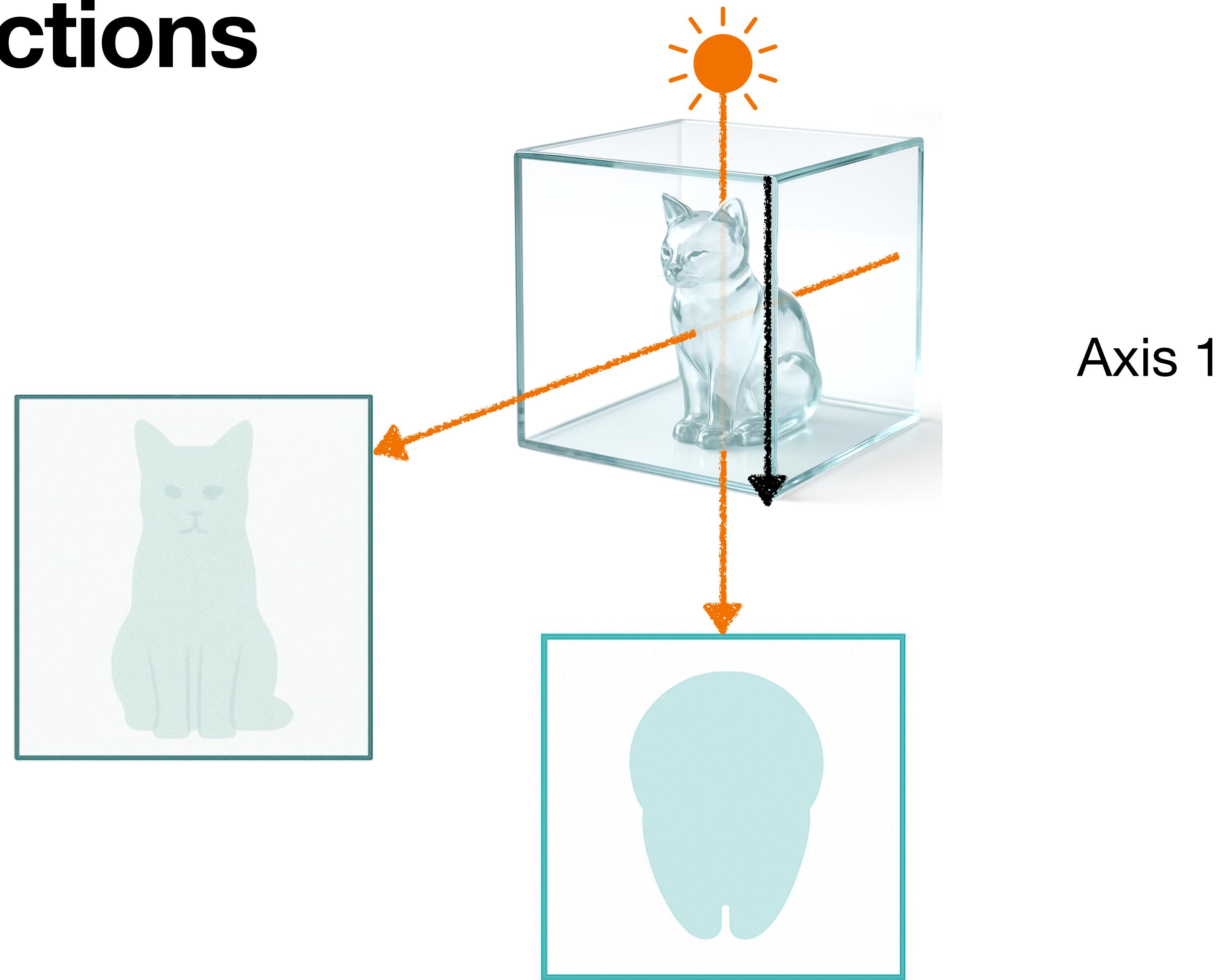


Projections





Projections

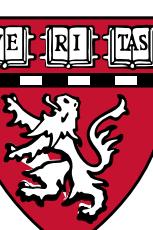
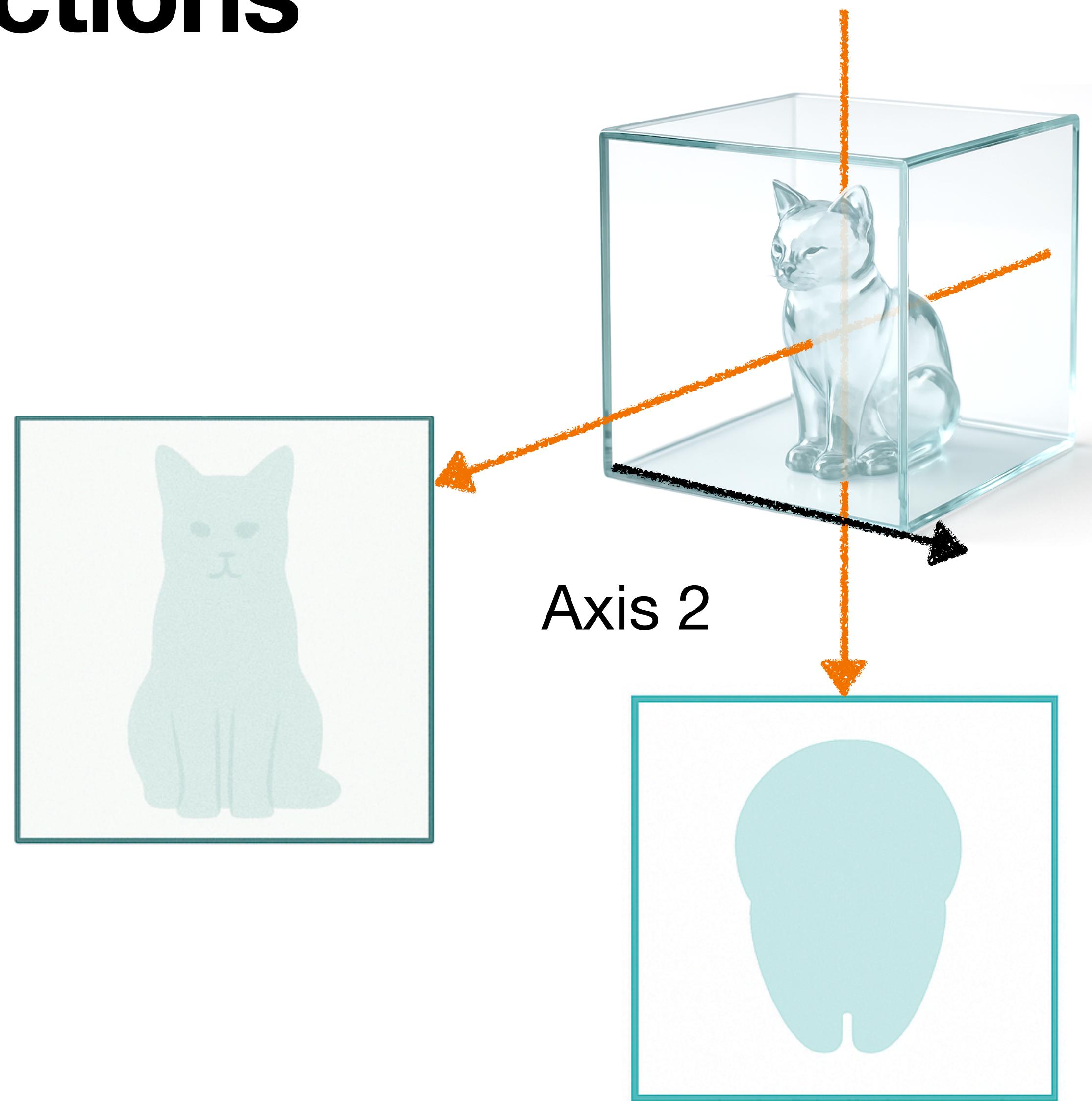


Axis 1



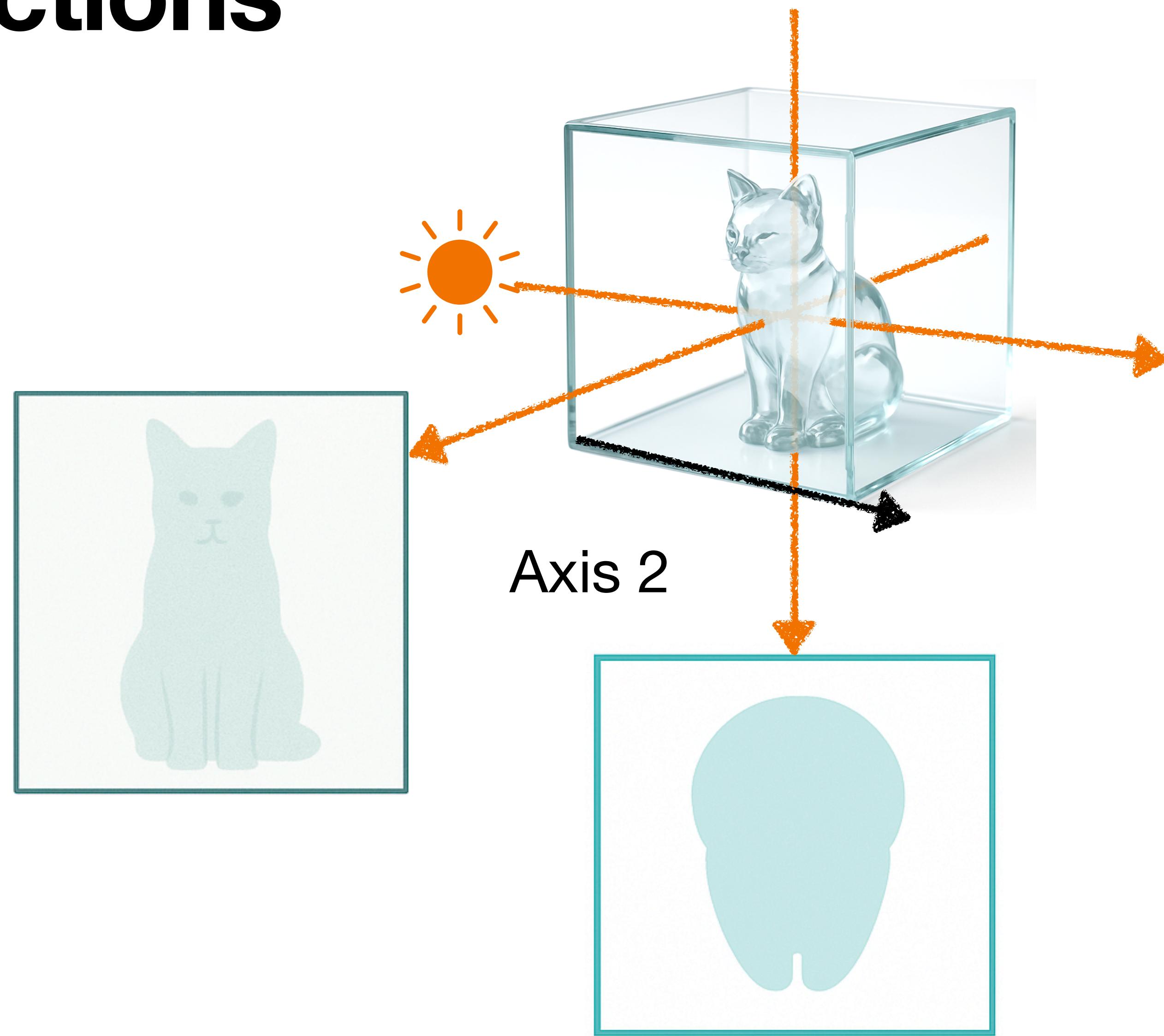


Projections



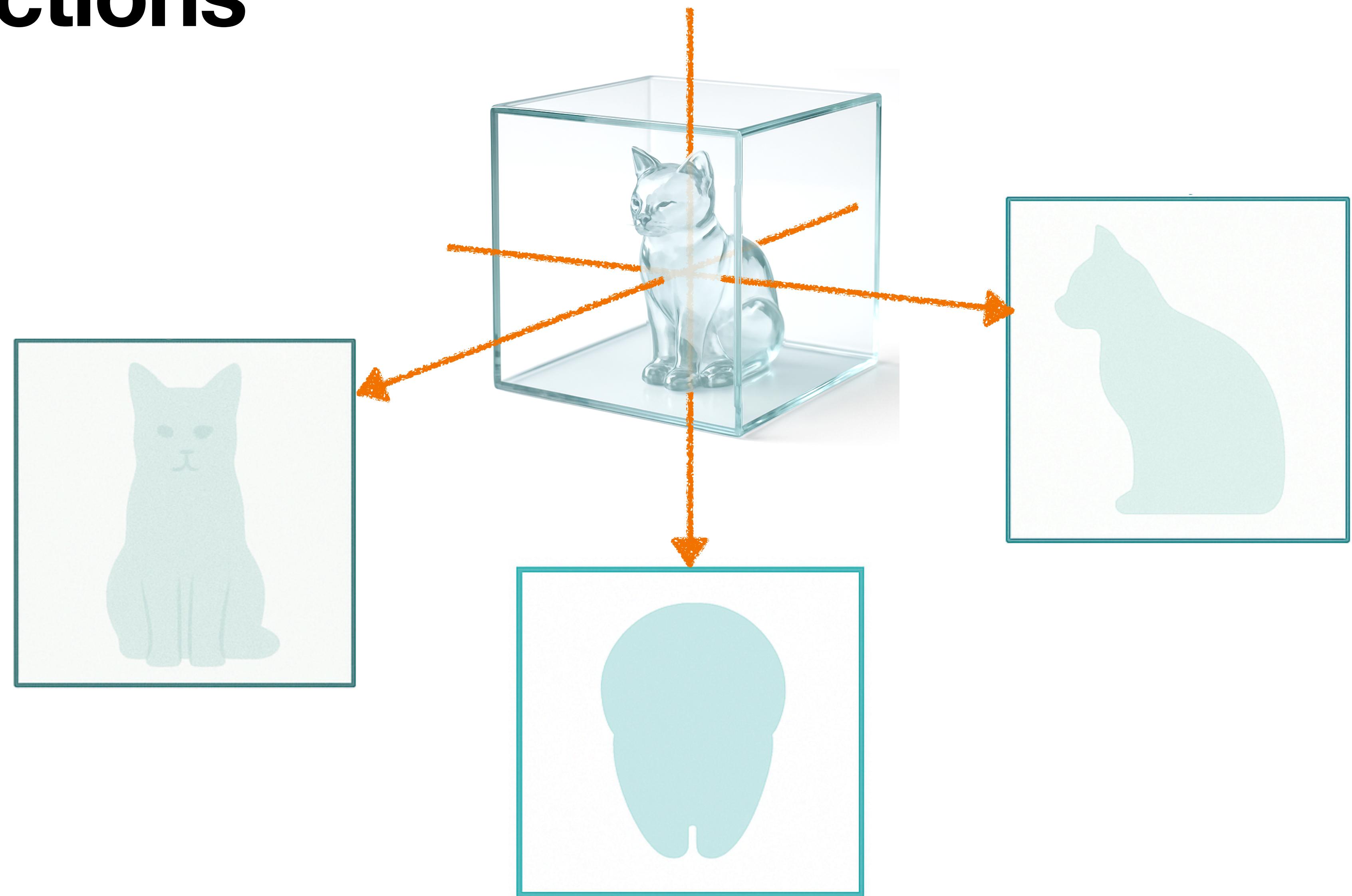


Projections



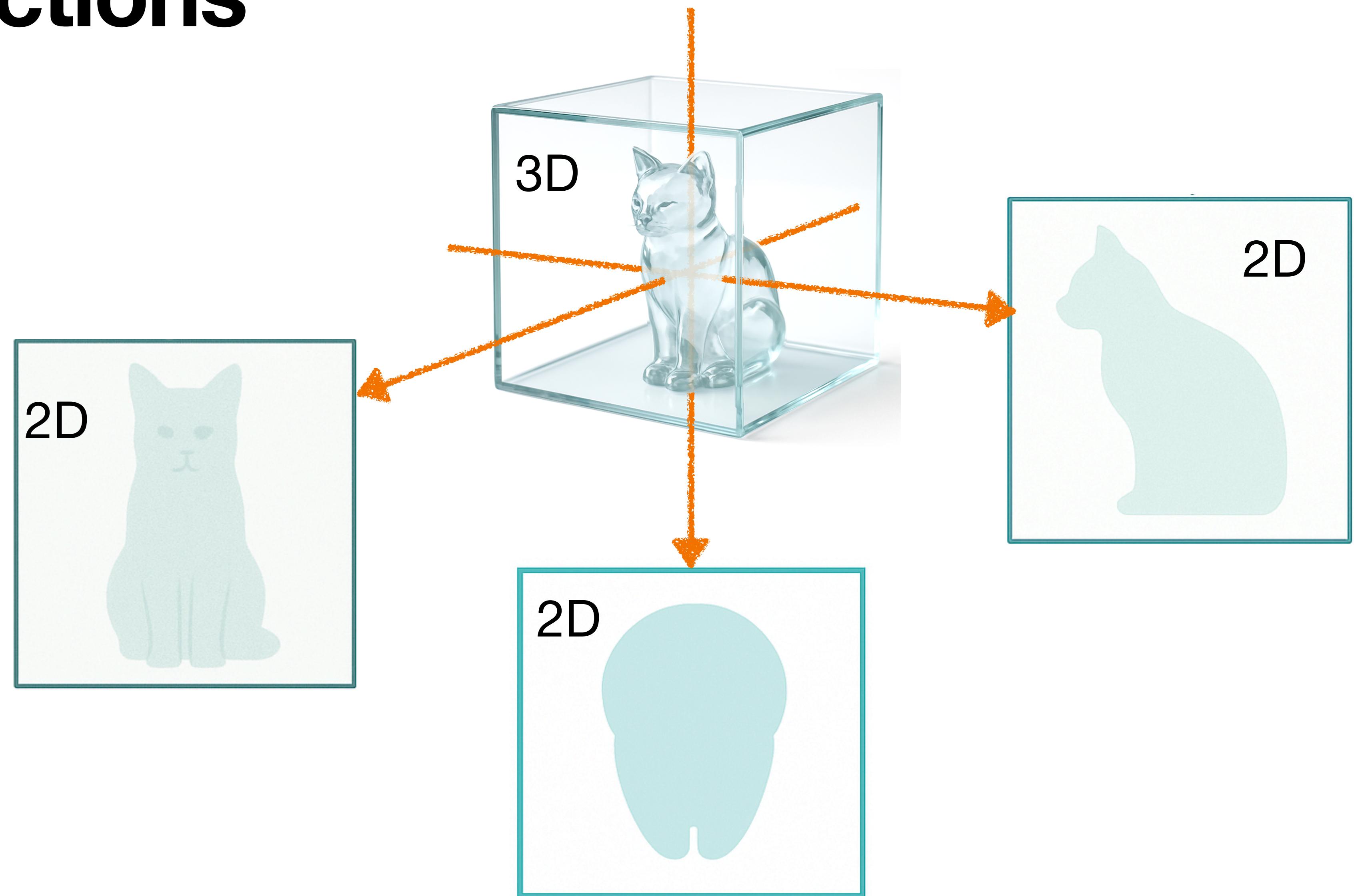


Projections



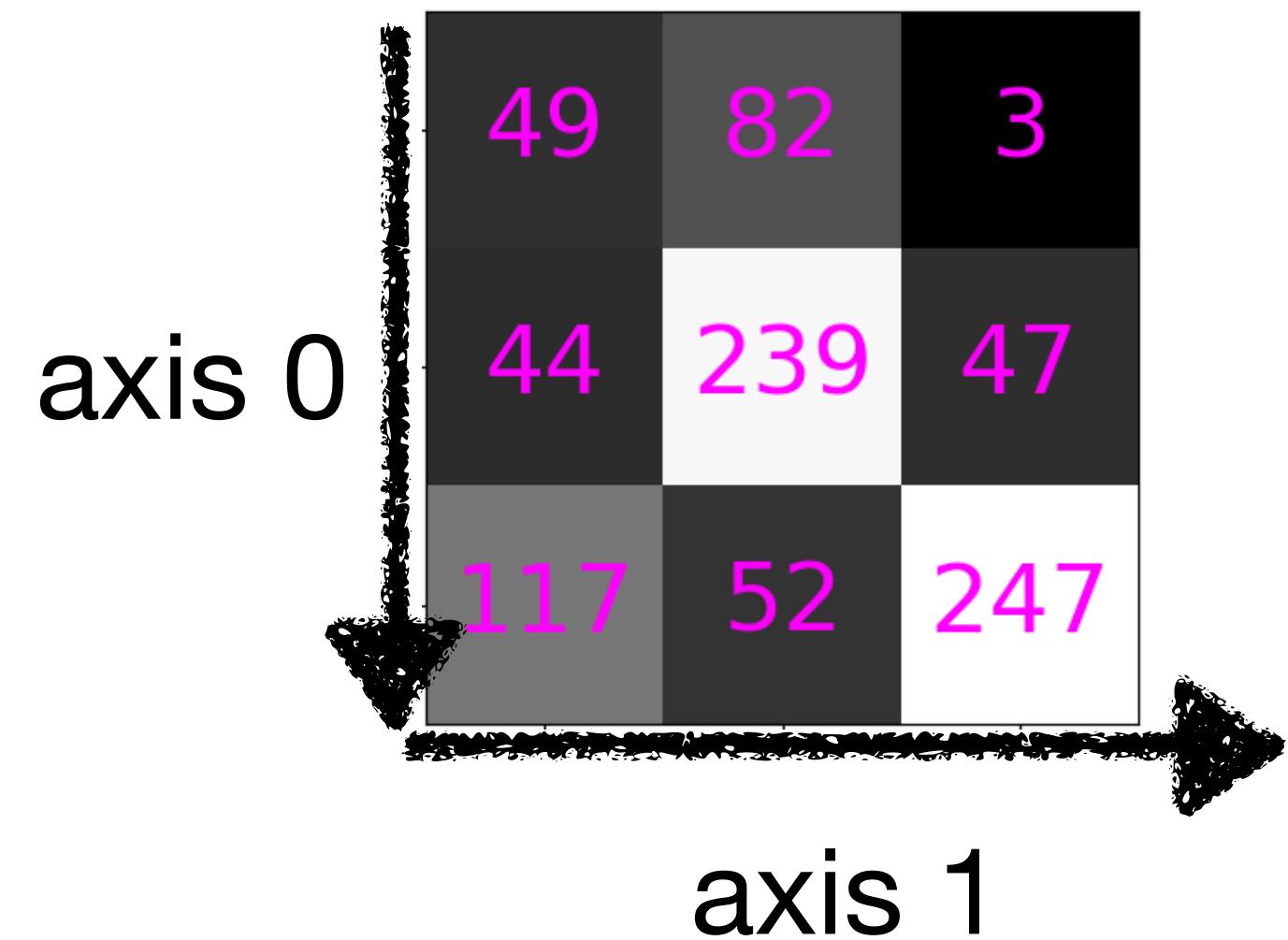


Projections





Projections

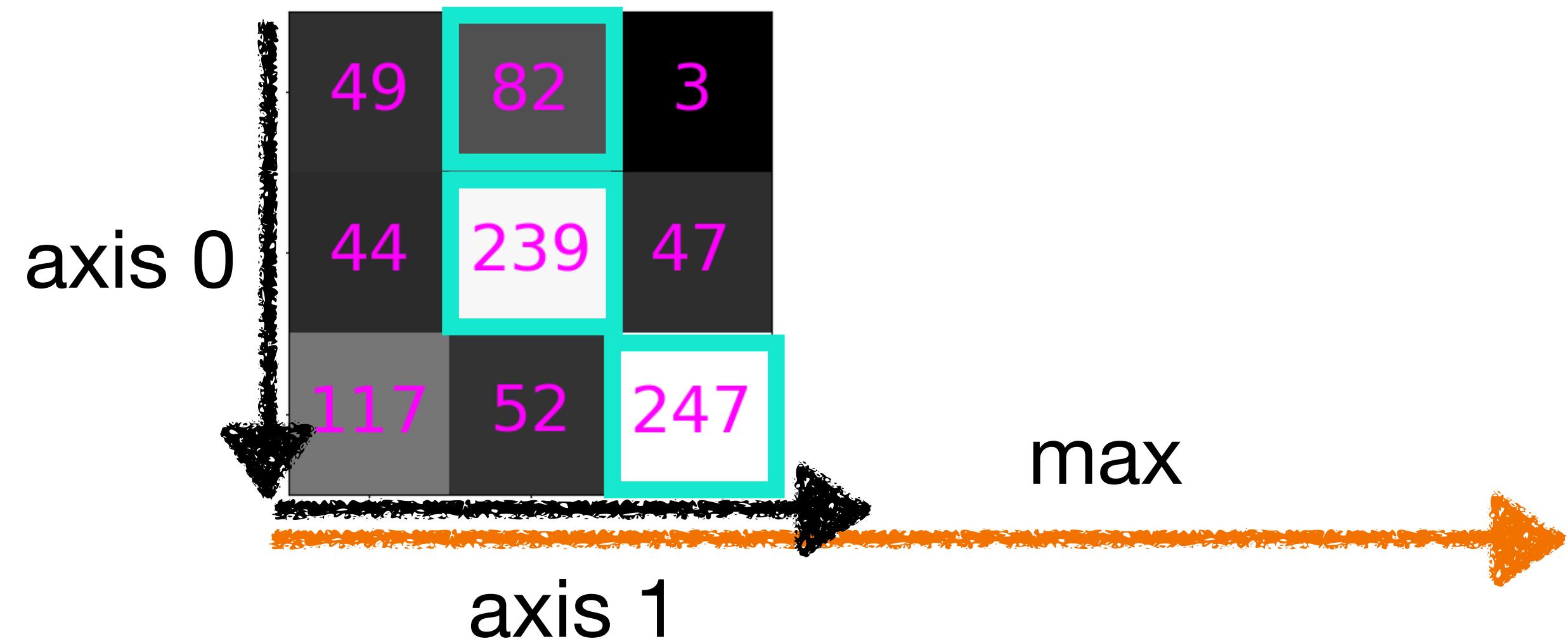


```
???? = np.max(this_array, axis = 1)
```





Projections

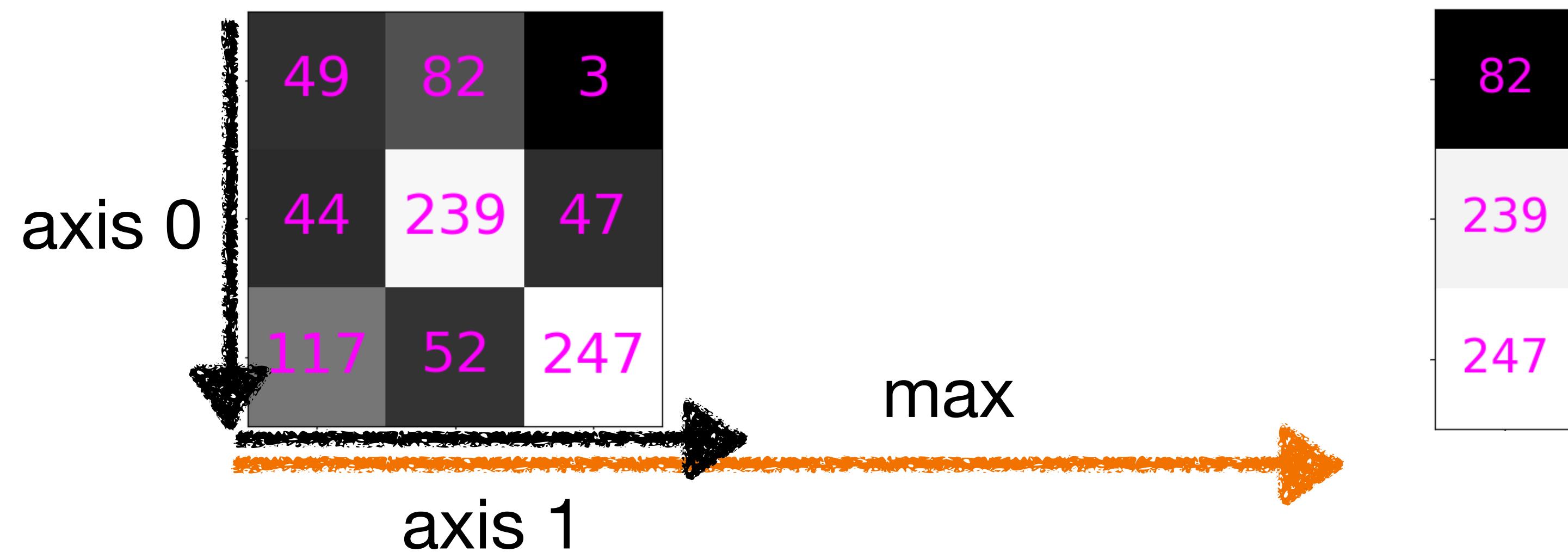


```
???? = np.max(this_array, axis = 1)
```





Projections

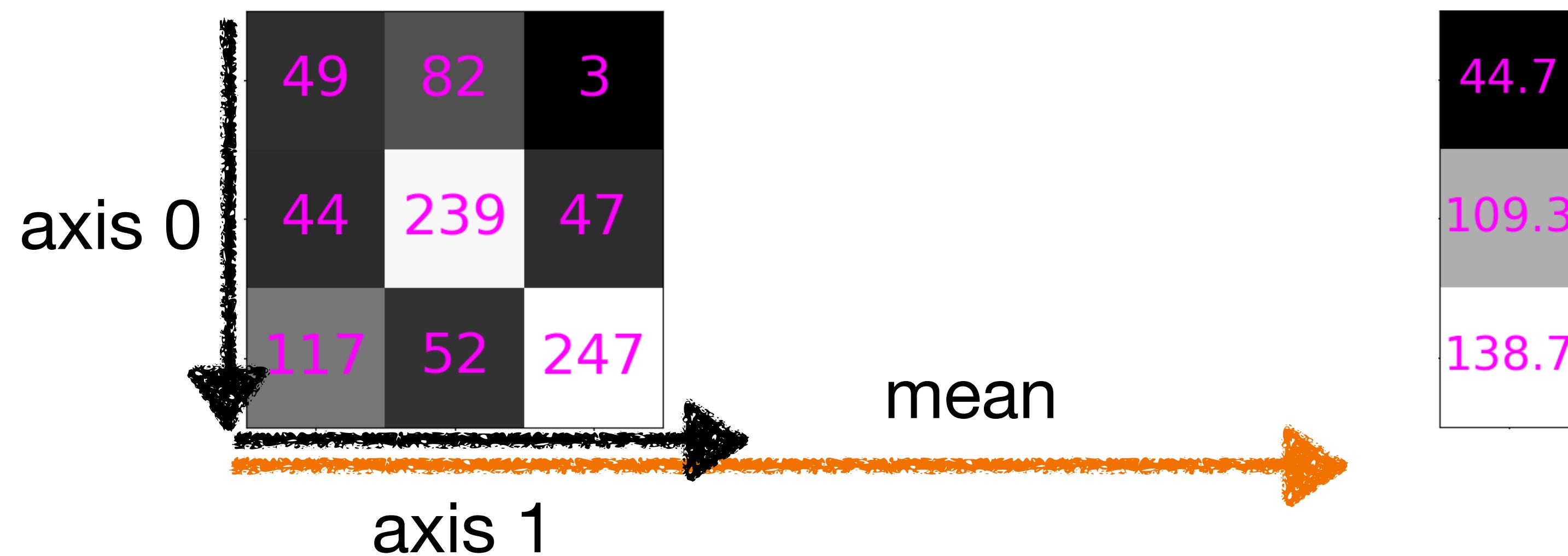
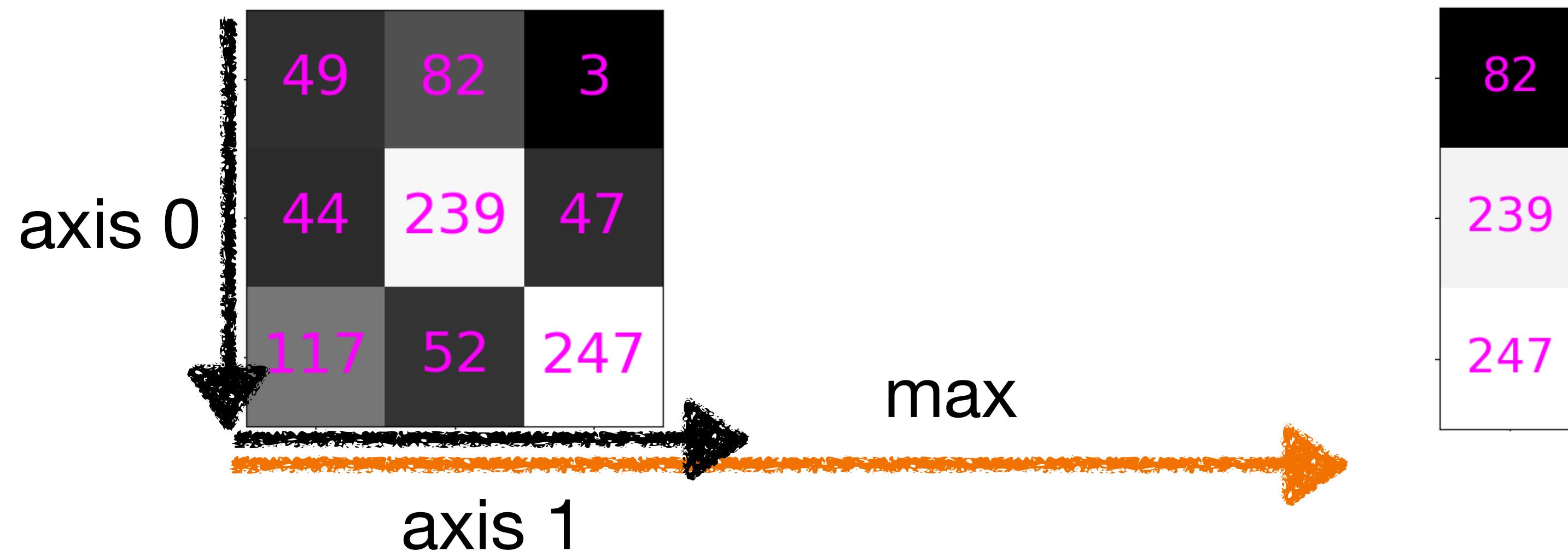


```
???? = np.max(this_array, axis = 1)
```



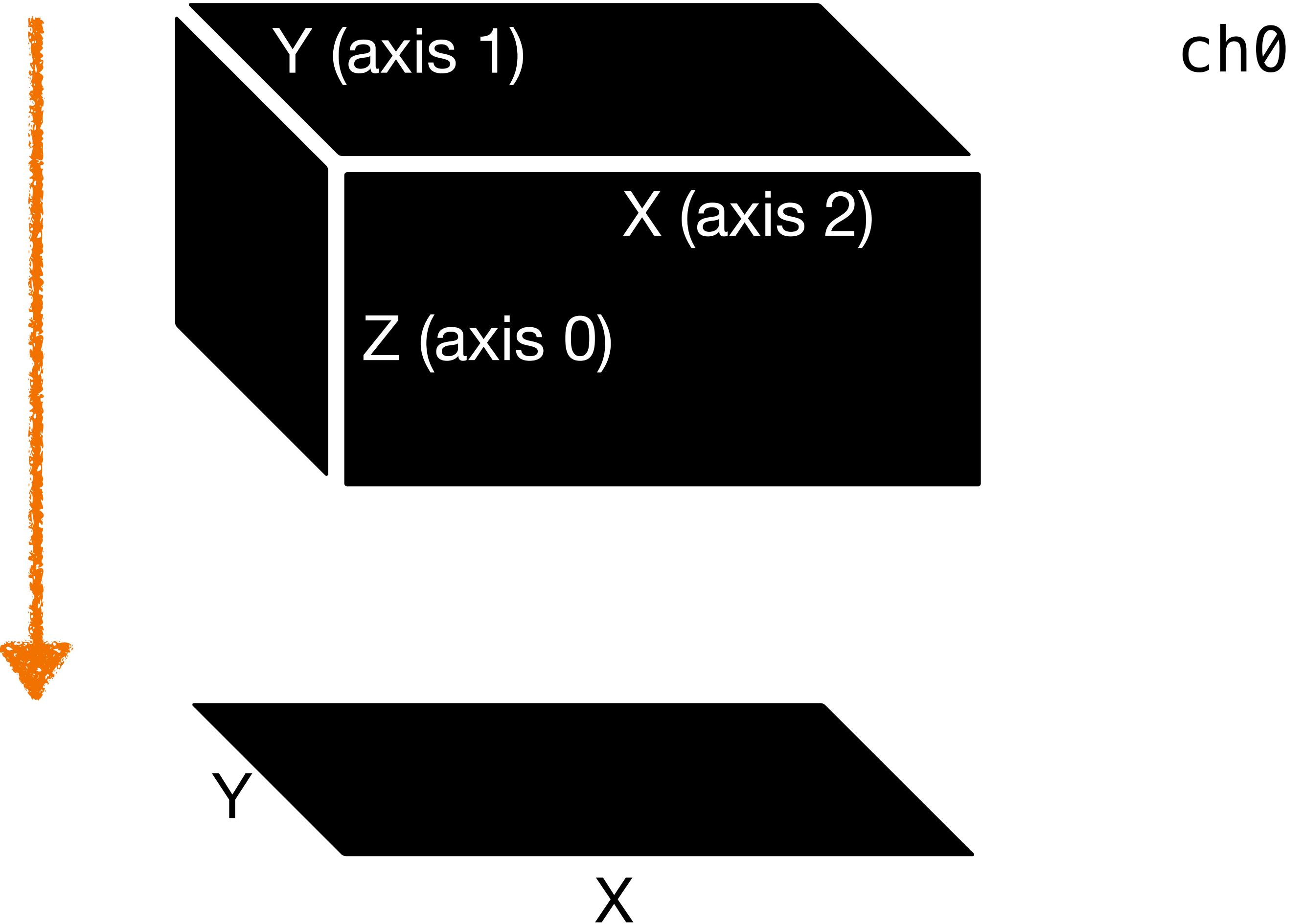


Projections





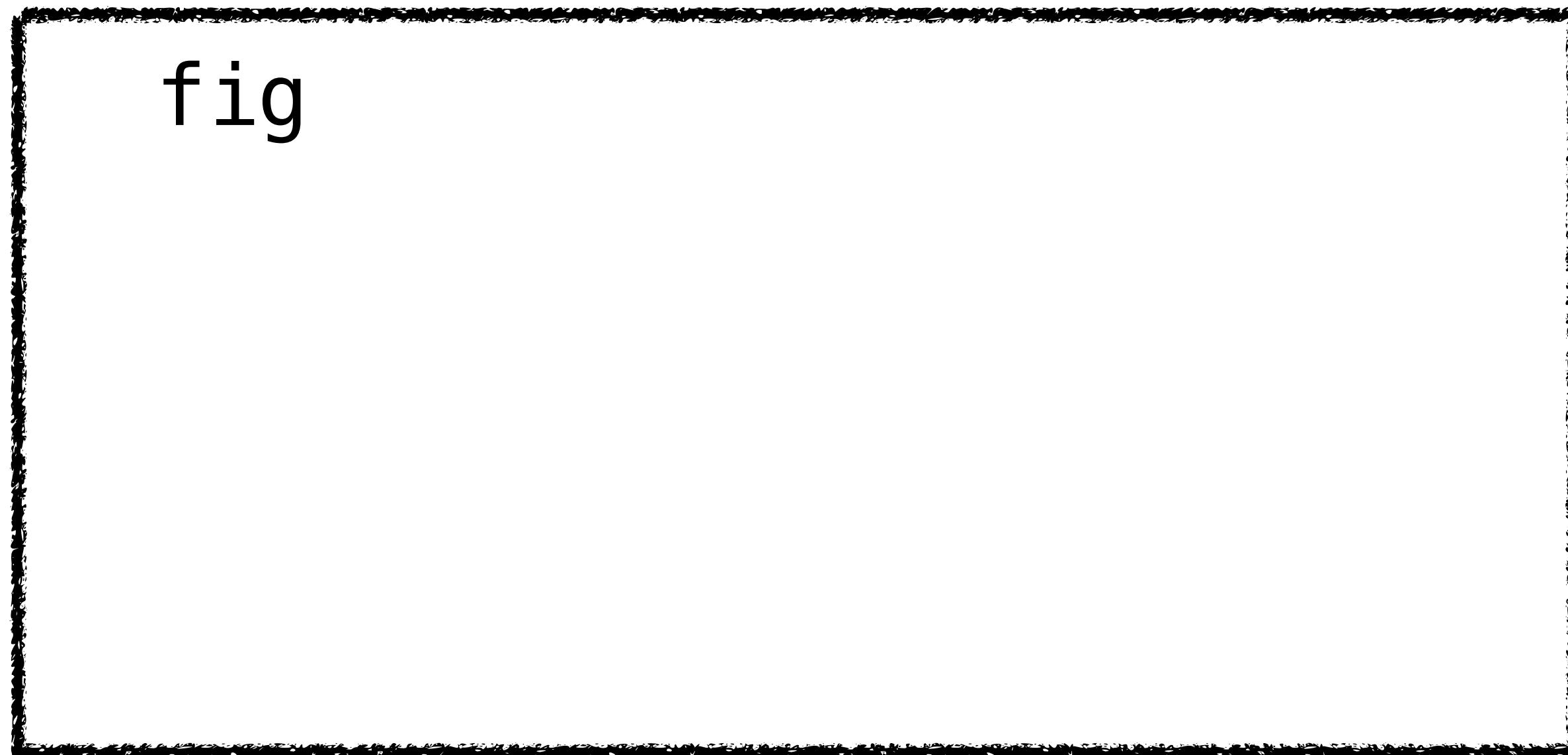
Projections





Matplotlib: Plotting

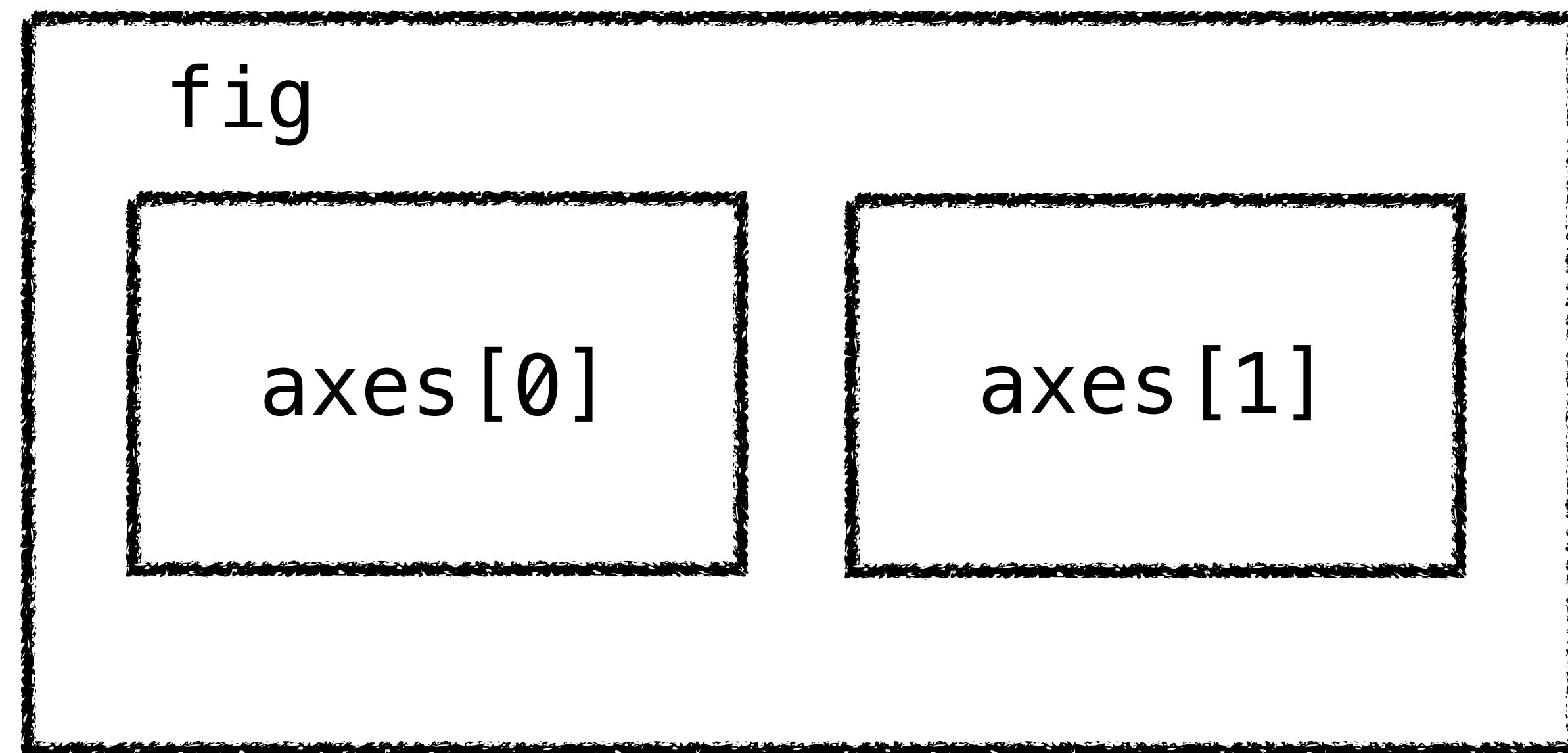
```
fig, axes = plt.subplots(1, 2)
```





Matplotlib: Plotting

```
fig, axes = plt.subplots(1, 2)
```



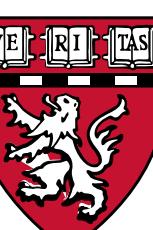
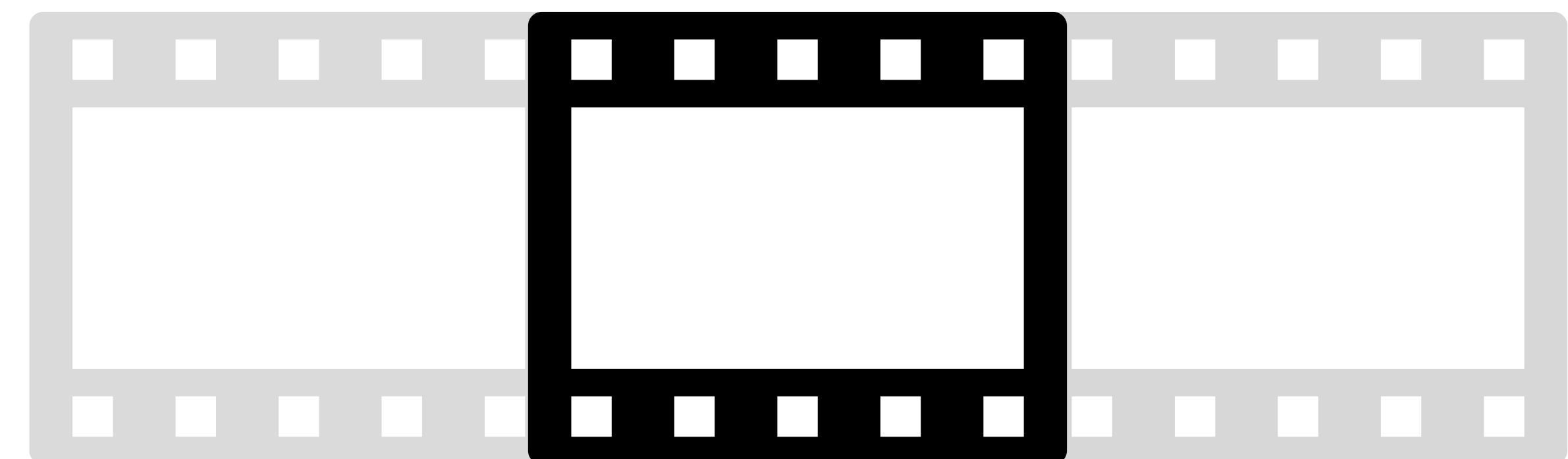


Axes of length one

```
print(img.dims)  
print(stack.shape)
```

<Dimensions [T: 1, C: 2, Z: 25, Y: 400, X: 400]>

(1, 2, 25, 400, 400)





Axes of length one

```
print(stack.shape) (1, 2, 25, 400, 400)
```

```
stack = stack.squeeze()
```



```
print(stack.shape) (2, 25, 400, 400)
```

