Last Class: Regular Indexing

① → Row - Comma - Column

$$A = \begin{bmatrix} 101 & 102 & 103 \\ 104 & 105 & 106 \end{bmatrix};$$

$$B = A(:,2) \rightarrow \begin{bmatrix} 102 \\ 105 \end{bmatrix}$$

② Linear Indexing

$$C = A(1:4) \longrightarrow \begin{bmatrix} 101 \\ 104 \\ 102 \\ 105 \end{bmatrix}$$

Logical Indexing

It's also possible to index into an array using a logical array of the same size.

↳ array of logical 1's and 0's.

Ex.

```
mag = [7.2  3.1  2.8  6.2];

ii = logical([1  1  0  0]);

mag(ii)  ⟶  [7.2  3.1]
```

Logical Indexing Cont.

Three ways to create a logical array.

(I) Convert an array of ones and zeroes using the function logical().

$$\text{logical}\Big(\underbrace{[1\ 0\ 0\ 1]}_{\substack{\text{class: double} \\ \text{(ie. a numeric value)}}}\Big) \longrightarrow \underbrace{[1\ 0\ 0\ 1]}_{\substack{\text{class: logical} \\ \text{(ie. "true" or "false")}}}$$

(II) Use a logical comparison (==, >, <, etc...).

↑ next week.

mag = [7.2 3.1 2.8 6.2]

mask = mag > 5; → [1 0 0 1]

mag (mask); → [7.2 6.2]

Note, you can do all this in one line:

mag(mag > 5); → [7.2 6.2]

(iii) Use a function that produces a logical array. Ex: isnan( ); isfinite( ); isequal( ); etc...

```
Data = [ 3 , 5 , NaN , 9 , 8, Inf ] ;
```

"Not a Number"
ie. a missing value

"infinite"

isfinite (Data) → [1 1 0 1 1 0]

logical

* Logical indexing is very useful for choosing specific subsets of a large array.