GENERAL ASSEMBLY

CLASS STARTUP

In small groups of 4, share your article find. Be able to:
1. Explain a quick summary of the article
2. What you found really great about it
3. Further questions you had, things you didn't understand, stuff you disagreed with.

We will run this for about 15 minutes, and then each group will share in more detail one of their articles.

GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LESSON 2: LINEAR ALGEBRA

LAST TIME...

WHAT IS DATA SCIENCE
DATA EXPLORATION AND WORKFLOW
PYTHON DATA STRUCTURES

ANY QUESTIONS?

I. LINEAR ALGEBRA REVIEW

II. THE PYTHON CONTROL FLOW

LAB:

III. MATRIX MULTIPLICATION IN PYTHON

IV. ADDING CONTROL FLOW INTO CLICKS AGGREGATION

I. LINEAR ALGEBRA REVIEW

LINEAR ALGEBRA REVIEW

In order to best understand most machine learning algorithms, we need some basis of linear algebra.

LINEAR ALGEBRA REVIEW

In order to best understand most machine learning algorithms, we need some basis of linear algebra.

Linear algebra is best defined as mathematics in the multidimensional space and the mapping between said spaces.

FOR EXAMPLE...

$$y = mx + b$$

FOR EXAMPLE...

$$y = m_1x_1 + m_2x_2 + b$$

FOR EXAMPLE...

$$y = m_1x_1 + m_2x_2 + m_3x_3 + m_4x_4 + b$$

FOR EXAMPLE...

$$y = m_1x_1 + m_2x_2 + m_3x_3 + m_4x_4 + m_5x_5 + m_6x_6 + m_7x_7 + m_8x_8 + m_9x_9 + m_{10}x_{10} + b$$

MATRICES

Matrices are an array of real numbers with m rows and n columns

Each value in a matrix is called an entry.

$$\begin{matrix} 1 & 5 & 8 & 7 \\ 2 & 1 & 3 & 6 \\ 3 & 5 & 1 & 0 \\ 4 & 6 & 0 & 1 \end{matrix}$$

Matrices are an array of real numbers with m rows and n columns

Each value in a matrix is called an entry.

$$A_{2\,1} \longrightarrow \begin{matrix} 1 & 5 & 8 & 7 \\ 2 & 1 & 3 & 6 \\ 3 & 5 & 1 & 0 \\ 4 & 6 & 0 & 1 \end{matrix}$$

Vectors are a special kind of matrix, as they only consist of one dimension of real numbers.

These look most like a numeric array (or **list**) in Python.

$$\begin{bmatrix} 1 & 3 & 9 & 2 \end{bmatrix}$$

Likewise, you can refer to each index or value similarly
(a[0] in Python is the same entity as 0 in vector a)

Rule 1!
Matrices can be added together only when they are the same size.
If they are not the same size, their sum is **undefined**.

$$\begin{bmatrix} 1 & 3 & 9 & 2 \end{bmatrix} + \begin{bmatrix} 2 & 5 & 9 & 4 \end{bmatrix} = \begin{bmatrix} 3 & 8 & 18 & 6 \end{bmatrix}$$

Rule 1!
Matrices can be added together only when they are the same size.
If they are not the same size, their sum is **undefined.**

$$\begin{bmatrix} 1 & 3 & 9 & 2 \end{bmatrix} + \begin{bmatrix} 2 & 5 & 9 & 4 \end{bmatrix} = \begin{bmatrix} 3 & 8 & 18 & 6 \end{bmatrix}$$

$$\begin{bmatrix} 8 & 72 & 3 & 1 \end{bmatrix} + \begin{bmatrix} 17 & 55 & 3 & 10 \end{bmatrix} = ?$$

HANDLING

Rule 2!

Matrices can be multiplied by a scalar (single entity) value.

Each value in the matrix is multiplied by the scalar value.

$$\begin{bmatrix} 1 & 3 & 9 & 2 \end{bmatrix} * 3 = \begin{bmatrix} 3 & 9 & 27 & 6 \end{bmatrix}$$

$$\begin{bmatrix} 8 & 72 & 3 & 1 \end{bmatrix} * 2 = ?$$

Rule 3!

Matrices and vectors can be multiplied together given that the matrix columns are as wide as the vector is long.

The result will always be a vector.

$$\begin{matrix}1 & 3 & 9 & 2\\2 & 4 & 6 & 8\end{matrix} * \begin{matrix}2\\3\\6\\5\end{matrix} = \begin{matrix}2+9+54+10\\4+12+36+40\end{matrix} = \begin{matrix}75\\92\end{matrix}$$

Rule 4!

Matrices can be multiplied together using the same rules that we have from matrix-vector multiplication.

The result will always be a matrix.

$$\begin{matrix} 1 & 3 & 9 & 2 \\ 2 & 4 & 6 & 8 \end{matrix} * \begin{matrix} 2 & 1 \\ 3 & 2 \\ 6 & 0 \\ 5 & 4 \end{matrix} = \begin{matrix} 75 & 15 \\ 92 & 40 \end{matrix}$$

THAT WAS FAST... WHY DOES THIS MATTER?

Matrices represent the multiple dimensions in our data! If we had a a vector that suggested how important each dimension of our data was, we could use that to find our best **linear model**.

THAT WAS FAST... WHY DOES THIS MATTER?

Matrices represent the multiple dimensions in our data! If we had a vector that suggested how important each dimension of our data was, we could use that to find our best **linear model**.

We will see matrices quite often in **all** of our data, so pay careful attention to how data is structured and how different algorithms interact with them

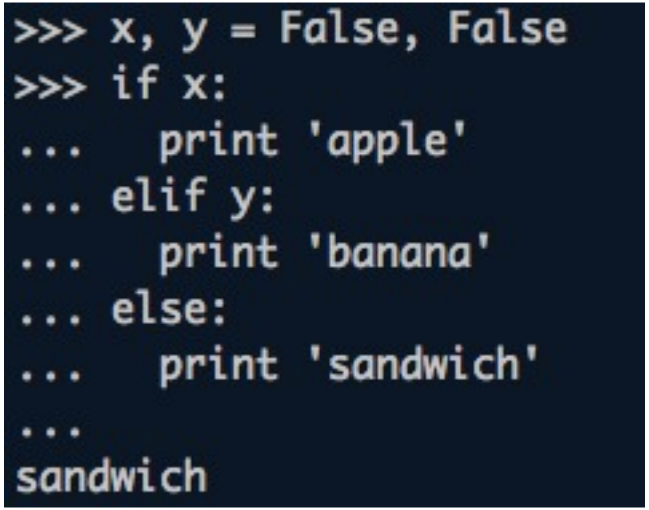
REVIEW

1. Complete the equations on the board.

II. PYTHON CONTROL FLOW

CONTROL FLOW

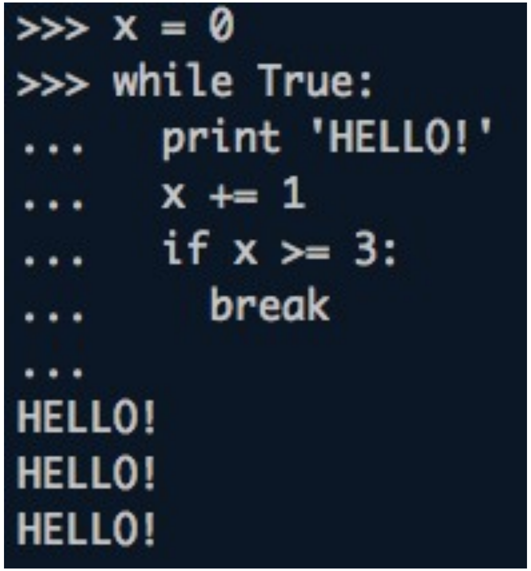
Python has a number of control flow tools that will be familiar from other languages. The first is the if-else statement, whose compound syntax looks like this:



```
>>> x, y = False, False
>>> if x:
...     print 'apple'
... elif y:
...     print 'banana'
... else:
...     print 'sandwich'
...
sandwich
```

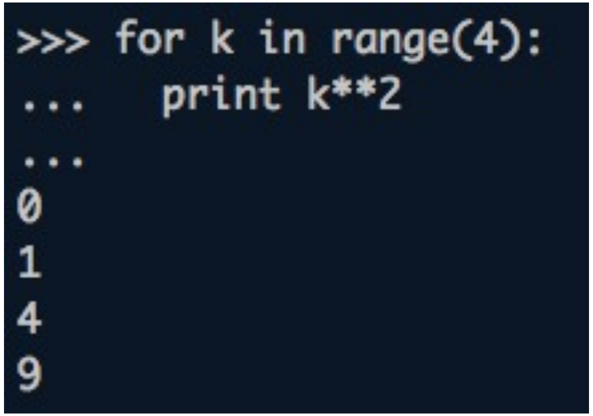
CONTROL FLOW

Next is the while loop. **This executes while a given condition evaluates to True.**



```
>>> x = 0
>>> while True:
...     print 'HELLO!'
...     x += 1
...     if x >= 3:
...       break
...
HELLO!
HELLO!
HELLO!
```

Another familiar (and useful) construct is the for loop. This executes a block of code for a range of values.

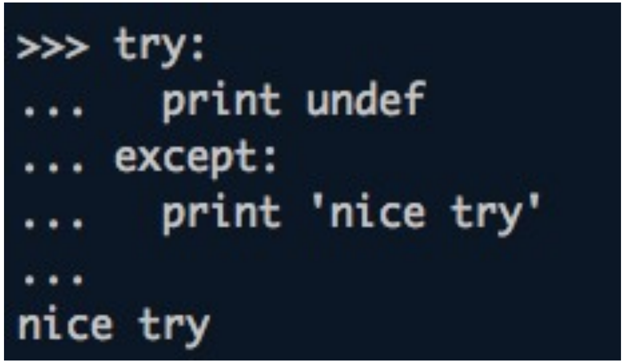


```
>>> for k in range(4):
...     print k**2
...
0
1
4
9
```

The object that a for loop iterates over is called (appropriately) an iterable.

CONTROL FLOW

A useful but possibly unfamiliar construct is the try-except block:

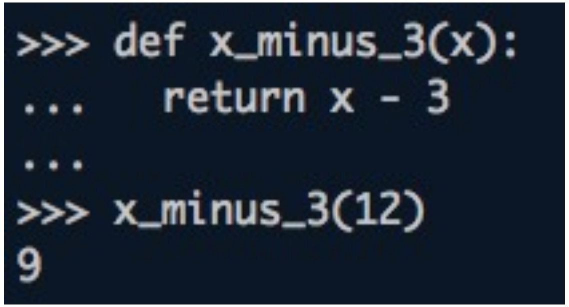


```
>>> try:
...   print undef
... except:
...   print 'nice try'
...
nice try
```

This is useful for catching and dealing with errors, also called exception handling.

FUNCTIONS

Python allows you to define custom functions as you would expect:

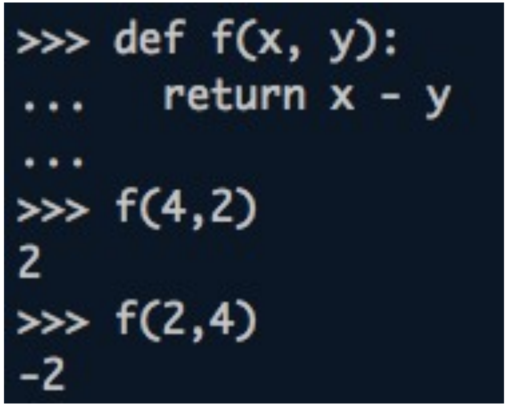


```
>>> def x_minus_3(x):
...     return x - 3
...
>>> x_minus_3(12)
9
```

Functions can optionally return a value with a return statement (as this example does).

Functions can take a number of arguments as inputs, and these arguments can be specified in two ways:

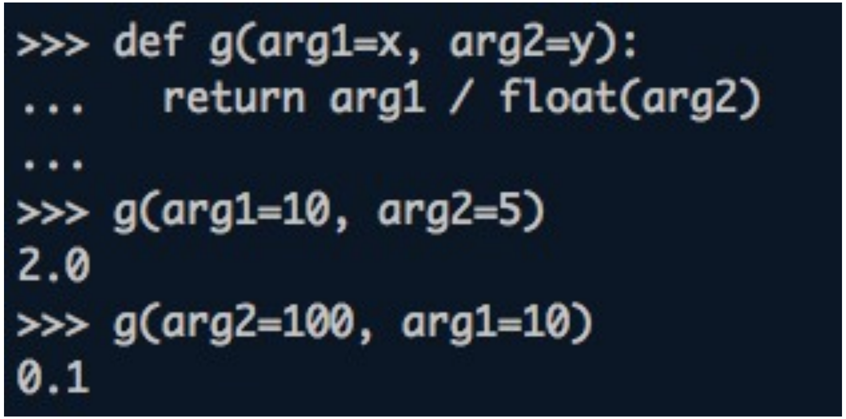
As positional arguments:



```
>>> def f(x, y):
...     return x - y
...
>>> f(4,2)
2
>>> f(2,4)
-2
```

Functions can take a number of arguments as inputs, and these arguments can be specified in two ways:

Or as keyword arguments:



```
>>> def g(arg1=x, arg2=y):
...     return arg1 / float(arg2)
...
>>> g(arg1=10, arg2=5)
2.0
>>> g(arg2=100, arg1=10)
0.1
```

LAB: MATRIX ADDITION IN PYTHON

DISCUSSION:

How does the workflow currently feel different from your current ‘data day to day?’ How is it the same?

The first machine learning problem we’ll work with (and will directly apply this linear algebra)

IN CLASS WORK

Write extra functions: mVectorMatrix() and mMatrixMatrix() that do the corresponding multiplications. No numpy cheating!

Write a function that creates an identity matrix. Explanation is included in the wiki.

NEXT CLASS SUBJECT: MACHINE LEARNING AND PYTHON LIBRARIES