Tuesday, October 20, 2020 | **Class #5**

Control flow

`For` loops, `while` loops, list comprehensions, and `if` statements

OCEAN 215 | Autumn 2020

Ethan Campbell and Katy Christensen

Using indexing to both retrieve **and** replace values

Google Doc with activities (also accessible from Canvas Modules or Google Drive folder):

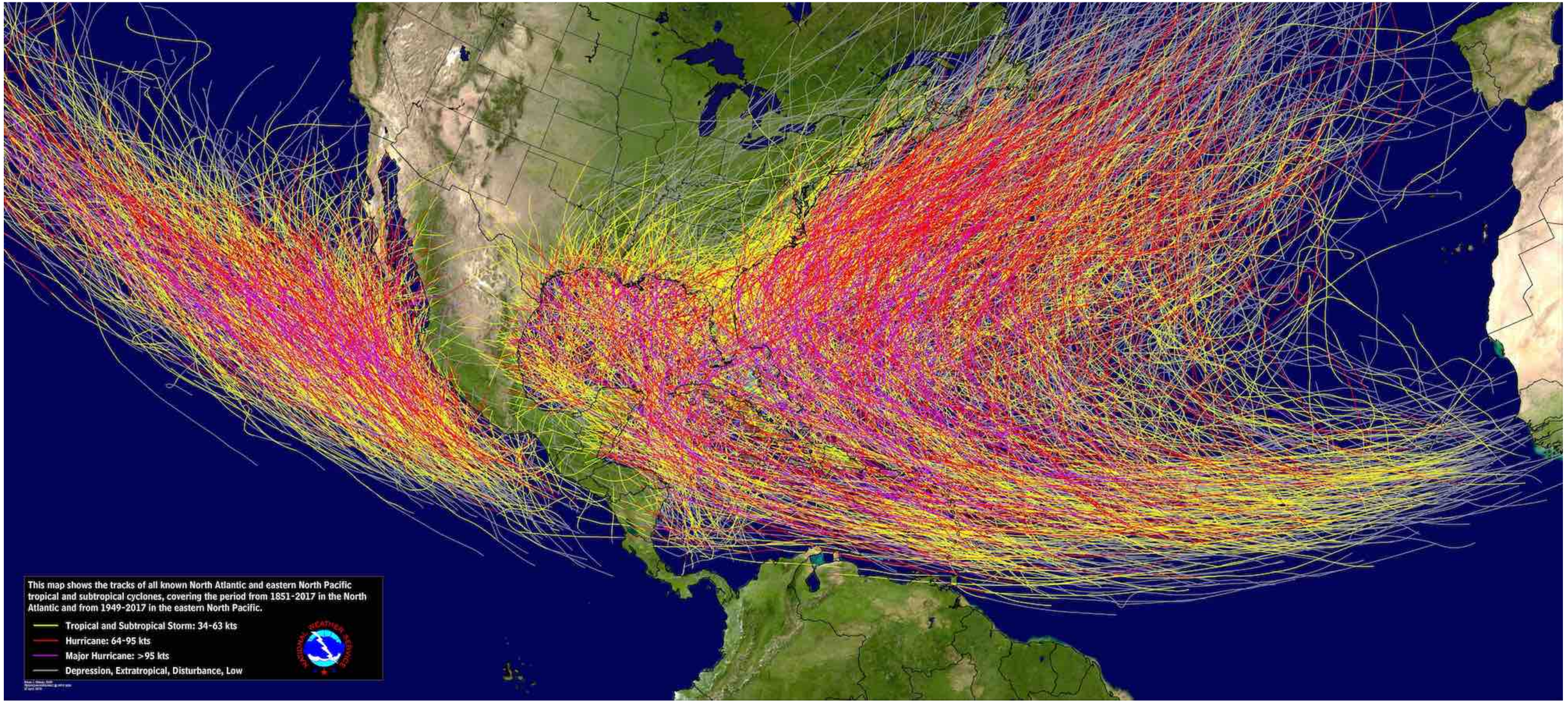
https://tinyurl.com/OCEAN215-Class5

Indexing using double brackets

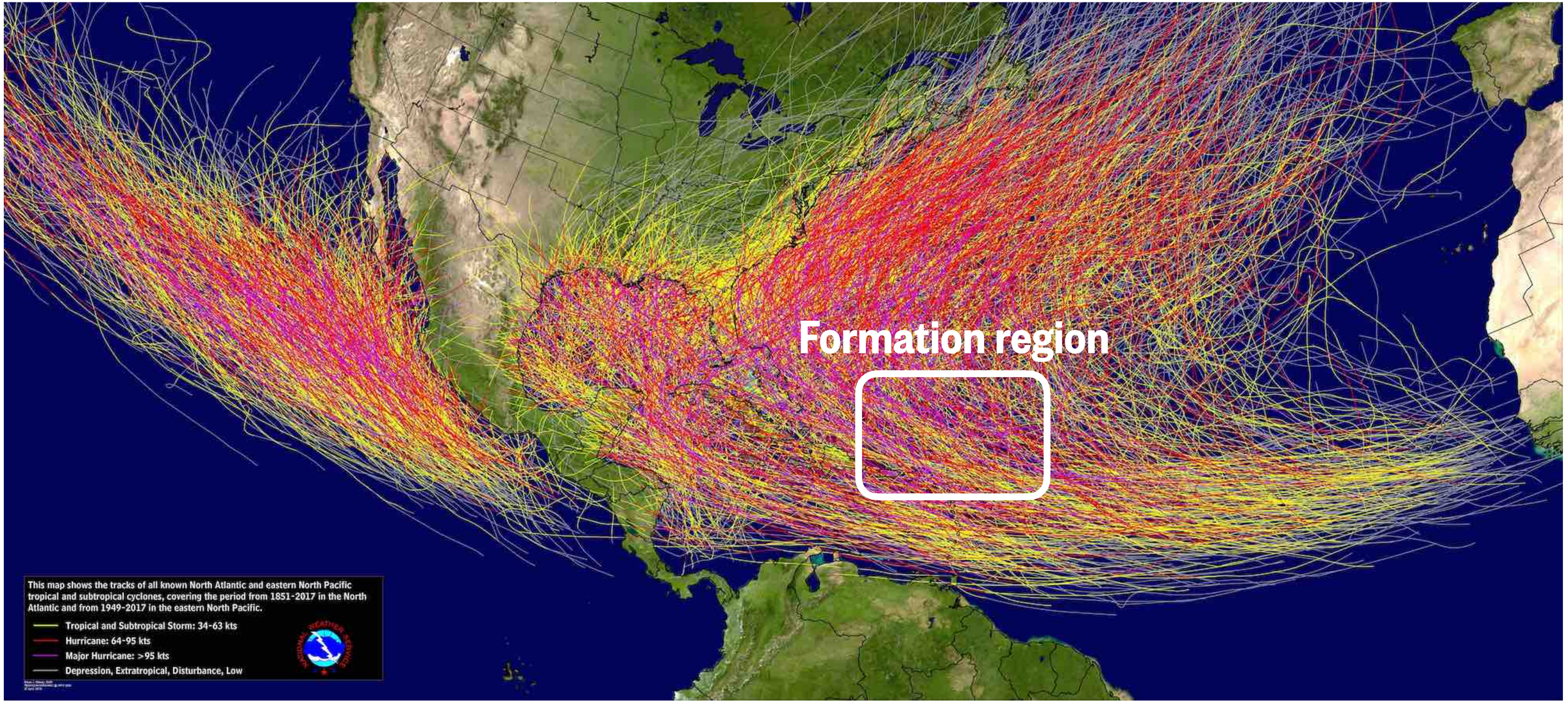
Google Doc with activities (also accessible from Canvas Modules or Google Drive folder):

https://tinyurl.com/OCEAN215-Class5

Coding activity: identifying hurricane season



Coding activity: identifying hurricane season



Coding activity: identifying hurricane season

Warm ocean water provides the energy for hurricanes to form. Almost 70 years ago, scientists realized that hurricanes can form only when the surface of the ocean is warmer than 79.7°F, or 26.5°C. Below this temperature, hurricanes will not form or will weaken rapidly.

I've downloaded sea surface temperatures (SSTs) for a location in the tropical Atlantic, northeast of Puerto Rico, where hurricanes tend to form. These are monthly values (January-December) for a typical, or "climatological," year.

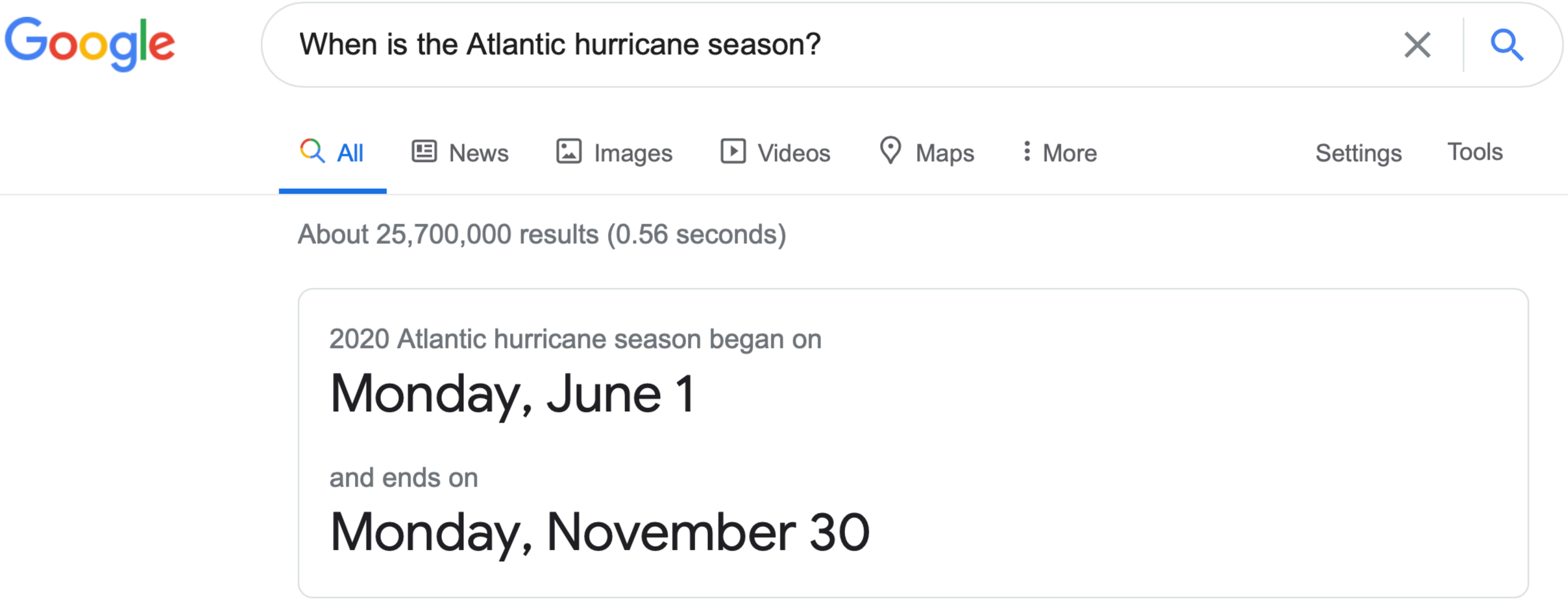
Use *for* loop(s) and *if* statement(s) to answer these questions:

- Question 1: What is the average SST over the 12 months? Round the answer to one decimal place using round().
- Question 2: How many months are favorable for hurricane formation?
- Question 3: What month is most favorable for hurricanes to form?
- Question 4: When does hurricane season start and end?

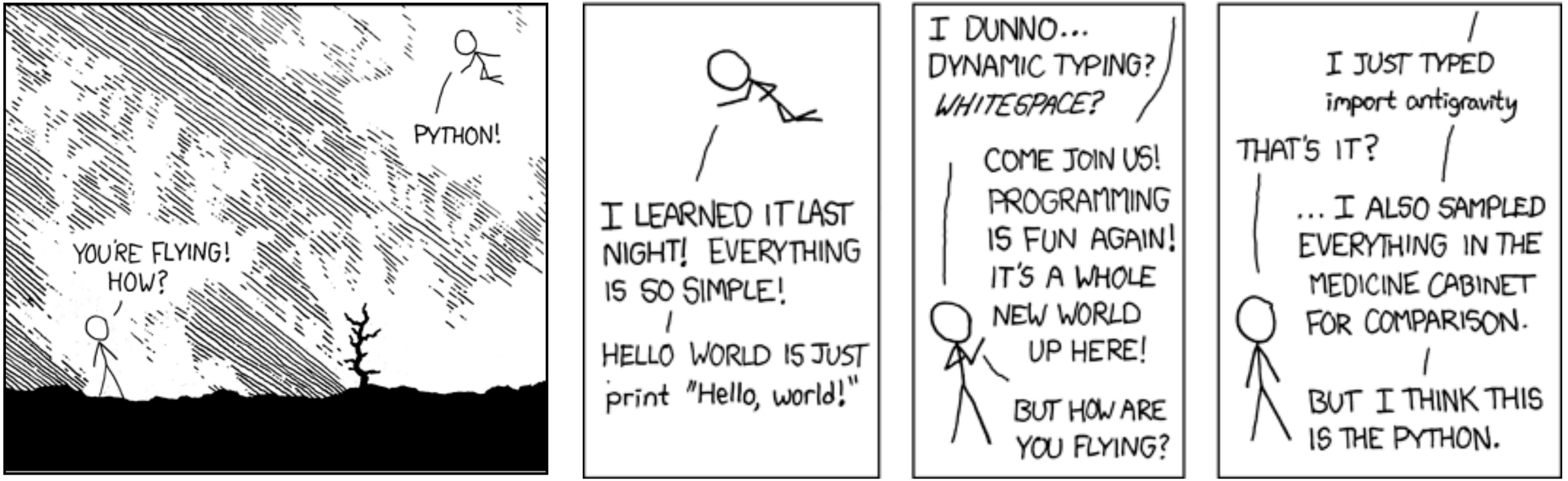
```
# Ocean temperatures at (20°N, 60°W)
#   from NOAA 5km SST climatology:
#   https://coralreefwatch.noaa.gov/product/5km/

months = ['January','February','March','April','May','June',
          'July','August','September','October','November','December']
sst_clim = [25.52,25.14,25.06,25.49,26.27,26.96,
            27.48,27.87,28.04,27.97,27.26,26.38]
```

Coding activity: identifying hurricane season



import antigravity



Source: XKCD

Why are NumPy arrays useful?

Google Doc with activities (also accessible from Canvas Modules or Google Drive folder):

https://tinyurl.com/OCEAN215-Class5

Review: arithmetic operations with arrays

Arithmetic operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponential
%	Remainder
//	Floor

Element-wise arithmetic between two or more arrays

```
1 a = np.array([1,2,3,4])
2 b = np.array([5,6,7,8])
3
4 print('a + b =',a + b)
5 print('a - b =',a - b)
6 print('a * b =',a * b)
```

```
a + b = [ 6  8 10 12]
a - b = [-4 -4 -4 -4]
a * b = [ 5 12 21 32]
```

Element-wise arithmetic with an array and a number

```
1 print('a + 10 =',a + 10)
2 print('10 * a =',10 * a)
3 print('a / 10 =',a / 10)
4 print('a**2 =',a**2)
```

```
a + 10 = [11 12 13 14]
10 * a = [10 20 30 40]
a / 10 = [0.1 0.2 0.3 0.4]
a**2 = [ 1  4  9 16]
```

Review: logical operations with arrays

Comparison operators

==	Equal
!=	Not equal
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to

Element-wise comparisons between two arrays or an array and a number

```
1 u = np.array([1,2,3,4])
2 v = np.array([0,2,4,6])
3
4 print(u == v)
5 print(u < v)
6 print(v != 0)
7 print(v <= 4)
```

```
[False  True False False]
[False False  True  True]
[False  True  True  True]
[ True  True  True False]
```

Instead of comparing Boolean arrays with `and`/`or`, use `np.logical_and()` and `np.logical_or()`

```
1 bool1 = np.array([True,False,True])
2 bool2 = np.array([True,False,False])
3
4 print(np.logical_and(bool1,bool2))
5 print(np.logical_or(bool1,bool2))
```

```
[ True False False]
[ True False  True]
```

Review: new indexing options with arrays

When you want to access certain value(s) in an array:

```
1 v = np.array([10,11,12,13])
2
3 print(v[3])
4
5 print(v[[2,3]])
6
7 print(v[v >= 12])
8
9 print(v[[False,False,True,True]])
```

Python prints:

```
13
[12 13]
[12 13]
[12 13]
```

Traditional list-style **single index**

Multiple indices retrieves multiple elements

Logical conditions also work…

… because they evaluate to **Boolean arrays**

When you want the indices of certain values in an array:

```
1 print(np.where(v >= 12))
2
3 print(np.where(v >= 12)[0])
```

```
(array([2, 3]),)
[2 3]
```

`np.where()` gives the indices at which a Boolean condition is satisfied…

… but you have to index into the result using `[0]`

Hurricane season, except with NumPy instead of loops!

Use NumPy functions to answer these questions:

- Question 1: What is the average SST over the 12 months?
- Question 2: How many months are favorable for hurricane formation?
- Question 3: What month is most favorable for hurricanes to form?
- Question 4: When does hurricane season start and end?

```
# Ocean temperatures at (20°N, 60°W)
#   from NOAA 5km SST climatology:
#   https://coralreefwatch.noaa.gov/product/5km/

months = ['January','February','March','April','May','June',
          'July','August','September','October','November','December']
sst_clim = [25.52,25.14,25.06,25.49,26.27,26.96,
            27.48,27.87,28.04,27.97,27.26,26.38]

# First step: convert the lists to arrays
#   (you can use the same variable names, or different ones)

```