PYTHON FOR DATA SCIENCE

UNIT 10

TUPLES & SETS

TOPICS TO BE COVERED

- What are tuples?
  - How are they related to lists?
  - Why use tuples (vs. using lists)

- Working with tuples
  - Creating tuples
  - Data retrieval from tuples

- Tuple Assignment
  - Assign multiple variables at once
  - Swapping values of variables

TUPLE BASICS

WHAT ARE TUPLES?

- Tuples are like lists that cannot be changed
- Recall: lists can be changed … i.e. lists are “mutable”
- Tuples cannot be changed … i.e. tuples are “immutable”

TUPLE BASICS

REASONS WHY / WHEN WE USE TUPLES? (INSTEAD OF USING OTHER DATA TYPES)

- Tuples use less space
- Tuples are faster than lists
  - You can loop / iterate through them faster
  - Your code is faster
- Tuples are good for things that won't change
  - Spatial dimension (x, y, z), months, days of the week
- Tuples protect your data
  - Tuples are immutable -- i.e. they can't be changed -- so you can't accidentally change one

TUPLE BASICS

TUPLES CONTAIN MULTIPLE VALUES

- In that regard, they are a lot like lists
- Tuples are written using parentheses … ( )
- Every value of a tuple has an associated index
- Tuple indexes start at 0

months = (‘Jan’, ‘Feb’, ‘Mar’, ‘Apr’, ‘May’, ‘June’, ‘July’, ‘Aug’, ‘Sept’, ‘Oct’, ‘Nov’, ‘Dec’)

index:	0	1	2	3	4	5	6	7	8	9	10	11
days:	Jan	Feb	Mar	Apr	May	June	July	Aug	Sept	Oct	Nov	Dec
index:	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

CREATING TUPLES

THERE ARE MULTIPLE WAYS TO CREATE TUPLES

- The simplest way to create a tuple is to enumerate items, separated by commas
- Another way to create a tuple is to enclose a group of items inside of parentheses
    - This approach is considered to be “best practice”

CREATING TUPLES

- **EXAMPLE: CREATING A TUPLE BY ENUMERATING ITEMS, SEPARATED BY COMMAS**

```
weekdays = 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday'
print(weekdays)
print(type(weekdays))
```

```
('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday')
<class 'tuple'>
```

CREATING TUPLES

- **EXAMPLE: CREATING A TUPLE BY ENCLOSING ITEMS INSIDE PARENTHESES**

```
months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months)
print(type(months))
```

```
('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
<class 'tuple'>
```

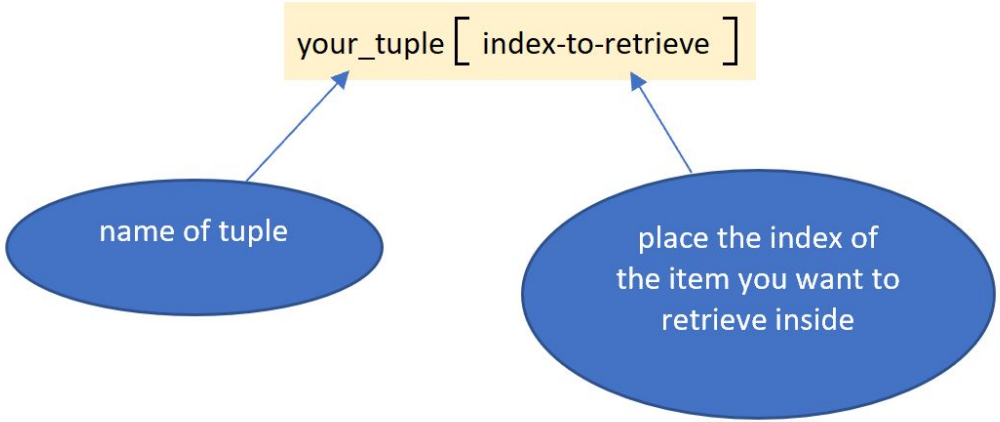
GETTING DATA FROM TUPLES

GETTING DATA FROM TUPLES IS LIKE GETTING DATA FROM OTHER SEQUENCES

- You can use the methods that you used with strings and lists
    - Indexing
    - Slicing

GETTING DATA FROM TUPLES

SYNTAX: HOW TO RETRIEVE SINGLE ITEMS FROM A TUPLE



GETTING DATA FROM TUPLES

EXAMPLE: HOW TO RETRIEVE SINGLE ITEMS FROM A TUPLE

- Using an index to retrieve a specific element

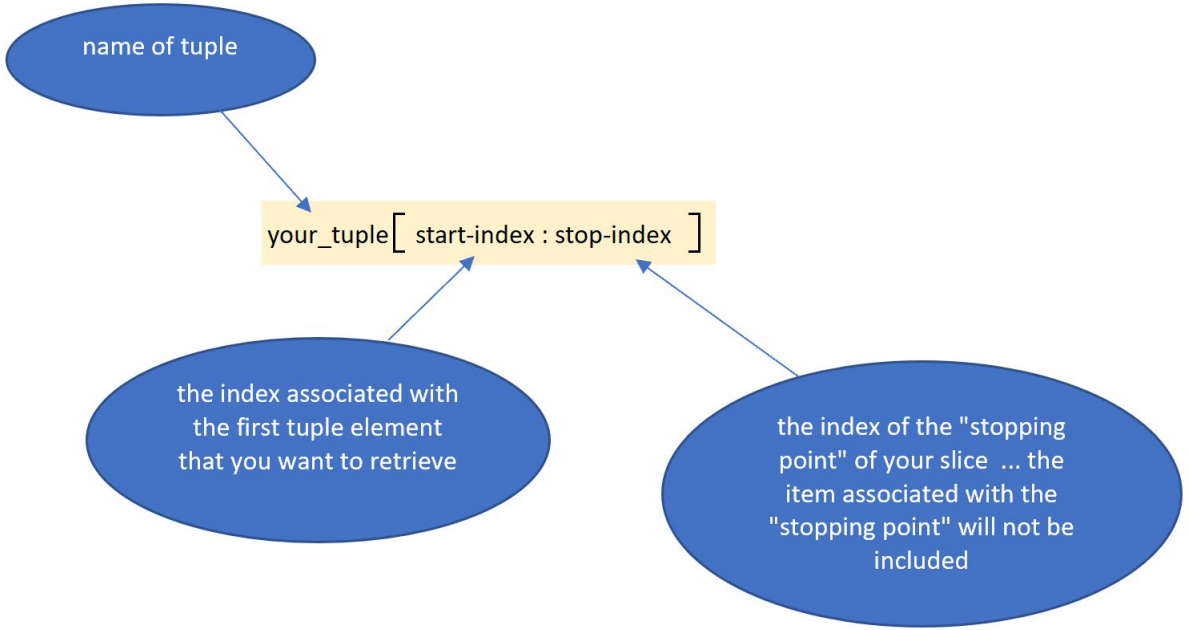
```
months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[0])
print(months[4])
print(months[-1])
print(months[-8])
```

```
Jan
May
Dec
May
```

GETTING DATA FROM TUPLES

HOW TO ACCESS A “SLICE” OF A TUPLE

- Using an index to retrieve a specific element



GETTING DATA FROM TUPLES

EXAMPLES: HOW TO ACCESS A SLICE OF A TUPLE

- Remember: the slice starts at the start index
- Remember: the slice goes up to the stop index, but excludes the stop index

```
# Retrieving a slice from a tuple

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[1:5])
```

```
('Feb', 'Mar', 'Apr', 'May')
```

```
# Retrieving a slice from a tuple ... faulty indexes

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[7:4])
```

```
()
```

GETTING DATA FROM TUPLES

EXAMPLES: HOW TO ACCESS A SLICE OF A TUPLE

```
# Retrieving a slice from a tuple ... no start index

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[6:])
```

```
('July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
```

```
# Retrieving a slice from a tuple ... no stop index

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[:3])
```

```
('Jan', 'Feb', 'Mar')
```

GETTING DATA FROM TUPLES

EXAMPLES: HOW TO ACCESS A SLICE OF A TUPLE

```
# Retrieving a slice from a tuple ... negative indexes

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[-7:-4])
```

```
('June', 'July', 'Aug')
```

```
# Retrieving a slice from a tuple ... faulty negative indexes

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[-3:-8])
```

```
()
```

GETTING DATA FROM TUPLES

EXAMPLES: HOW TO ACCESS A SLICE OF A TUPLE

```
# Retrieving a slice from a tuple ... negative indexes and no start index

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[:-10])
```

```
('Jan', 'Feb')
```

```
# Retrieving a slice from a tuple ... negative indexes and no stop index

months = ('Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
print(months[-9:])
```

```
('Apr', 'May', 'June', 'July', 'Aug', 'Sept', 'Oct', 'Nov', 'Dec')
```

TUPLE ASSIGNMENT

YOU CAN USE “TUPLE ASSIGNMENT” TO ASSIGN VALUES TO VARIABLES

- You can use tuples to assign values to multiple variables in one line of code
- Sometimes this is called “tuple unpacking”

TUPLE ASSIGNMENT

EXAMPLE #1: TUPLE ASSIGNMENT

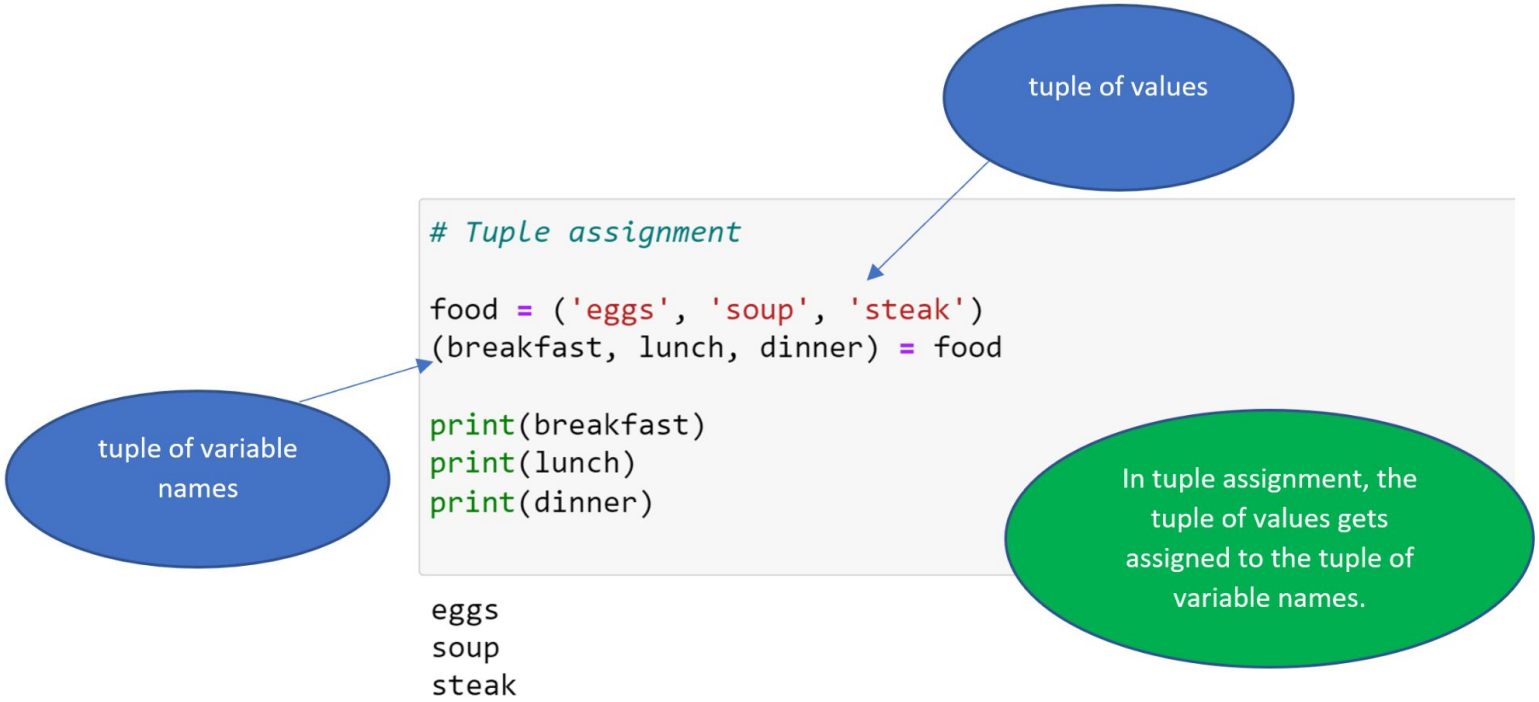
```
# Tuple assignment

food = ('eggs', 'soup', 'steak')      # Creating a tuple with 3 values
(breakfast, lunch, dinner) = food     # Creating a second tuple with 3 variable names and the values
                                      # from the first tuple'eggs', 'soup', 'steak' are being assigned
                                      # to those 3 variables

print(breakfast)
print(lunch)
print(dinner)
```

```
eggs
soup
steak
```

Python Tutor -- Visualize Code Execution

TUPLE ASSIGNMENT

EXAMPLE #1: TUPLE ASSIGNMENT CONTINUED



TUPLE ASSIGNMENT

EXAMPLE #2: TUPLE ASSIGNMENT

```
# Tuple assignment

record = (10, 6, 1)
(wins, losses, ties) = record

print(f'{wins} - {losses} - {ties}')
```

```
10 - 6 - 1
```

Python Tutor -- Visualize Code Execution

TUPLE ASSIGNMENT

YOU CAN USE TUPLE ASSIGNMENT TO SWAP VALUES BETWEEN VARIABLES

```
# Using tuple assignment to swap values between variables

x = 10
y = 15
z = 20

print(x, y, z)

(x, y, z) = (y, z, x)

print(x, y, z)
```

```
10 15 20
15 20 10
```

Python Tutor -- Visualize Code Execution

TUPLE ASSIGNMENT

YOU CAN USE TUPLE ASSIGNMENT TO SWAP VALUES BETWEEN VARIABLE

```
# Using tuple assignment to swap values between variables

first_letter = 'a'
second_letter = 'b'
third_letter = 'c'

print(first_letter, second_letter, third_letter)

first_letter, second_letter, third_letter = (third_letter, first_letter, second_letter)
print(first_letter, second_letter, third_letter)
```

```
a b c
c a b
```

Python Tutor -- Visualize Code Execution