# CME 211 Lecture 4: Python containers

## Containers

- *Containers* are objects that contain one or more other objects
- *Containers* are sometimes called "collections" or "data structures"
- In lecture 3, we introduced the Python `list` container
- Today we will see Dictionaries, Sets, and Tuples

### Sequential

**Sequential containers** store data items in a specified order. Think elements of a vector, names in a list of people that you want to invite to your birthday party. The most fundamental Python data type for this is called a `list`. Later in the course we will learn about containers that are more appropriate (and faster) for numerical data that come from NumPy.

We have seen lists in lecture 3:

```
my_list = [4, 8.99, 'list item', ["sub","list"]]
print(my_list)
```

Python has another built-in sequential container called a `tuple`. Tuples are a lot like lists, but the elements cannot be modified after the tuple is created.

```
my_tup = (1,2,"str",99.99)
print(my_tup)
print(my_tup[1])
```

```
my_tup[1] = "new item"
```

Tuples are said to be immutable. We'll see why this is important later.

### Associative

**Associative containers** store data, organized by a unique *key*. Think of a dictionary of word definitions. They unique *key* is a word, the value associated with the key is the definition. In Python, this is represented with the built-in `dict` or Dictionary type. You will soon learn the greatness of dictionaries in Python.

```
emails = dict()
emails['andreas'] = 'andreas@mail.com'
emails['nick'] = 'nwh@stanford.net'
print(emails)
```

Access a single element:

```
emails['nick']
```

What if a key is not there?

```
emails['cindy']
```

### Sets / Unique

**Set containers** store unique data items. They are related to dictionaries, because dictionaries require the keys to be unique.

```
dinos = set()
dinos.add('triceratops')
dinos.add('t-rex')
dinos.add('raptor')
print(dinos)
dinos.add('pterodactyl')
dinos.add('t-rex')
print(dinos)
```