Announcements

- Farhan’s tutoring today: Dan will substitute this week.
- Try to get help from me and tutors
- Reading assignment for this slide set:
  - Chapter 15 of Downey
- Break around 10:15am
Plan

- We will skip tuples for now and come back to it when we need to
- Study basics of objects next
- JavaScript afterwards
Objects

- So far we have been doing procedural programming, namely, programming with procedures (aka functions)
- Python also supports objects
- Let’s see the basics of creating and using objects
Built-in Types vs. User-defined Types

- We have studied most of the built-in types
  - Started out with primitive data types
    - `int`, `float`, `bool`
  - Compound data types
    - Strings, lists, dictionaries

- We can create our own types, i.e., user-defined types (by user we mean a Python programmer)
  - `Point`, `Student`, `Computer`, `Marker`, `Book`, etc.
Class

- In Python we create a new user-defined type by defining a `class`.
- Once we create a class, which becomes a user-defined type, we can create instances (aka `objects`) of that class (type).
- For example, when we create a class named `Point` to represent a point in 2-dimensional plane, we can create point instances such as `p1` and `p2` to be used to represent two corners of a rectangle object.
- See `point1.py` for an example.
- Using `Point` object(s) we could define a `Rectangle` class and create rectangle objects but I will leave that as an exercise for now.
- This much on objects will be needed to read code on objects as we do web programming. More on objects will be added as needed.
Lists vs. dictionaries vs. objects

- Suppose we have three points with their x and y values to manage: (1,2), (1,4), (3, 4)
- As a list of lists
  - `[[1,2], [1, 4], [3,4]]`
  - Need to remember that index 0 represents x, index 1 represents y
- As a list of dictionaries:
  - `[{‘x’: 1, ‘y’: 2}, {‘x’: 1, ‘y’: 4}, {‘x’: 3, ‘y’: 4}]`
  - x and y have to be repeated in each dictionary
- As a list of objects (as we will see soon)
  - `[p1, p2, p3]` with `p1`, `p2`, and `p3` defined as three point objects with appropriate x- and y-values
  - `p1.x`, `p1.y` will access the x and y values in `p1`
  - x and y are not stored with the x- and y-values in memory
  - They are implicit with the Point class defined with the attribute names
Examples

- See `point1.py`
- See `point2.py`
- See `points.py` (optional)
- Read chapters 15, 16, and 17 of the text as a reference if you need