Data Structures: Linked Lists
Why “Data Structures”? (Next Few lectures)

- Data Structures
  - OOP helps us organize our programs
  - Data Structures help us organize our data!
  - You already know lists and dictionaries!
  - We’ll see two new ones today
- Enjoy this stuff? Take 61B!
- Find it challenging? Don’t worry! It’s a different way of thinking.
Linked Lists
Data Structures

• A data structure is a way to organize or *model* a bunch of independent pieces of data.
  – Lists (arrays)
  – Dictionaries
  – Tuples

• A class, on its own, is *not* necessarily a data structure, it represents a new data type.
  – a “car” or a “person” is an instance of that data type.
  – Lists, Dicts, etc are also data types; their goal is to organize other data.

• These are common patterns that can be used to solve a wide variety of problems.

• Sometimes we’re giving structure to make it easier as a programmer, sometimes we’re trying to be fast or efficient. (Next lecture!)
Linked Lists

• A Recursive List, sometimes called a "rlist"
• Linked lists contain other linked lists
• A series of items with two pieces:
  – A value, usually called "first"
  – A “pointer” to the rest of the items in the list.

![Diagram of linked lists]

• We’ll use a very small Python class “Link” to model this.
• `Link(12, Link(99, Link(37, Link.empty)))`
Recursion Is Implicit

self.rest
Demo – See the Notebook
Why are linked lists useful?

• Honestly, a list() is easier most of the time
  – Python handles all the hard details!

• In terms of efficiency: Linked list make it fast to move items around, inserts and deletes.
  – But they are slower to finding any single item.

• In Ants Project: You’ll see a list of `Place` objects which are linked together via an entrance and an exit.