Outline

What is a data structure?
Strings
Lists and Arrays
Dictionaries
Trees
Graphs
Classes
A *data structure* is a way of organizing a collection of data. Examples include strings, lists, dictionaries, trees, and graphs.

An *abstract data type* (ADT) is a data structure together with a set of methods for operating on that data. An example is a *stack*, which consists of a list plus two operations: *push* (putting an element onto the “top” of the stack) and *pop* (taking an element off the top of the stack).
Strings

A string is a sequence of characters.
It’s the basic representation for text.

```python
>>> s = "This is some text."
>>> s.upper()
"THIS IS SOME TEXT."
>>> s.lower()
"this is some text."
>>> s
"This is some text."
>>> len(s)
18
>>> s.index('s')
3
>>> s.index('s', 4)
6
```
A list is a sequence of data objects.

In Python, a list is written using square brackets.

```python
>>> student = ["Name", "Cindy", "Age", 20]
>>> student[0]
"Name"
>>> student[0]
"Cindy"
>>> student[-1]
20
>>> len(student)
4
>>> student.reverse()
[20, "Age", "Cindy", "Name"]
>>> student[1:3]
["Cindy", "Age", 20]
```
An one-dimensional array is also a sequence of data objects.

In Python, a one-dimensional array is often just a list.

```python
>>> a = [2, 0, 6, 5, 4, 3, 1, 6, 9, 5]
>>> a[0]
2
```

In Python, a two-dimensional array is often just a list of lists.

```python
>>> b = [['X', 'O', 'X'], ['O', 'O', 'X'], ['X', 'X', 'O']]
>>> b[0][0]
'X'
```

```
X O X
O O X
X X O
```
A dictionary is a collection of key/value pairs.

In Python, a dictionary is written using curly brackets.

```python
>>> style = {"Monet": "impressionist", "Rembrandt": "Baroque"}
>>> style["Monet"]
"impressionist"
```

Dictionary entries are accessed with square-bracketed expressions, like lists, except that the subscripts are not limited to integers.

They are more flexible, but a little less efficient than arrays.
A tree organizes data into a hierarchy. Each data element is called a node. There is a single top node called the root. Each node can have one or more children. There is a unique parent for each node, except that the root has no parent.
A tree can be represented in Python as a list of lists.

Tree = ['President',
        ['Vice Pres., sales',
         ['Mgr., foreign sales'],
         ['Mgr., domestic sales']],
        ['Vice Pres., R&D',
         ['Head of image processing']]]
A tree could also be represented as a dictionary of node/parent pairs:

```
ParentRelation = {
    "President": None,
    "Vice Pres., sales": "President",
    "Mgr., foreign sales": "Vice Pres., sales",
    "Mgr., domestic sales": "Vice Pres., sales",
    "Vice Pres., R&D": "President",
    "Head of image processing": "Vice Pres., R&D"
}
```
Trees (cont.)

Tree representations of an image:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0,7</td>
<td>1,7</td>
<td>7,7</td>
</tr>
<tr>
<td>0,1</td>
<td>1,1</td>
<td>7,1</td>
</tr>
<tr>
<td>0,0</td>
<td>1,0</td>
<td>7,0</td>
</tr>
</tbody>
</table>

- **Table:**
  - Rows and columns represent pixel coordinates.
  - Each cell contains the detail of the pixel at that coordinate.

- **Diagram:**
  - Represents the hierarchical structure of the image.
  - Each node represents a pixel or a set of pixels.
  - The structure is built from the bottom up, with each level representing a set of pixels from the previous level.
Trees (cont.)

Tree representations of an image:

- Diagram of a scene divided into various parts: sky, sun, house, lawn, man, roof, wall, gable, head, body, arm, hand, leg, foot.
- Corresponding image showing a house, a person waving, and a window.
Tree representations of an image:

Image organized into quadrants and subquadrants

Representation as a quadtree
A graph is a network of nodes and their interconnections. It consists of a set of vertices and a set of edges.

\[ G = (V, E) \]
\[ V = \{A, B, C, D\} \]
\[ E = \{(A, C), (B, C), (C, D), (D, B)\} \]

Graphs can represent airline networks, city street maps, communication networks, social networks of friends in Facebook, information about what is next to what in an image, or what is connected to what in the human body.
Graphs (continued)

Digital image

Region-adjacency graph

```
2 2 7 7 7 7 7 7
2 2 2 7 7 5 7 7
4 2 2 1 1 5 5 5
4 4 1 1 1 1 5 5
4 4 1 1 1 1 3 3
4 4 4 1 1 3 3 3
4 4 6 6 3 3 3 3
6 6 6 6 3 3 3 3
```
A class is a way of representing an abstract data type (data structure + set of methods for operating on the data).

class artWork:
    def __init__(self, artist_name, title, medium):
        self.artist = artist_name
        self.title = title
        self.medium = medium
    def dimension(self):
        if self.medium == 'moving sculpture':
            return 4
        if self.medium == 'sculpture' or
            self.medium == 'film':
            return 3
        else: return 2
    def report(self):
        return '"'+self.title+'"
        ('+self.medium+' by '+self.artist+' having dimension '+str(self.dimension())+'"}
Classes

Once a class has been defined, many instances of the class can be constructed and manipulated. Here’s just one instance.

```python
>>> mona = artWork ("Leonardo da Vinci", "La Joconde", "oil painting")
>>> mona.dimension()
2
>>> mona.location = "Louvre" # new fields can be added later .
>>> print mona.report()
"La Joconde"
(oil painting by Leonardo da Vinci having dimension 2)
```
Classes are the basis for a style of programming called “object-oriented programming.”

Object-oriented programming has been the leading style for software engineering for the past 25 years.