Lecture Notes
Week 11
“Everyday Python”
CSC111 Spring 2015

Dominique Thiébaut
dthiebaut@smith.edu
Lists of Lists
(Chapter 11—Designing with Lists and Classes)
Two Types of Lists

Useful List Operations

Two Approaches to Filtering Data

Examples
Two Types of Lists

• someList1 = [1, “hello”, 6.5]

• someList2 = (1, “hello”, 6.5)
Two Types of Lists

- `someList1 = [ 1, "hello", 6.5 ]`  
  mutable

- `someList2 = ( 1, "hello", 6.5 )`  
  immutable
someList1 = [ 1, 2, 3 ]

for i in range( 20, 30 ):
    someList1.append( i )

someList2 = ( 1, 2, 3 )

for i in range( 20, 30 ):
    someList2.append( i )
someList1 = [ 1, 2, 3 ]

for i in range( 20, 30 ):
    someList1.append( i )

someList2 = ( 1, 2, 3 )

for i in range( 20, 30 ):
    someList2.append( i )
Notation

```python
someList1 = [ 1, 2, 3 ]
for i in range( 20, 30 ):
    someList1.append( i )

someList2 = ( 1, 2, 3 )
for i in range( 20, 30 ):
    someList2.append( i )
```

Tuple
Two Types of Lists

Useful List Operations

Two Approaches to Filtering Data

Examples
Useful List Operations

```python
Python 3.1.1 (r311:74543, Aug 24 2009, 18:44:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> L = [3, 10, 3, 5, 1, -1, 0, 6]
>>> L
[3, 10, 3, 5, 1, -1, 0, 6]
>>> L.sort()
>>> L
[-1, 0, 1, 3, 3, 5, 6, 10]
>>> L.reverse()
>>> L
[10, 6, 5, 3, 3, 1, 0, -1]
>>> L[0]
10
>>> L[0:3]
[10, 6, 5]
>>> L[-3:]
[1, 0, -1]
>>> S = set(L)
>>> S
{0, 1, 3, 5, 6, 10, -1}
>>> L = list(S)
>>> L
[0, 1, 3, 5, 6, 10, -1]
>>> |
```
Useful List Operations

Sorting Tuples

```python
Python 3.1.1 (r311:74543, Aug 24 2009, 18:44:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> L = [ (10, "Smith"), (1, "Amherst"), (3, "Umass"), (5, "Hampshire") ]
>>> L
[(10, 'Smith'), (1, 'Amherst'), (3, 'Umass'), (5, 'Hampshire')]
>>> L.sort()
>>> L
[(1, 'Amherst'), (3, 'Umass'), (5, 'Hampshire'), (10, 'Smith')]

>>> L2 = [ ("Smith", 10), ("Amherst", 1), ("Umass", 3 ), ("Hampshire", 5 ) ]
>>> L2.sort()
>>> L2
[('Amherst', 1), ('Hampshire', 5), ('Smith', 10), ('Umass', 3)]
```
Two Types of Lists

Useful List Operations

Two Approaches to Filtering Data

Examples
Two Types of Filtering Problems...
The Problem at Hand

Textual Info.
The Problem at Hand

OPTION 1:
We are only interested in the red information, and only the smaller or larger items…
The Problem at Hand

Textual Info.

[ (         ,          ,          ),
  (         ,          ,          ),
  (         ,          ,          ),
  ...
  (         ,          ,          ) ]

[ (         ,          ,          ),
  (         ,          ,          ),
  (         ,          ,          ),
  ...
  (         ,          ,          ) ]
The Problem at Hand

[ (         ,          ,          ),
 (         ,          ,          ),
 (         ,          ,          ),
 ...  
 (         ,          ,          ) ]

SORT

[ (         ,          ,          ),
 (         ,          ,          ),
 (         ,          ,          ),
 ...  
 (         ,          ,          ) ]
OPTION 2:
We are only interested in the lines that contain the red information
The Problem at Hand

Textual Info.

FILTER

\[
\begin{align*}
\{ & (\quad, \quad, \quad ), \\
& (\quad, \quad, \quad ), \\
& (\quad, \quad, \quad ), \\
& \ldots \\
& (\quad, \quad, \quad ) \}
\end{align*}
\]
Two Types of Lists

Useful List Operations

Two Approaches to Filtering Data

Examples
Example 1
10 **Rainiest** Months In **Cambridge**, U.K.?  

http://cs.smith.edu/~dthiebaut/UKTemperatures/
Example 2
List of email addresses for students enrolled in several classes.

Need a list of all Smith students without duplicates and a list of all 5-College students without duplicates
Example 3
<table>
<thead>
<tr>
<th>U.S. Presidents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>text=</strong>&quot;Presidency, President, Took office, Left office, Party, Home State**&quot;</td>
</tr>
<tr>
<td><strong>1. George Washington, 30/04/1789, 4/03/1797, Independent, Virginia</strong></td>
</tr>
<tr>
<td><strong>2. John Adams, 4/03/1797, 4/03/1801, Federalist, Massachusetts</strong></td>
</tr>
<tr>
<td><strong>3. Thomas Jefferson, 4/03/1801, 4/03/1809, Democratic-Republican, Virginia</strong></td>
</tr>
<tr>
<td><strong>4. James Madison, 4/03/1809, 4/03/1817, Democratic-Republican, Virginia</strong></td>
</tr>
<tr>
<td><strong>5. James Monroe, 4/03/1817, 4/03/1825, Democratic-Republican, Virginia</strong></td>
</tr>
<tr>
<td><strong>6. John Quincy Adams, 4/03/1825, 4/03/1829, Democratic-Republican/National Republican, Massachusetts</strong></td>
</tr>
<tr>
<td><strong>7. Andrew Jackson, 4/03/1829, 4/03/1837, Democratic, Tennessee</strong></td>
</tr>
<tr>
<td><strong>8. Martin Van Buren, 4/03/1837, 4/03/1841, Democratic, New York</strong></td>
</tr>
<tr>
<td><strong>9. William Henry Harrison, 4/03/1841, 4/04/1841, Whig, Ohio</strong></td>
</tr>
<tr>
<td><strong>10. John Tyler, 4/04/1841, 4/03/1845, Whig, Virginia</strong></td>
</tr>
<tr>
<td><strong>11. James K. Polk, 4/03/1845, 4/03/1849, Democratic, Tennessee</strong></td>
</tr>
<tr>
<td><strong>12. Zachary Taylor, 4/03/1849, 9/07/1850, Whig, Louisiana</strong></td>
</tr>
<tr>
<td><strong>13. Millard Fillmore, 9/07/1850, 4/03/1853, Whig, New York</strong></td>
</tr>
<tr>
<td><strong>14. Franklin Pierce, 4/03/1853, 4/03/1857, Democratic, New Hampshire</strong></td>
</tr>
<tr>
<td><strong>15. James Buchanan, 4/03/1857, 4/03/1861, Democratic, Pennsylvania</strong></td>
</tr>
<tr>
<td><strong>16. Abraham Lincoln, 4/03/1861, 15/04/1865, Republican/National Union, Illinois</strong></td>
</tr>
<tr>
<td><strong>17. Andrew Johnson, 15/04/1865, 4/03/1869, Democratic/National Union, Tennessee</strong></td>
</tr>
<tr>
<td><strong>18. Ulysses S. Grant, 4/03/1869, 4/03/1877, Republican, Ohio</strong></td>
</tr>
<tr>
<td><strong>19. Rutherford B. Hayes, 4/03/1877, 4/03/1881, Republican, Ohio</strong></td>
</tr>
<tr>
<td><strong>20. James A. Garfield, 4/03/1881, 19/09/1881, Republican, Ohio</strong></td>
</tr>
<tr>
<td><strong>21. Chester A. Arthur, 19/09/1881, 4/03/1885, Republican, New York</strong></td>
</tr>
<tr>
<td><strong>22. Grover Cleveland, 4/03/1885, 4/03/1889, Democratic, New York</strong></td>
</tr>
<tr>
<td><strong>23. Benjamin Harrison, 4/03/1889, 4/03/1893, Republican, Indiana</strong></td>
</tr>
<tr>
<td><strong>24. Grover Cleveland, 4/03/1893, 4/03/1897, Democratic, New York</strong></td>
</tr>
<tr>
<td><strong>25. William McKinley, 4/03/1897, 14/9/1901, Republican, Ohio</strong></td>
</tr>
<tr>
<td><strong>27. William Howard Taft, 4/3/1909, 4/03/1913, Republican, Ohio</strong></td>
</tr>
<tr>
<td><strong>28. Woodrow Wilson, 4/03/1913, 4/03/1921, Democratic, New Jersey</strong></td>
</tr>
<tr>
<td><strong>29. Warren G. Harding, 4/03/1921, 2/8/1923, Republican, Ohio</strong></td>
</tr>
<tr>
<td><strong>30. Calvin Coolidge, 2/8/1923, 4/03/1929, Republican, Massachusetts</strong></td>
</tr>
<tr>
<td><strong>31. Herbert Hoover, 4/03/1929, 4/03/1933, Republican, Iowa</strong></td>
</tr>
<tr>
<td><strong>32. Franklin D. Roosevelt, 4/03/1933, 12/4/1945, Democratic, New York</strong></td>
</tr>
<tr>
<td><strong>33. Harry S. Truman, 12/4/1945, 20/01/1953, Democratic, Missouri</strong></td>
</tr>
<tr>
<td><strong>34. Dwight D. Eisenhower, 20/01/1953, 20/01/1961, Republican, Texas</strong></td>
</tr>
<tr>
<td><strong>40. Ronald Reagan, 20/01/1981, 20/01/1989, Republican, California</strong></td>
</tr>
<tr>
<td><strong>42. Bill Clinton, 20/01/1993, 20/01/2001, Democratic, Arkansas</strong></td>
</tr>
</tbody>
</table>
| **43. George W. Bush, 20/01/2001, 20/01/2009, Republican, Texas***"
We stopped here last time...
Class Inheritance
Class Inheritance
The Idea...
Car class
- constructor
- draw
- move
- undraw
- changeColor
- etc...
Car class
- constructor
- draw
- move
- undraw
- etc...

Wanted with same features
Car class
- constructor
- draw
- move
- undraw
- changeColor
- etc...

Wanted with same features

Wanted with same features
• We *could* write complete new classes for the two new car shapes…

• But instead, we can **save code, save time, save debugging aggravation**, by reusing the original car class.
• We could write complete new classes for the two new car shapes…

• But instead, we can save code, save time, save debugging aggravation, by reusing the original Car class.

• We will derive new classes from the Car class. The new classes will be derived from it, and will inherit all the member variables and methods.

• The Car class will become the super class.
Another Way to Look at Inheritance
Another Way to Look at Inheritance
Coding Example
class Car:
    """A default simple car class. Sets the color to "Yellow" ""
    def __init__( self, rp):
        """builds a car 180 pixels long on a reference point""
        x = rp.getX()
y = rp.getY()
        length = 180  
        # geometry of the car
        height = length // 3
        firstQuarter = length//4
        thirdQuarter = length * 3 // 4
        p2 = Point( x+length, y+height )
        self.body = Rectangle( rp, p2 )
        self.body.setFill( "yellow" )
        self.w1  = Wheel( Point( x+firstQuarter, y+height ), 20,10 )
        self.w2  = Wheel( Point( x+thirdQuarter, y+height ), 20,10 )

    def draw( self, win ) :
        """ draws the body and wheels on the window""
        self.body.draw( win )
        self.w1.draw( win )
        self.w2.draw( win )

    def move( self, dx, dy ) :
        """ moves the body and wheels some delta x and y""
        self.body.move( dx, dy )
        self.w1.move( dx, dy )
        self.w2.move( dx, dy )

def main():
    # open the window
    win = GraphWin( "CSC111 Inheritance Demo", WIDTH, HEIGHT )

car = Car( Point( 100,250 ) )
car.draw( win )

win.getMouse()
win.close()

def main():
```python
class Car:
    """A default simple car class. Sets the color to "Yellow" ""
    def __init__( self, rp ):
        """builds a car 180 pixels long on a reference point""
        x = rp.getX()
        y = rp.getY()
        length = 180
        height = length // 3
        firstQuarter = length//4
        thirdQuarter = length * 3 // 4
        p2 = Point( x+length, y+height )
        self.body = Rectangle( rp, p2 )
        self.body.setFill( "yellow" )
        self.w1 = Wheel( Point( x+firstQuarter, y+height ), 20,10 )
        self.w2 = Wheel( Point( x+thirdQuarter, y+height ), 20,10 )

def draw( self, win ):
    """draws the body and wheels on the window""
    self.body.draw( win )
    self.w1.draw( win )
    self.w2.draw( win )

def move( self, dx, dy ) :
    """moves the body and wheels some delta x and y""
    self.body.move( dx, dy )
    self.w1.move( dx, dy )
    self.w2.move( dx, dy )

class ColoredCar( Car ):
    """Class derived from Car. Allows resetting of body color""
    def setColor( self, color ):
        """sets the color of the body""
        self.body.setFill( color )

def main():
    # open the window
    win = GraphWin( "CSC111 Inheritance Demo", WIDTH, HEIGHT )
    car2 = ColoredCar( Point( 200, 100 ) )
    car2.setColor( "lightgreen" )
    car2.draw( win )
```

The Derived Class
Creating a **Truck**
Derived from a **Car**
Two Different Syntaxes

```python
def draw(self, win):
    super().draw(win)
    self.top.draw(win)
```

```python
def draw(self, win):
    Car.draw(self, win)
    self.top.draw(win)
```
Two Different Syntaxes

```python
def draw( self, win ):  
    super().draw( win ) 
    self.top.draw( win )
```

```python
def draw( self, win ):  
    Car.draw( self, win ) 
    self.top.draw( win )
```

Cleaner code
Graphics Libraries Are Built on Inheritance:

Hierarchy of Classes
The Essential Qt 3.0 Class Hierarchy
Coding Exercise

• Create an **Animal** class (name, dob, vaccinated, tattooed)

• Create a sub-class of Animal: **Cow** (milkProduction)

• Create **Dog** a sub-class of Animal (trained, hospital, seeingDog, rescue)