PYTHON FUNCTIONS AND BUILT-IN DATA TYPES

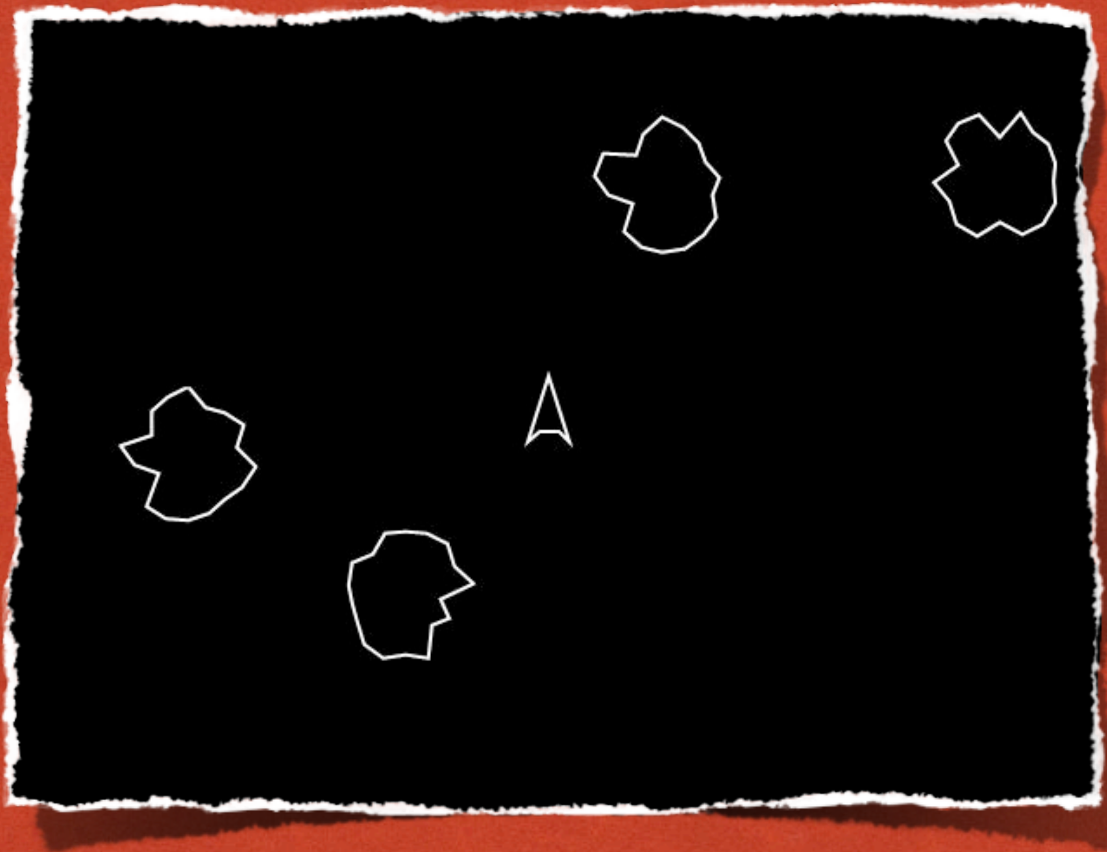
Curt Clifton
Rose-Hulman Institute of Technology

ANNOUNCEMENTS

- Homework 1 due now
- Homework 2 due start of class Thursday
  - Read through it soon!
  - I suspect you might have questions about the Haar wavelet problem

TODAY’S PLAN

- Look at some animations
- Highlight key “Pythonic” ideas from reading
- See one way to write unit tests for Python
- Language brainstorm



SAMPLE ANIMATIONS

WHO WANTS TO SHARE?

SOME COOL “PYTHONIC” FEATURES

- Subscripting and slicing lists (and strings)
- Formal parameters
  - Default arguments
  - Keyword arguments
- Docstrings
- Functions on lists
- Multiple assignment
- Dictionaries

Q1

SUBSCRIPTING AND SLICING

```
my_list = ["I'm", 'a', "lumberjack", 42]

print my_list[0]
print my_list[1:]
print my_list[-1]
print my_list[1:-1]
print my_list[0][-1]

my_list[0] = "You're"
print my_list[:-1]
my_list[2:3] = ['dead', 'parrot']
print my_list
print ' '.join(map(str,my_list))
```

Slicing

Assignment to a slice

The *str* function converts its argument to a string

Q2

DEFAULT ARGUMENTS

```
def complain(complaint = 'This is a dead parrot'):
    print "Customer:", complaint

complain()
complain("If you hadn't nailed 'im to the perch, he'd be pushin'\
 up daisies!")

def mutable_weirdness(n, l=[]):
    l.append(n)
    print l

mutable_weirdness(4, [1,2,3])
mutable_weirdness(1)
mutable_weirdness(2)
```

Default argument value

Line continuation

Q3

KEYWORD ARGUMENTS, DOCSTRINGS

- When a function has several parameters with default values, you can use *keyword arguments* to just give a few values

```
def converse(complaint = 'Bereft of life, he rests in piece',
             response = "He's pinnin' for the fjords"):
    """Conducts a short conversation.

    Conducts a short conversation between a complaining
    customer and a shopkeeper.
    """
    print "Customer:", complaint
    print "Shopkeeper:", response

converse(response="There, he moved!")
help(converse)
print converse.__doc__
```

Docstring

Keyword argument

Docstring uses

Q4–5

“CARTOON” OF THE DAY

- http://www.youtube.com/watch?v=npjOSLCR2hE

LIST FUNCTIONS

- Some list functions:
  - *append(x) · insert(i, x) · remove(x) · pop(i=-1) · index(x) · count(x) · sort() · reverse()*
- Lists as stacks:
  - Use *append(x)* to push items and *pop()* to pop them
- Lists as queues:
  - Use *append(x)* to enqueue items and *pop(0)* to dequeue them

MULTIPLE ASSIGNMENT

- Swap?
- Most languages:

  -
    ```
    temp = x
    x = y
    y = temp
    ```
- Python:

  -
    ```
    x, y = y, x
    ```

DICTIONARIES

- Also known as *associative arrays* or *maps*
- Creating: *d* = *{key1: value1, key2:value2, …}*
- Mutating: *d[key]* = *value*
- Accessing: *d[key]*
- Checking membership: *d.has_key(key)*

Q6–7

UNIT TESTING IN PYTHON

- Multiple approaches
- Easiest is probably the doctest module plus conditional execution

DOCTEST EXAMPLE

```python
import doctest

# The following function is from the Python Tutorial
def average(values):
    """Computes the arithmetic mean of a list of numbers.

    >>> print average([1])
    1.0
    >>> print average([1,2])
    1.5
    >>> print average([1,2,3])
    2.0
    >>> print average([1,-2,3])
    0.666666666667
    """
    return sum(values, 0.0) / len(values)

if __name__ == '__main__':
    doctest.testmod()
```

Test cases and expected results

Conditional Execution

MILESTONE 1

- Have you found three languages for your essay?
- Avoid “toy” languages:
  - Funny
  - Not fun to live with for a term