

## Objectives

- Dictionary wrap up
- Loose ends:
  - Using/improving documentation
  - Default parameter values
- Digging into Object Oriented Programming
- Broader Issue: Diversity in Computing

Oct 26, 2007

Sprenkle - CS111

1

## Lab 6 Solution

- My complete solution will be in today's examples directory

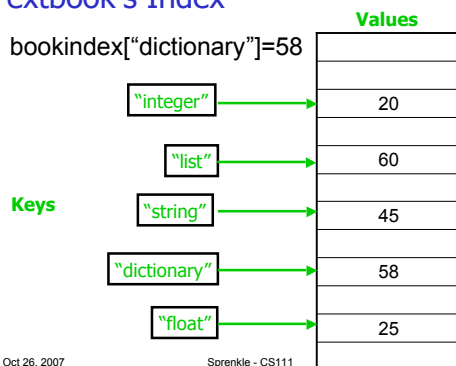
Oct 26, 2007

Sprenkle - CS111

2

## Textbook's Index

- `bookindex["dictionary"] = 58`



Oct 26, 2007

Sprenkle - CS111

3

## Lists vs. Dictionaries

Lists	Dictionaries
<i>integer positions</i> (0, ...) to any type of value	Map <i>immutable keys</i> (int, float, string) to any type of value
Ordered	Unordered
Slower to find a value ( <b>in</b> )	Fast to find a value (use key)
Fast to print in order	Slower to print in order (by key)
Only as big as you make it	Takes up a lot of space (so can add elements in the middle)

Oct 26, 2007

Sprenkle - CS111

4

## Getting Documentation

- **dir**: function that returns a list of methods and attributes in an object
  - `dir(<type>)`
- **help**: get documentation
  - Example Use in the Python shell
    - `help(<type>)`
    - `import <module name>`
    - `help(<module name>)`

Oct 26, 2007

Sprenkle - CS111

5

## Where is Documentation Coming From?

- Comes from the code itself in "**doc strings**"
  - i.e., "documentation strings"
- Doc strings are simply strings *after* the function header

➢ Typically use triple-quoted strings because documentation goes across several lines

```
def verse(animal, sound):  
    """ prints a verse of Old MacDonald, filling  
    in the strings for animal and sound """
```

Oct 26, 2007

Sprenkle - CS111

6

## Defaults for Parameters

- Saw with the **xrange** function
  - Didn't have to specify start or increment when calling the function
  - Default start to 0
  - Default increment to 1
- Can assign a **default value** to a parameter
  - In general, default parameter should come after all the parameters that need to be defined

Oct 26, 2007

Sprenkle - CS111

7

## Using Default Parameters

- By default the rollDie function could assume that a die has 6 sides

Assigns a value **ONLY IF** not passed a parameter

```
def rollDie(sides=6):  
    return random.randint(1,sides)
```

Examples of calling the function:

```
rollDie(6)  
rollDie()  
rollDie(12)
```

Oct 26, 2007

Sprenkle - CS111

game.py

8

## Problem: Student Majors

- We want to keep track of the number of majors of each type
  - Twist: Not every student has a major (don't declare until sophomore year)
  - Make a function to handle updating the dictionary
  - New data file: data/majors.all.dat

Oct 26, 2007

Sprenkle - CS111

9

## Problem: Student Majors, revised

- Students can have more than one major
  - Should count these separately
- How can we modify the previous program to do that?

Oct 26, 2007

Sprenkle - CS111

10

## Why Majors Problem Ad Nauseam?

- "Parsing" data files for different purposes is very common in science

### Simplified web application access log:

```
128.4.131.54 [09/Aug/2005:14:01:35] GET /dspace/simple-search  
128.4.133.79 [09/Aug/2005:14:13:13] GET /dspace/simple-search  
128.4.133.139 [09/Aug/2005:14:28:20] GET /dspace/simple-search  
128.4.133.139 [09/Aug/2005:14:32:45] GET /dspace/adv-search  
...
```

### I write scripts to

- create user sessions (use as test cases)
- analyze user sessions (avg. length, patterns)
- emulate user sessions

Oct 26, 2007

Sprenkle - CS111

11

## Programming Paradigm: Imperative

- Most modern programming languages are **imperative**
- Have **data** (numbers and strings in variables)
- Perform **operations** on data using operations, such as + (addition and concatenation)
- Data and operations are separate
- Add to imperative: object-oriented programming

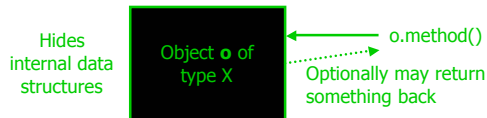
Oct 26, 2007

Sprenkle - CS111

12

## Object-Oriented Programming

- Program is a collection of **objects**
- Objects **combine** data and methods together
- Objects interact by invoking methods on other objects
  - Methods perform some operation on object



Oct 26, 2007

Sprenkle - CS111

13

## Using a Graphics Module/Library

- Allows us to handle graphical input and output
  - Example input: Pictures
  - Example output: Mouse clicks
- Not part of a standard Python distribution
- Made up of a collection of related **classes** of data and operations (**methods**) that manipulate the data
- ➔ Use the library to help us learn OO programming

Oct 26, 2007

Sprenkle - CS111

14

## Using a Graphics Module/Library

- Handout lists the various classes
  - **Constructor** is in bold
    - Review: creates an object of that type
  - Methods and their parameters are listed for each class
  - Drawn objects have some common methods
    - Listed at end of handout
- Known as an **API**
  - **Application Programming Interface**

Oct 26, 2007

Sprenkle - CS111

15

## Snippet of Code

- Using the handout, what does this code do?

```
from graphics import *

def main():
    win = GraphWin("My Circle", 100, 100)
    c = Circle(Point(50,50), 10)
    c.draw(win)
    win.getMouse()

main()
```

Oct 26, 2007

Sprenkle - CS111

16

## Snippet of Code

- Using the handout, what does this code do?

```
from graphics import *

def main():
    win = GraphWin("My Circle", 100, 100)
    c = Circle(Point(50,50), 10)
    c.draw(win)
    win.getMouse()

main()
```

GraphWin object → win = GraphWin("My Circle", 100, 100)  
Also known as an **instance of the GraphWin class**  
Constructor → win = GraphWin("My Circle", 100, 100)  
Method called on GraphWin object → win.getMouse()

Oct 26, 2007

Sprenkle - CS111

17

## Mid-Semester Evaluation: Agree on

- Course material - interesting
- Lectures - clear
- Examples - about right (1 dissent each direction)
- Textbook - N/A or useless (2 dissent, positive)
- Web Site - good
- Grading - fair
- Lab assignments: helpful in learning; just about right length/diff
  - 3 dissents - too long; 1 dissent - too hard
- Difficulty - same or increasing
- ➔ Readings - right length/difficulty; good discussion, understanding; OK to enjoyable

Oct 26, 2007

Sprenkle - CS111

18

## Disagree on

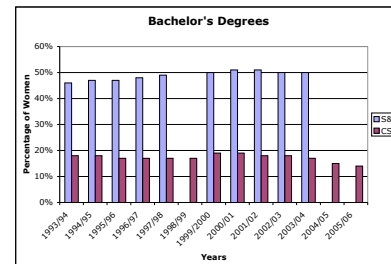
- Pace: 5 - right; 2 - fast; 3 slow
- Workload compared to others: bell curve

Oct 26, 2007

Sprenkle - CS111

19

## Diversity in Computer Science



- S&E around 50%, CS < 20%

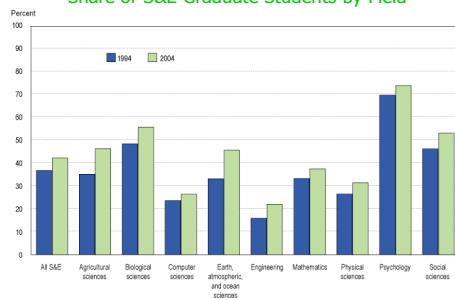
Oct 26, 2007

Sprenkle - CS111

20

## Graduate Enrollment

Share of S&E Graduate Students by Field



Oct 26, 2007

Sprenkle - CS111

21

## Attracting Computer Scientists

- Demand for software engineers is high
  - Create new software, applications --> improve productivity
- Computer science majors are decreasing
  - Dropped with dot-com bust
  - Losing "normal" people of all genders, ethnicities
- Need to attract and retain more majors
  - Maintain technical innovation, diversity of ideas
  - Computing's effect on other fields
- Does attracting new majors compromise/weaken the discipline?

Oct 26, 2007

Sprenkle - CS111

22

## Discussion

- Two groups. Any guesses how divided?

Stereotype	% Truth	Perception changed?	How to Address?

Some stereotypes have some truth to them

Since W&L or since CS111

In classroom?  
As a profession?  
Recruitment tools?

Oct 26, 2007

Sprenkle - CS111

23