

Objectives

- More on Dictionaries

Oct 24, 2007

Sprenkle - CS111

1

Dictionaries in Python

- Map **keys** to **values**
 - Keys are not necessarily alphabetized
 - Mappings are from *one* key to one *or more* values
 - Keys are **unique**, Values are not necessarily unique
 - Example: student ids --> last names
 - Keys must be **immutable** (numbers, strings)
- Similar to Hashtables/Hashmaps in other languages

How would we handle if there is more than one value for a key?

Oct 24, 2007

Sprenkle - CS111

2

Why Dictionaries?

- Useful for **fast** lookups
 - Lookup by keyword
 - Don't need to look through entire list or sort list first

Oct 24, 2007

Sprenkle - CS111

3

Examples of Dictionaries

- What are the keys and values for
 - Dictionary
 - Textbook's index
 - Cookbook
 - URL (Uniform Resource Locator)
- Examples from class
 - Function name --> function definition
 - Variable name --> value
 - ASCII value --> character

Oct 24, 2007

Sprenkle - CS111

4

Creating Dictionaries in Python

Syntax:

```
{<key>:<value>, ..., <key>:<value>}
```

```
empty = {}
```

```
ascii = { 'a':97, 'b':98, 'c':99, ..., 'z':122 }
```

Or, use the constructor:

```
obj = dict([( <key>, <value> ), ..., ( <key>, <value> )])
```

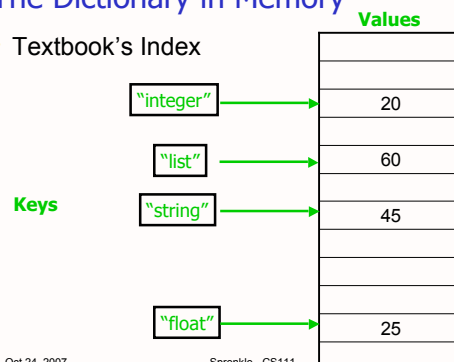
Oct 24, 2007

Sprenkle - CS111

5

The Dictionary in Memory

- Textbook's Index



Oct 24, 2007

Sprenkle - CS111

6

Accessing Values using Keys

- Syntax:
`<dictionary>[<key>]`
- Examples:
`ascii['z']`
`directory['registrar']`
- **KeyError** if key is not in dictionary

Oct 24, 2007

Sprenkle - CS111

7

Adding/Modifying Key-Value Pairs

- Syntax:
`<dictionary>[<key>] = <value>`
- `ascii['a'] = 97`

Oct 24, 2007

Sprenkle - CS111

`ascii_dictionary.py`

8

Textbook's Index

- `bookindex["dictionary"] = 58`

Values	
"integer"	20
"list"	60
"string"	45
"dictionary"	58
"float"	25

Keys

Oct 24, 2007

Sprenkle - CS111

9

Dictionary Operations

Indexing	<code><dict>[<key>]</code>	
Length	<code>len(<dict>)</code>	← # of keys
Iteration	<code>for <key> in <dict>:</code>	
Membership	<code><key> in <dict></code>	
Deletion	<code>del <dict>[<key>]</code>	

Unlike strings and lists, doesn't make sense to do slicing, concatenation, repetition for dictionaries

Oct 24, 2007

Sprenkle - CS111

10

Dictionary Methods

Method Name	Functionality
<code><dict>.clear()</code>	Remove all items from dictionary
<code><dict>.keys()</code>	Returns a copy of dictionary's <i>list</i> of keys
<code><dict>.values()</code>	Returns a copy of dictionary's <i>list</i> of values
<code><dict>.get(x, default)</code>	Returns <code><dict>[x]</code> if <code>x</code> is a key; Otherwise, returns <code>None</code> (or default value)

Oct 24, 2007

Sprenkle - CS111

11

Special Value: **None**

- Special value we can use
 - E.g., Return value from function when there is an error
- Similar to **null** in Java
- If you execute

```
list = list.sort()
print list
```

 - Prints `None` because `list.sort()` does **not** return anything

Oct 24, 2007

Sprenkle - CS111

`Example in shell`

12

Example Problem

- Given a file of the form
 - <lastname> <major>
- Create a mapping between the last names and majors

Oct 24, 2007

Sprenkle - CS111 majors_dictionary.py 13

Example Problem

- Modify the previous program to keep track of the number of majors of each type

➤ Could we solve this using a list?

Oct 24, 2007

Sprenkle - CS111 majors_dictionary2.py 14

Lists vs. Dictionaries

Oct 24, 2007

Sprenkle - CS111

15

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 (in)	Fast to find a value (use key)
Fast to print in order	Slower to print in order (by key)

Oct 24, 2007

Sprenkle - CS111

16

Mid-Semester Survey

Oct 24, 2007

Sprenkle - CS111

17