Announcements

Reminders:
- Program 8 – due Tues, Dec. 5th by 11:55pm

Lab from Last Time

Dictionaries

**Dictionary**: object that stores a collection of data
- Each element consists of a *key* and a *value*
  - Often referred to as *mapping* of key to value
  - Key must be an immutable object (cannot be changed!)
- To retrieve a specific value, use the key associated with it
- Format for creating a dictionary
  ```python
dictionary = {key1: val1, key2: val2}
```
Retrieving a Value from a Dictionary

- Elements in dictionary are unsorted
- General format for retrieving value from dictionary:
  \[ \text{dictionary}[\text{key}] \]
  - If \text{key} in the dictionary, associated value is returned, otherwise, \text{KeyError} exception is raised
- Test whether a key is in a dictionary using the \text{in} and \text{not in} operators
  - Helps prevent \text{KeyError} exceptions

Program Output
{ 'Chris': '555-1111', 'Katie': '555-3333', 'JoAnne': '555-2222'}

Katie's phone number is: 555-3333

Adding Elements to an Existing Dictionary

- Dictionaries are mutable objects
- To add a new key-value pair:
  \[ \text{dictionary}[\text{key}] = \text{value} \]
  - If key exists in the dictionary, the value associated with it will be changed

Program Output

Before Add: { 'JoAnne': '555-2222', 'Chris': '555-1111', 'Katie': '555-3333'}

After Add: { 'JoAnne': '555-2222', 'Chris': '555-1111', 'Katie': '555-3333', 'Andy': '555-0123'}
Deleting Elements From an Existing Dictionary

- To delete a key-value pair:
  ```python
def main():
    phonebook = {'Chris': '555-1111', 'Katie': '555-3333', 'JoAnne': '555-2222'}
    print("Before Delete:", phonebook)
    del phonebook[JoAnne]
    print("After Delete:", phonebook)
```

Program Output
Before Delete: {'Katie': '555-3333', 'JoAnne': '555-2222', 'Chris': '555-1111'}
After Delete: {'Katie': '555-3333', 'Chris': '555-1111'}

Using a Dictionary as a Color Map

Dictionaries map one value to another

We could map numbers to color names instead of names to phone #s, or we could map numbers to images.

Example:
```python
```

Getting the Number of Elements and Mixing Data Types

- \texttt{len} function: used to obtain number of elements in a dictionary
- Keys must be immutable objects, but associated values can be any type of object
  - One dictionary can include keys of several different immutable types
- Values stored in a single dictionary can be of different types
Creating an Empty Dictionary and Using for Loop to Iterate Over a Dictionary

• To create an empty dictionary:
  – Use {}
  – Use built-in function dict()
  – Elements can be added to the dictionary as program executes

• Use a for loop to iterate over a dictionary
  – General format: for key in dictionary:

Some Dictionary Methods

• get method: gets a value associated with specified key from the dictionary
  – Format: dictionary.get(key, default)
  • default is returned if key is not found
  – Alternative to [] operator
  • Cannot raise KeyError exception

• keys method: returns all the dictionaries keys as a sequence
  – Format: dictionary.keys()

Some Dictionary Methods (cont’d.)

• values method: returns all the dictionaries values as a sequence
  – Format: dictionary.values()
  – Use a for loop to iterate over the values

Some Dictionary Methods

```python
>>> wordDictionary = {'the':16, 'a':7, 'whoose':1, 'its': 3}
>>> wordDictionary.keys()
['the', 'a', 'whoose', 'its']
>>> wordDictionary.get('the', 0)
16
>>> wordDictionary.get('latter', 0)
0
>>> wordDictionary.values()
dict_values([('its', 3), ('whoose', 1), ('a', 7), ('the', 16)])
```
### Dictionary Methods

**Table 10-1** Some of the dictionary methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear</td>
<td>Clears the contents of a dictionary.</td>
</tr>
<tr>
<td>get</td>
<td>Gets the value associated with a specified key. If the key is not found, the method does not raise an exception. Instead, it returns a default value.</td>
</tr>
<tr>
<td>items</td>
<td>Returns all the keys in a dictionary and their associated values as a sequence of tuples.</td>
</tr>
<tr>
<td>keys</td>
<td>Returns the values associated with a specified key and removes that key-value pair from the dictionary. If the key is not found, the method returns a default value.</td>
</tr>
<tr>
<td>pop</td>
<td>Returns a randomly selected key-value pair as a tuple from the dictionary and removes that key-value pair from the dictionary.</td>
</tr>
<tr>
<td>values</td>
<td>Returns all the values in the dictionary as a sequence of tuples.</td>
</tr>
</tbody>
</table>

---

### Practice

- Write a program that opens the file Lincoln.txt (in Box.com folder) and counts the number of occurrences of each word.
- Print out each word and the number of times it occurs on a separate line.
- Loop through the dictionary to find the word that occurred the most – print that out.
- Hints: You will need to go through the file line by line and split the line – the split function returns a list – loop through that list of words and put them into your dictionary and keep track of how many times you saw each word.