Last Class We Covered

• Python’s tuple data structure
• Tuples in functions (and as return values)
• Basic tuples operations, including...
  – Creation
  – Conversion
  – Repetition
  – Slicing
  – Traversing
Any Questions from Last Time?
Tuple Practice

def min_max(t):
    """Returns the smallest and largest elements of a sequence as a tuple""
    return (min(t), max(t))

seq = [64, 71, 42, 73, 85, 33]
minOutput, maxOutput = min_max(seq)
print(minOutput, maxOutput)

string = 'We are the Knights who say... NI.'
print (min_max(string))  
(33, 85)
(' ', 'y')
def printall(______):
    print (args)

printall(1, 2.0, 'three')
def printall(*args):
    print (args)

printall(1, 2.0, 'three')

What does this do?
Any Questions from Last Time?
Lesson objectives

• Construct dictionaries and access entries in those dictionaries
• Use methods to manipulate dictionaries
• Decide whether a list or a dictionary is an appropriate data structure for a given application
Dictionaries

• A dictionary organizes information by association, not position
  – Example: When you use a dictionary to look up the definition of “mammal,” you don’t start at page 1; instead, you turn directly to the words beginning with “M”

• Data structures organized by association are also called tables or association lists

• In Python, a dictionary associates a set of keys with data values
Dictionary Keys

• In Python, a **dictionary** is a set of 'keys' (words) all pointing to their own 'values' (meanings).

```python
dict1 = {"first_name" : "John", "last_name" : "Cleese"}
```
Dictionaries

- Keys can be data of any immutable types, including other data structures.
- It is best to think of a dictionary as an unordered set of *key: value* pairs, with the requirement that the keys are unique (within one dictionary).

```
dict1['John'] = 'Leo'
```

<table>
<thead>
<tr>
<th>Dictionary name</th>
<th>Key 1</th>
<th>Value 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>
Creating Dictionaries
Creating Dictionaries

• There are three main ways to create a dictionary in Python:
  1. Construct a python dictionary (with curly braces syntax)
  2. You can also construct a dictionary from a list (or any iterable data structure) of key, value pairs
  3. Construct a dictionary from parallel lists
Creating Dictionaries (Curly Braces)

• The empty dictionary is written as two curly braces containing nothing

```python
dict1 = {}
```

• To cast a list as a dictionary, you use `dict()`

```python
dict1 = {"fname" : "John", "lname" : "Cleese"}
print (dict1)
```

```python
{ 'lname': 'Cleese', 'fname': 'John'}
```
Creating Dictionaries

dict1 = [('a', 'apple')]
print (dict1, type(dict1))

Is this a dictionary?

[('a', 'apple')] <class 'list'>

Must use curly braces {} to define a dictionary
Creating Dictionaries

dict2 = {'a', 'apple'}
print (dict2, type(dict2))

Must use a colon (:) between items, not a comma
Creating Dictionaries

dict3 = {'a':'apple'}
print (dict3, type(dict3))

Is this a dictionary?

{'a': 'apple'} <class 'dict'>

Hooray!
Creating a Dictionary

```python
eng2sp = dict()
print(eng2sp)
   {} <class 'dict'>
eng2sp['one'] = 'uno'
print(eng2sp)
   {'one': 'uno'} <class 'dict'>
eng2sp['two'] = 'dos'
print(eng2sp)
   {'two': 'dos', 'one': 'uno'} <class 'dict'>
```

What does this output?

What does this output?

What does this output?
Creating Dictionaries (From List)

- To cast a list as a dictionary, you use `dict()`

```python
myList = [(5, 'candy'), (15, 'cookies'), (23, 'ice cream')]
myDict = dict(myList)
print(type(myDict))
```

Must be key pairs

```
<class 'dict'>
```
Creating Dictionaries (From Parallel Lists)

Here we have two parallel lists that we are putting together into a dictionary.

```python
names = ['Tina', 'Pratik', 'Amber']
major = ['Social Work', 'Pre-Med', 'Art']

major_dict = {}
for i in range(len(names)):
    major_dict[names[i]] = major[i]

print(major_dict)

{'Pratik': 'Pre-Med', 'Tina': 'Social Work', 'Amber': 'Art'}
```

From: https://docs.python.org/3.3/tutorial/datastructures.html
Creating Dictionaries (From Parallel Lists)

• Rather than using a for loop, there is a built-in function that can put parallel lists together (either into a tuple or dictionary)

• *Zip* is a built-in function that takes two or more sequences and “zips” them into a list of tuples, where each tuple contains one element from each sequence
Creating Dictionaries (From Parallel Lists)

names = ["Tina", "Pratik", "Amber"]
major = ["Social Work", "Pre-Med", "Art"]
majors_dict = dict(zip(names, major))
print(majors_dict)
print(type(majors_dict))

{'Amber': 'Art', 'Tina': 'Social Work', 'Pratik': 'Pre-Med'}
<class 'dict'>
Creating Dictionaries

• One other way to create a dictionary is by using **dictionary comprehension**

```python
dict1 = {x: x**2 for x in (2, 4, 6)}
print(dict1)
```

```python
{2: 4, 4: 16, 6: 36}
```

What does this output?
Dictionary Operations
Dictionary Operations

1. Accessing Values in Dictionary
2. Updating Dictionaries
3. Delete Dictionary Elements
Accessing Values in Dictionary

• To access dictionary elements, you can use the square brackets along with the key to obtain its value

```python
dict1 = {'FName': 'Mike', 'LName': 'Jones', 'Age': 18};

print ("dict1['FName']": ", dict1['FName'])
print ("dict1['Age']": ", dict1['Age'])
```

dict1['FName']': Mike
dict1['Age']': 18
Updating Dictionaries

dict1 = {'FName': 'Mike', 'LName': 'Jones', 'Age': 18};

print("Before Update")
print("dict1['FName']: ", dict1['FName'])
print("dict1['Age']: ", dict1['Age'])

dict1['School'] = "UMBC"
dict1['Age'] = 19

print("After Update")
print("dict1['School']: ", dict1['School'])
print("dict1['Age']: ", dict1['Age'])
Updating Dictionaries

Before Update

dict1['FName']: Mike

dict1['Age']: 18

After Update

dict1['School']: UMBC

dict1['Age']: 19
Delete Dictionary Elements

• You can either remove individual dictionary elements or clear the entire contents of a dictionary.

• You can also delete an entire dictionary in a single operation.
Delete Dictionary Elements

dict1 = {'FName': 'Mike', 'LName': 'Jones', 'Age': 18};

print("Before Update")
print("dict1['FName']": ", dict1['FName'])
print("dict1['LName']": ", dict1['LName'])
print("dict1['Age']": ", dict1['Age'])

del dict1['FName'] # remove entry with key 'Name'
#dict1.clear()       # remove all entries in dict
#del dict1           # delete entire dictionary

print("After Update")
print("dict1['LName']": ", dict1['LName'])
print("dict1['Age']": ", dict1['Age'])

If we remove, the dictionary, it will cause an error.
Dictionary Functions and Methods
Functions and Methods

- `len(dict)`
- `str(dict)`
- `type(variable)`
- `dict.clear()`
- `dict.copy()`
- `dict.fromkeys()`
- `dict.get(key, default=None)`
- `dict.items()`
- `dict.values()`
- `dict.keys()`
- `dict.setdefault(key, default=None)`
- `dict.update(dict2)`

From: http://www.tutorialspoint.com/python/python_dictionary.htm
Functions

• \textit{len} (dict)
  - Gives the total length of the dictionary. This would be equal to the number of items in the dictionary.

• \textit{str} (dict)
  - Produces a printable string representation of a dictionary

• \textit{type} (variable)
  - Returns the type of the passed variable. If passed variable is dictionary, then it would return a dictionary type.

From: http://www.tutorialspoint.com/python/python_dictionary.htm
Methods

- `dict.clear()`
  - Removes all elements of dictionary `dict`

- `dict.copy()`
  - Returns a shallow copy of dictionary `dict`

- `dict.fromkeys(seq, value=None)`
  - Create a new dictionary with keys from seq and values `set` to `value`.

- `dict.get(key, default=None)`
  - For `key` key, returns value or default if key not in dictionary

From: http://www.tutorialspoint.com/python/python_dictionary.htm
Methods

• `dict.items()`
  – Returns a list of `dict`'s (key, value) tuple pairs

• `dict.values()`
  – Returns list of dictionary `dict`'s values

• `dict.keys()`
  – Returns list of dictionary `dict`'s keys

From: http://www.tutorialspoint.com/python/python_dictionary.htm
Methods

- **dict.setdefault(key, default=None)**
  - Similar to get(), but will set dict[key]=default if key is not already in dict

- **dict.update(dict2)**
  - Adds dictionary dict2's key-values pairs to dict
When to Use a Dictionary?

• You have to retrieve things based on some identifier, like names, addresses, or anything that can be a key.

• You don't need things to be in order. Dictionaries do not normally have any notion of order, so you have to use a list for that.

• You are going to be adding and removing elements and their keys.
Dictionary Examples
Example: The Hexadecimal System

- You can keep a hex-to-binary lookup table to aid in the conversion process

```python
hexToBinaryTable = {'0': '0000', '1': '0001', '2': '0010', '3': '0011', '4': '0100', '5': '0101', '6': '0110', '7': '0111', '8': '1000', '9': '1001', 'A': '1010', 'B': '1011', 'C': '1100', 'D': '1101', 'E': '1110', 'F': '1111'}
```
Example: The Hexadecimal System

• You can keep a hex-to-binary **lookup table** to aid in the conversion process

```python
def convert(number, table):
    binary = ''
    for digit in number:
        binary = binary + table[digit]
    return binary

def main():
    print(convert("34A", hexToBinaryTable))
    print(convert("11C", hexToBinaryTable))
main()
```

001101001010
000100011100
Dictionary Example (Psychotherapist)

• Doctor in this kind of therapy responds to patient’s statements by rephrasing them or indirectly asking for more information

• For example:
  – Writing a program that emulates a nondirective psychotherapist
Dictionary Example (Psychotherapist)

-bash-4.1$ python psych.py
Good morning, I hope you are well today.
What can I do for you?

>> my dad and I don't like each other
You seem to think that your dad and you don't like each other

>> my mother and father are mean to each other
Why do you say that your mother and father are mean to each other

>> I like to eat candy
Many of my patients tell me the same thing.
Dictionary Example (Psychotherapist)

• When user enters a statement, program responds in one of two ways:
  – With a randomly chosen hedge, such as “Please tell me more”
  – By changing some key words in user’s input string and appending the string to a randomly chosen qualifier

• Thus, to “My teacher always plays favorites,” the program might reply, “Why do you say that your teacher always plays favorites?”

From: Fundamentals of Python: From First Programs through Data Structures
Dictionary Example (Psychotherapist)

• Program consists of a set of collaborating functions that share a common data pool

• Pseudocode:
  output a greeting to the patient
  while True
    prompt for and input a string from the patient
    if the string equals “Quit”
      output a sign-off message to the patient
    break
  call another function to obtain a reply to this string
  output the reply to the patient
import random

hedges = ("Please tell me more.",
          "Many of my patients tell me the same thing.",
          "Please continue.")

qualifiers = ("Why do you say that ",
              "You seem to think that ",
              "Can you explain why ")

replacements = {"I":"you", "me":"you", "my":"your",
                "we":"you", "us":"you", "mine":"yours"}
Dictionary Example (Psychotherapist)

def reply(sentence):
    probability = random.randint(1,4)
    if probability == 1:
        return random.choice(hedges)
    else:
        return random.choice(qualifiers) + changePerson(sentence)

def changePerson(sentence):
    words = sentence.split()
    replyWords = []
    for word in words:
        replyWords.append(replacements.get(word, word))
    return " ".join(replyWords)
Dictionary Example (Psychotherapist)

```python
def main():
    print("Good morning, I hope you are well today.")
    print("What can I do for you?")
    while True:
        sentence = input("\n>> ")
        if sentence.upper() == "QUIT":
            print ("Have a nice day!")
            break
        print(reply(sentence))
    main()
```

From: Fundamentals of Python: From First Programs through Data Structures
Dictionary Example (Psychotherapist)

• Functions in this program can be tested in a bottom-up or a top-down manner

• Program’s replies break down when:
  – User addresses the therapist in the second person
  – User uses contractions (for example, I’m and I’ll)

• With a little work, you can make the replies more realistic
Any Other Questions?
Announcements

• No Lab this week (November 23rd to 26th)
  – No office hours after Wednesday at 2:30pm

• Homework 8 has been posted
  – Due on Tuesday, November 24th at 8:59pm

• Project 2
  – Will be posted on Tuesday, November 24th
  – Due on Tuesday, December 8th

• Next Class: Algorithms and Analysis
Have a Happy Thanksgiving!