# W18 Reading Unit 6

June 26, 2016

## 1 Modules and Packages

Programs: you have been weirring a series of "mini programs"

```
you can save the Python snippets into a regular text file, add the ".py" extension, and run it on the co

    e.g. python test.py

you can get command line arguments by importing sys and using the argv list.
    e.g. import sys
          print ('Program Arguments: ', sys.argv)
```

### 1.1 Modules

Modules are contained in a single Pyhon file and contains all of the variables. functions, and classes that you can use in an external program

you will be able to interact with the module as if the code was pasted right on top of your code, with a caveat

e.g.

```
import sys
print('Program Arguments: ', sys.argv)

we have imported a module called sys that we wanted to add to our code. we import the sys module to use

The caveat is that in order to prevent naming conflicts, in order to access anything in the sys module,
```

### 1.2 Packages

- A directory that contains a collection of modules.
- Packages are a great way to organize modules that are related to each other.
- Packages are essentially directories that have a special empty fiile named "_init_.py".

e.g. import numpy.matlib - in this example "numpy" is the package and matlib is the module within the package numpy. - the package "numpy" has an empty file named "_init_.py" that tells Python that numpy is a package.

## 2 Modules and the Import Statement

### 2.1 Standalone Programs

```
In [9]: !python helloworld.py

Hello World, revisited
```

```
In [7]: !cat helloworld.py
```

```
# helloworld.py
#
# This is our first python script rum from the command line as a stand alone
# application

print("Hello World, revisited")
```

```
In [15]: !cat showarguments.py
```

```
# showarguments.py
#
# Prints out the program arguments that have been passed into this standalone
# application

import sys # We need to import the sys module in order to gain access to the arguments

print("Program argument:" , sys.argv)
```

```
In [16]: !python showarguments.py
```

```
Program argument: ['showarguments.py']
```

```
In [18]: !python showarguments.py firstArg secondArg thirdArg
```

```
Program argument: ['showarguments.py', 'firstArg', 'secondArg', 'thirdArg']
```

## 2.2 Modules and the import stetement

```
In [25]: !cat weatherman.py
```

```
# weatherman.py
#
#Provides the current forecasr for the weather in Berkeley, California

from urllib.request import urlopen
import json

def get_report():
    """
    Returns the current forecast of Berkeley right now
    """
    response = urlopen(
        'http://api.openweathermap.org/data/2.5/weather?q=Berkeley,ca&appid=7dc34849d7e8b6fbdcb3f12454c
        rawWeatherData = response.read().decode("utf-8")
        weatherData = json.loads(rawWeatherData)

        forecast = "Berkeley, CA Forecast: " + weatherData["weather"][0]["main"]
        return forecast
```

```
In [27]: import weatherman

         print(weatherman.get_report())
```

```
Berkeley, CA Forecast: Drizzle
```

### 2.3 1. Import a specific function directly from he external module

```
In [29]: # we import this specific function that we need directly from the weatherman
         # module, notice we do not need to write weatherman in front of it when we call
         # it, but you should be careful if two different modules have the same name
         from weatherman import get_report
         print(get_report())
```

```
Berkeley, CA Forecast: Mist
```

### 2.4 2. Import the module name to avoid naming conflicts

```
In [32]: import weatherman
         print(weatherman.get_report())
```

```
Berkeley, CA Forecast: Mist
```

### 2.5 3. import only what you want from a module by renaming the function

```
In [34]: from weatherman import get_report as do_it
         print(do_it())
```

```
Berkeley, CA Forecast: Mist
```

### 2.6 Note on Module Search Path

```
In [36]: import sys
         for place in sys.path:
             print(place)
```

```
C:\Anaconda3\python35.zip
C:\Anaconda3\DLLs
C:\Anaconda3\lib
C:\Anaconda3
c:\anaconda3\lib\site-packages\setuptools-20.3-py3.5.egg
C:\Anaconda3\lib\site-packages
C:\Anaconda3\lib\site-packages\Sphinx-1.3.5-py3.5.egg
C:\Anaconda3\lib\site-packages\win32
C:\Anaconda3\lib\site-packages\win32\lib
C:\Anaconda3\lib\site-packages\Pythonwin
C:\Anaconda3\lib\site-packages\IPython\extensions
C:\Users\hrli1\.ipython
```

## 3 Packages

```
In [40]: !cat weather/daily.py
```

```
# daily.py
#
# Provides the weather for Berkeley, CA for today

from urllib.request import urlopen
import json

def forecast():
    """
    Returns the daily weather for Berkeley, CA
    """
    response = urlopen('http://api.openweathermap.org/data/2.5/forecast/daily?q=Berkeley&mode=json&unit
    rawWeatherData = response.read().decode("utf-8")
    weatherData = json.loads(rawWeatherData)

    forecastStr = "Forecast for Berkeley, CA: " + weatherData["list"][0]["weather"][0]["main"] + "\n" \
        "Current Temp: " + str(weatherData["list"][0]["temp"]["day"]) + " degrees \n"  \
        "High Temp: " + str(weatherData["list"][0]["temp"]["max"]) + " degrees \n" \
        "Low Temp: " + str(weatherData["list"][0]["temp"]["min"]) + " degrees"

    return forecastStr
```

```
In [41]: !cat weather/weekly.py
```

```
# weekly.py
#
# Provides the weather for Berkeley, CA for the week

from urllib.request import urlopen
import json
import datetime

def forecast():
    """
    Returns the daily weather for Berkeley, CA
    """
    response = urlopen('http://api.openweathermap.org/data/2.5/forecast/daily?q=Berkeley&mode=json&unit
    rawWeatherDataList = response.read().decode("utf-8")
    weatherDataList = json.loads(rawWeatherDataList)

    forecastStr = ""
    for i in range(7):
        forecastStr += _daily_forecast(weatherDataList["list"][i]) + "\n\n"

    forecastStr = forecastStr[:-2] # Remove the two newlines at the end
    return forecastStr


def _daily_forecast(weatherData):
    """
    Helper function that prints a single day's forecast
    """

    # Using python datetime support to convert a timestamp into a full date

    # First need to define the UTC offset for Berkeley, CA (UTC - 8:00) (not daylight savings time)
    current_utc_offset = -datetime.timedelta(hours=8)

    # Next we create a timezone based on the utc offset for Pacific Standard Time
    current_timezone = datetime.timezone(current_utc_offset)

    # Last we create a datetime object based on the timestamp provided by the response, and
    # we localize the timezone to represent Pacific Standard Time
    current_datetime = datetime.datetime.fromtimestamp(weatherData["dt"], current_timezone)

    # Printing of the forecast

    # Note we use strftime to format how we would like to print out the datetime
    forecastStr = "Forecast for Berkeley, CA on " + current_datetime.strftime("%A, %B %d, %Y %H:%M %p")
    "Weather: "  + weatherData["weather"][0]["main"] + "\n" \
    "Current Temp: " + str(weatherData["temp"]["day"]) + " degrees \n"  \
    "High Temp: " + str(weatherData["temp"]["max"]) + " degrees \n" \
    "Low Temp: " + str(weatherData["temp"]["min"]) + " degrees"

    return forecastStr
```

```
In [42]: !touch weather/__init__.py
```

```
In [43]: from weather import daily,weekly
         print("Daily")
         print(daily.forecast())
         print()
         print("Weekly")
         print(weekly.forecast())
```

```
Daily
Forecast for Berkeley, CA: Clear
Current Temp: 64.24 degrees
High Temp: 64.24 degrees
Low Temp: 55.45 degrees

Weekly
Forecast for Berkeley, CA on Saturday, June 25, 2016 12:00 PM local time
Weather: Clear
Current Temp: 64.24 degrees
High Temp: 64.24 degrees
Low Temp: 55.45 degrees

Forecast for Berkeley, CA on Sunday, June 26, 2016 12:00 PM local time
Weather: Clear
Current Temp: 89.04 degrees
High Temp: 89.28 degrees
Low Temp: 54.09 degrees

Forecast for Berkeley, CA on Monday, June 27, 2016 12:00 PM local time
Weather: Clear
Current Temp: 90.14 degrees
High Temp: 91.15 degrees
Low Temp: 55.4 degrees

Forecast for Berkeley, CA on Tuesday, June 28, 2016 12:00 PM local time
Weather: Clouds
Current Temp: 86.13 degrees
High Temp: 93.69 degrees
Low Temp: 61.38 degrees

Forecast for Berkeley, CA on Wednesday, June 29, 2016 12:00 PM local time
Weather: Clear
```

```
Current Temp: 83.1 degrees
High Temp: 91.89 degrees
Low Temp: 61.61 degrees

Forecast for Berkeley, CA on Thursday, June 30, 2016 12:00 PM local time
Weather: Clear
Current Temp: 86.77 degrees
High Temp: 94.44 degrees
Low Temp: 59.2 degrees

Forecast for Berkeley, CA on Friday, July 01, 2016 12:00 PM local time
Weather: Clear
Current Temp: 83.95 degrees
High Temp: 90.81 degrees
Low Temp: 58.5 degrees
```

```
In [45]: from weather.daily import forecast as daily_forecast
         from weather.weekly import forecast as weekly_forecast
         print("Daily")
         print(daily_forecast())
         print()
         print("Weekly")
         print(weekly_forecast())
```

```
Daily
Forecast for Berkeley, CA: Clear
Current Temp: 64.24 degrees
High Temp: 64.24 degrees
Low Temp: 55.45 degrees

Weekly
Forecast for Berkeley, CA on Saturday, June 25, 2016 12:00 PM local time
Weather: Clear
Current Temp: 64.24 degrees
High Temp: 64.24 degrees
Low Temp: 55.45 degrees

Forecast for Berkeley, CA on Sunday, June 26, 2016 12:00 PM local time
Weather: Clear
Current Temp: 89.04 degrees
High Temp: 89.28 degrees
Low Temp: 54.09 degrees

Forecast for Berkeley, CA on Monday, June 27, 2016 12:00 PM local time
Weather: Clear
Current Temp: 90.14 degrees
High Temp: 91.15 degrees
Low Temp: 55.4 degrees

Forecast for Berkeley, CA on Tuesday, June 28, 2016 12:00 PM local time
Weather: Clouds
Current Temp: 86.13 degrees
High Temp: 93.69 degrees
Low Temp: 61.38 degrees
```

```
Forecast for Berkeley, CA on Wednesday, June 29, 2016 12:00 PM local time
Weather: Clear
Current Temp: 83.1 degrees
High Temp: 91.89 degrees
Low Temp: 61.61 degrees

Forecast for Berkeley, CA on Thursday, June 30, 2016 12:00 PM local time
Weather: Clear
Current Temp: 86.77 degrees
High Temp: 94.44 degrees
Low Temp: 59.2 degrees

Forecast for Berkeley, CA on Friday, July 01, 2016 12:00 PM local time
Weather: Clear
Current Temp: 83.95 degrees
High Temp: 90.81 degrees
Low Temp: 58.5 degrees
```

### 3.1 Note on datetimes in Python 3

```
In [57]: import datetime

         weatherData = {}
         weatherData["dt"] = 1440270976

         current_utc_offset = -datetime.timedelta(hours=8)

         current_timezone = datetime.timezone(current_utc_offset)

         current_datetime = datetime.datetime.fromtimestamp(weatherData["dt"],current_timezone)

         forecastStr = "Forecast for Berkeley, CA on "+current_datetime.strftime("%A, %B %d, %Y, %H:%M

         print(forecastStr)
```

```
Forecast for Berkeley, CA on Saturday, August 22, 2015, 11:16 AM local time
```

## 4 Python Standard Library

### 4.1 Handling KeyErrors automatically using setdefault

```
In [59]: periodic_table = {'Hydrogen':1,'Helium':2}
         periodic_table['Carbon']
```

```
        ---------------------------------------------------------------------------

        KeyError                                  Traceback (most recent call last)

        <ipython-input-59-fe3a730bccc8> in <module>()
          1 periodic_table = {'Hydrogen':1,'Helium':2}
    ----> 2 periodic_table['Carbon']
```

```
        KeyError: 'Carbon'
```

```
In [63]: print(periodic_table.setdefault('Hydrogen',12))
         print(periodic_table.setdefault('Helium',12))
         periodic_table.setdefault('Carbon',12)
         print(periodic_table)
```

```
1
2
{'Carbon': 12, 'Helium': 2, 'Hydrogen': 1}
```

## 4.2 Create a dictionary with a default value using defaultdic()

```
In [65]: from collections import defaultdict

         def not_an_element():
             return int()

         no_error_periodic_table = defaultdict(not_an_element)

         no_error_periodic_table['Hydrogen'] = 1
         no_error_periodic_table['Helium'] = 2
```

```
In [68]: print(no_error_periodic_table['Hydrogen'])
         print(no_error_periodic_table['Helium'])
         print(no_error_periodic_table['Blastium'])
```

```
1
2
0
```

```
In [78]: from collections import defaultdict
         from urllib.request import urlopen
         import json

         response = urlopen('https://www.googleapis.com/books/v1/volumes?q=berkeley&maxResults=40')
         rawData = response.read().decode("utf-8")
         book_data = json.loads(rawData)

         publisher_counter = defaultdict(int)

         for item in book_data['items']:
             publisher = item["volumeInfo"].setdefault("publisher", "None")
             publisher_counter[publisher] += 1

         for publisher, count in publisher_counter.items():
             print(publisher,count)
```

```
Cornell University Press 2
Frog Books 1
Husband Press 1
Cambridge University Press 1
Indiana University Press 1
```

```
Psychology Press 2
Genealogical Publishing Com 1
Springer Science & Business Media 1
ProQuest 1
Jones & Bartlett Learning 1
Courier Corporation 1
Lexington Books 1
University of Chicago Press 1
Arcadia Publishing 1
McFarland 1
Transaction Publishers 2
Oxford University Press on Demand 1
Penn State Press 1
Univ of California Press 3
North Atlantic Books 1
Basic Books 1
Princeton Architectural Press 1
Johnston Press 1
None 11
Oxford University Press, USA 1
```

```
In [81]: no_error_periodic_table = defaultdict(lambda: 99999)
         no_error_periodic_table['Hydrogen'] = 1
         no_error_periodic_table['Helium'] = 2
         no_error_periodic_table['Blastium']
```

```
Out[81]: 99999
```

## 4.3 Count items with Counter()

```
In [86]: from collections import Counter
         from urllib.request import urlopen
         import json

         response = urlopen('https://www.googleapis.com/books/v1/volumes?q=berkeley&maxResults=40')
         rawData = response.read().decode("utf-8")
         book_data = json.loads(rawData)

         berkeley_list = list()
         for item in book_data['items']:
             berkeley_list.append(item["volumeInfo"].setdefault("publisher", "None"))

         berkeley_counter = Counter(berkeley_list)
         print(berkeley_counter)
```

```
Counter({'None': 11, 'Univ of California Press': 3, 'Cornell University Press': 2, 'Psychology Press':
```

```
In [88]: berkeley_counter.most_common(3)
```

```
Out[88]: [('None', 11),
          ('Univ of California Press', 3),
          ('Cornell University Press', 2)]
```

```
In [89]: from collections import Counter
         from urllib.request import urlopen
         import json
```

```
        response = urlopen('https://www.googleapis.com/books/v1/volumes?q=stanford&maxResults=40')
        rawData = response.read().decode("utf-8")
        book_data = json.loads(rawData)

        stanford_list = list()
        for item in book_data['items']:
            stanford_list.append(item["volumeInfo"].setdefault("publisher", "None"))

        stanford_counter = Counter(stanford_list)
        print(stanford_counter)

Counter({'None': 16, 'Stanford University Press': 6, 'Cambridge University Press': 3, 'Oxford Universit

In [91]: berkeley_counter + stanford_counter

Out[91]: Counter({'Arcadia Publishing': 1,
                  'Basic Books': 1,
                  'California State Library': 1,
                  'Cambridge University Press': 4,
                  'Columbia University Press': 1,
                  'Cornell University Press': 2,
                  'Courier Corporation': 2,
                  'Frog Books': 1,
                  'Genealogical Publishing Com': 1,
                  'Hogarth': 1,
                  'Husband Press': 1,
                  'Indiana University Press': 1,
                  'IndyPublish.com': 1,
                  'Jessica Kingsley Publishers': 1,
                  'John Wiley & Sons': 2,
                  'Johnston Press': 1,
                  'Jones & Bartlett Learning': 1,
                  'Lexington Books': 1,
                  'MIT Press': 1,
                  'McFarland': 1,
                  "McGill-Queen's Press - MQUP": 1,
                  'New York Review of Books': 1,
                  'None': 27,
                  'North Atlantic Books': 1,
                  'Oxford University Press on Demand': 3,
                  'Oxford University Press, USA': 1,
                  'Penn State Press': 1,
                  'Princeton Architectural Press': 2,
                  'ProQuest': 1,
                  'Psychology Press': 2,
                  'Sports Publishing LLC': 1,
                  'Springer Science & Business Media': 1,
                  'Stanford University Press': 6,
                  'Transaction Publishers': 2,
                  'Univ of California Press': 3,
                  'University of Chicago Press': 1})

In [94]: berkeley_counter & stanford_counter
```

```
Out[94]: Counter({'Cambridge University Press': 1,
                  'Courier Corporation': 1,
                  'None': 11,
                  'Oxford University Press on Demand': 1,
                  'Princeton Architectural Press': 1})
```

```
In [96]: berkeley_counter | stanford_counter
         # | is or
```

```
Out[96]: Counter({'Arcadia Publishing': 1,
                  'Basic Books': 1,
                  'California State Library': 1,
                  'Cambridge University Press': 3,
                  'Columbia University Press': 1,
                  'Cornell University Press': 2,
                  'Courier Corporation': 1,
                  'Frog Books': 1,
                  'Genealogical Publishing Com': 1,
                  'Hogarth': 1,
                  'Husband Press': 1,
                  'Indiana University Press': 1,
                  'IndyPublish.com': 1,
                  'Jessica Kingsley Publishers': 1,
                  'John Wiley & Sons': 2,
                  'Johnston Press': 1,
                  'Jones & Bartlett Learning': 1,
                  'Lexington Books': 1,
                  'MIT Press': 1,
                  'McFarland': 1,
                  "McGill-Queen's Press - MQUP": 1,
                  'New York Review of Books': 1,
                  'None': 16,
                  'North Atlantic Books': 1,
                  'Oxford University Press on Demand': 2,
                  'Oxford University Press, USA': 1,
                  'Penn State Press': 1,
                  'Princeton Architectural Press': 1,
                  'ProQuest': 1,
                  'Psychology Press': 2,
                  'Sports Publishing LLC': 1,
                  'Springer Science & Business Media': 1,
                  'Stanford University Press': 6,
                  'Transaction Publishers': 2,
                  'Univ of California Press': 3,
                  'University of Chicago Press': 1})
```

## 4.4 Ordering dictionaries with OrderedDict()

```
In [97]: from collections import OrderedDict
         berkeley_publishers = OrderedDict(berkeley_counter)
         for publisher in berkeley_publishers:
             print(publisher)
```

```
Cornell University Press
Frog Books
```

```
Husband Press
Cambridge University Press
Indiana University Press
Psychology Press
Genealogical Publishing Com
Springer Science & Business Media
ProQuest
Jones & Bartlett Learning
Courier Corporation
Lexington Books
University of Chicago Press
Arcadia Publishing
McFarland
Transaction Publishers
Oxford University Press on Demand
Penn State Press
Univ of California Press
North Atlantic Books
Basic Books
Princeton Architectural Press
Johnston Press
None
Oxford University Press, USA
```

## 4.5 Using deques

```
In [101]: berkeley_publisher_list_ordered = list()
          for key, value in berkeley_counter.most_common():
              berkeley_publisher_list_ordered.append(key)
          print(berkeley_publisher_list_ordered)
```

```
['None', 'Univ of California Press', 'Cornell University Press', 'Psychology Press', 'Transaction Publi
```

```
In [104]: from collections import deque
          berkeley_publisher_list_ordered_deque = deque(berkeley_publisher_list_ordered)
          berkeley_publisher_list_ordered_deque.pop()
          berkeley_publisher_list_ordered_deque.popleft()
          print(berkeley_publisher_list_ordered_deque)
```

```
deque(['Univ of California Press', 'Cornell University Press', 'Psychology Press', 'Transaction Publishe
```

## 4.6 Pretty print with pprint

```
In [107]: from pprint import pprint
          pprint(berkeley_publisher_list_ordered)
```

```
['None',
 'Univ of California Press',
 'Cornell University Press',
 'Psychology Press',
 'Transaction Publishers',
 'Frog Books',
 'Husband Press',
 'Cambridge University Press',
 'Indiana University Press',
 'Genealogical Publishing Com',
 'Springer Science & Business Media',
 'ProQuest',
 'Jones & Bartlett Learning',
 'Courier Corporation',
 'Lexington Books',
 'University of Chicago Press',
 'Arcadia Publishing',
 'McFarland',
 'Oxford University Press on Demand',
 'Penn State Press',
 'North Atlantic Books',
 'Basic Books',
 'Princeton Architectural Press',
 'Johnston Press']
```

## 4.7 Note on installing third party packages

```
In [109]: pip install [package_name]
```

```
      File "<ipython-input-109-d8d590b65ae3>", line 1
    pip install [package_name]
              ^
SyntaxError: invalid syntax
```