Computing for Medicine: Phase 3, Seminar 2 Project

Jennifer Campbell
Associate Professor, Teaching Stream
campbell@cs.toronto.edu
Project Deadlines

- Original plan: project due 4 weeks after each session

- New plan:
  - No restriction on completing a project within a particular time period after each session.
  - Instead, there are three due dates by which you must submit any one of the projects:
    - December 20, 2016 6:00pm
    - January 17, 2016 6:00pm
    - February 28, 2017 6:00pm
Seminar 2 Project

- The project handout is posted:
  - http://c4m.cdf.toronto.edu/cohort1/phase3/

- Two approaches for doing your work:
  - Use the Computer Science Teaching Labs computing network.
  - Use your personal computer.

- Python3 packages to install:
  - numpy
  - matplotlib (pyplot)
JSON
JSON (JavaScript Object Notation)

- Standard data-interchange format.
- Commonly used in web programming for communication between a web browser and server.
- Example of JSON to represent a person:

```json
{
    "firstName": "John",
    "lastName": "Smith",
    "isAlive": true,
    "age": 25,
    "children": []
}
```

(Example source: Wikipedia)
Another JSON Example

```
{
  "id": 1,
  "name": "Foo",
  "price": 123,
  "tags": [ "Bar", "Eek" ],
  "stock": {
    "warehouse": 300,
    "retail": 20
  }
}
```

(Example source: Wikipedia)
Terminology from handout

- “In this project, we will work with food labels stored in a JSON format since most APIs (e.g. Open Food Facts, MyNetDiary, Spoonacular’s food API) provide detailed information in this format.”

- API: Application programming interface
  - A set of programming routines (e.g., functions) used for producing software applications.
EXPLORING MATPLOTLIB
matplotlib.pyplot

- [http://matplotlib.org/api/pyplot_api.html](http://matplotlib.org/api/pyplot_api.html)

- For this project, you will need to explore the pyplot documentation to find appropriate functions to use for the data visualization tasks.

- Demo: using pyplot to display a pie chart.
PYTHON: IMPORT AND MAIN
Example 1: without __main__

```python
def fahr_to_cels(temp):
    return (temp - 32) * 5 / 9

if __name__ == '__main__':
    t = input("Enter a temp: ")
    result = fahr_to_cels(float(t))
    print("Celsius: ", result)
```

Sample run of `temperature.py`:
Enter a temp: 102.6
Celsius: 39.22222222222222
Patient’s temp: 101.4
Fever report:  True

```python
import convert

def has_fever(c_temp):
    f_temp = convert.fahr_to_cels(c_temp)
    return f_temp > 37.5

t = input("Patient's temp: ")
result = has_fever(float(t))
print("Fever report: ", result)
```

```
convert.py

```
```
temperature.py
```

Example 1: without __main__

```python
def fahr_to_cels(temp):
    return (temp - 32) * 5 / 9

t = input("Enter a temp: ")
result = fahr_to_cels(float(t))
print("Celsius: ", result)
```

```python
import convert

def has_fever(c_temp):
    f_temp = convert.fahr_to_cels(c_temp)
    return f_temp > 37.5

t = input("Patient's temp: ")
result = has_fever(float(t))
print("Fever report: ", result)
```

```python
def fahr_to_cels(temp):
    return (temp - 32) * 5 / 9

if __name__ == '__main__':
    t = input("Enter a temp: ")
    result = fahr_to_cels(float(t))
    print("Celsius: ", result)
```

```python
import convert

def has_fever(c_temp):
    f_temp = convert.fahr_to_cels(c_temp)
    return f_temp > 37.5

t = input("Patient's temp: ")
result = has_fever(float(t))
print("Fever report: ", result)
```
Example 2: with `__main__`

```python
def fahr_to_cels(temp):
    return (temp - 32) * 5 / 9

if __name__ == '__main__':
    t = input("Enter a temp: ")
    result = fahr_to_cels(float(t))
    print("Celsius:", result)

import convert

def has_fever(c_temp):
    f_temp = convert.fahr_to_cels(c_temp)
    return f_temp > 37.5

t = input("Patient's temp: ")
result = has_fever(float(t))
print("Fever report: ", result)
```

`convert.py`

`temperature.py`
Summary

- Importing a module, executes the code in that module.
- If the module being imported contains a main block (`if __name__ == '__main__'`), the code within the main block will NOT be executed when that module is imported.
- However, when that module is run directly, both the code inside and outside of the main block is executed.
How this applies to your project

- You will write code in three files:
  - seminar2_part1.py
  - seminar2_part2.py
  - seminar2_part3.py

- To reuse functions written in part 1, you will import `seminar2_part1` in `seminar2_part2.py`.

- To prevent the user interaction code from `seminar2_part1` from being executed when that module is imported by `seminar2_part2`, place that code within a main block.
Seminar 3: Dr. Jared Simpson

- Tuesday November 22 2016 6-8pm
- Location: DCS Innovation Lab
- Topic: Informatics and Bio-computing
- https://oicr.on.ca/person/researcher/jared-simpson
FEEDBACK
Phase 3, Seminar 2 Survey

- You will receive an email with the subject “C4M: Phase 3, Seminar 2 Feedback Survey”.
- Please complete that survey now:
  - [https://www.surveymonkey.com/r/C4MSeminar2](https://www.surveymonkey.com/r/C4MSeminar2)