Computing for Medicine: Phase 3, Seminar 6 Project

Jennifer Campbell
Associate Professor, Teaching Stream
campbell@cs.toronto.edu
Seminar 6 Project

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

- No software installation required.
OVERVIEW
Starter code and data

- **Starter code:**
  - phenotips_project.py (TODOs)
  - ontology_parser.py (complete)
  - ontology_explorer.py (TODOs)

- **Data:**
  - hp.obo (Human phenotype ontology)
Your tasks

0) Explore PhenoTips using your web browser.
1) Write a program to interact with PhenoTips.
2) Write a program to get information about the Human Phenotype Ontology (HPO).
3) Q&A: revisiting design decisions; more exploration
**Type set**

- Python sets are unordered collections of unique immutable objects.
  - [https://docs.python.org/3/library/stdtypes.html#set](https://docs.python.org/3/library/stdtypes.html#set)
  - `s = set()`  # an empty set
  - `s.add(1)`
  - `s.add(2)`
  - `s2 = set([1, 2, 3, 4])`  # new set with 4 items
  - `s2.add(3)`  # 3 already in s2, so s2 is unchanged

- Note: using type `set` is not a requirement, but you may find it helpful.
Assigning parameters default values

- For certain functions, including `range` and `print`, the number of arguments that you pass to them can vary.

- For example:
  - `print('hello')`
  - `print(1, 2, 3)`
  - `print('a', 'b', 'c', end='xyz')`
  - `print(1, 2, 3, sep='..', end='!')`

- Demo: `default_parameters.py`
Recursion

- To solve a problem, identify how it can be broken down into smaller instances with the same structure.
- A *recursive function* is a function that calls itself.
- Any problem that we can solve with recursion can be solved with iteration (loops) and vice versa.
  - Some problems have simple recursive solutions and complex iterative solutions.

- Demo:
  - Searching a list; reversing a list

- Resources:
Trees can represent data that has a hierarchical structure.

A, B, C ... J are nodes.

A is the root of the tree.

A is the parent of B, C, D.

B, C, D are children of A.

E, F, G, J, I are leaf nodes (nodes with no children).
More recursion: getting leaf nodes

Demo:
`tree_recursive.py`

```
tree = {
    'A': list(['B', 'C', 'D']),
    'B': list(['E', 'F']),
    'C': list(['G', 'H']),
    'D': list(['I']),
    'H': list(['J'])
}
```
Starter code: class Ontology

- You already know how to use Python’s classes (e.g., `str` and `list`) and methods (e.g., `str.startswith` and `list.append`).
- In `ontology_parser.py`, a class named `Ontology` is defined.
- For this project, you will use class `Ontology` and call on its methods.
- You do not need to know how to define classes to complete this project. However, if you would like to learn more about object-oriented programming, you may find these videos helpful.
Ontology representation

- The starter code produces two dictionaries to represent the human phenotype ontology:
  - `pid_to_name`: each key is an ID for a phenotypic feature and each value is its name
  - `pid_to_parents`: each key is an ID for a phenotypic feature and each value is a list of its parent IDs
Example

Phenotypic abnormality
0000118

Abnormality of the immune system
0002715

Abnormality of the immune system
0010987

Abnormality of leukocytes
0001881

Abnormality of lymphocytes
0004332

Abnormality of blood and blood-forming tissues
0001871

0001871: [0000118],
0001881: [
    0010987, 0001871],
0010987: [0002715],
0001871: [0000118],
0002715: [0000118]
Tips

- Test your code as you write it, a little bit at a time.
- Use the Wing debugger, especially when implementing recursive code.
- The `Ontology` class represents the HPO using a dictionary that maps a child node to its parents, unlike the tree dictionary which mapped a parent to its children. If you prefer to have the opposite, you can write a helper function to invert the dictionary.
SUBMITTING PROJECTS
Project submission

- **Deadlines:** TODAY And February 29th by 6:00pm
- **Submission process:**
  - Submit on MarkUs:
    - [https://markus.teach.cs.toronto.edu/c4m-2016-01](https://markus.teach.cs.toronto.edu/c4m-2016-01)
  - Email Jen <campbell@cs.toronto.edu> to specify which project you submitted.
- **Getting help:**
  - Piazza ([https://piazza.com/utoronto.ca/fall2016/c4mph3](https://piazza.com/utoronto.ca/fall2016/c4mph3))
  - Email (campbell@cs.toronto.edu)
FEEDBACK
Phase 3, Seminar 6 Survey

- You will receive an email with the subject “C4M: Phase 3, Seminar 6 Feedback Survey”.
- Please complete that survey now:
  - https://www.surveymonkey.com/r/C4MSeminar6