CSC148 fall 2013
more recursion, testing
week 4

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~heap/148/F13/
416-978-5899

September 30, 2013
Outline

Class design for cheese

Recursion on nested lists

Testing, big and small
Separation of concerns

Tkinter documentation version 8.5

- Represents rules and procedures for moving cheese from stool
- Combines moves visual cheese around
- Visual rep of cheese + how it responds to events, response recorded

DomainStools → ManualController ← CheeseView

Can’t move bottom cheese - it's a stool (seat)
Define the nesting-depth of L as 1 plus the maximum nesting depth of L’s elements if L is a list, otherwise 0.

- the definition is almost exactly the Python code you write! That’s beauty of recursion!

- start by writing return and pythonese for the definition:
  
  ```python
  return 1 + max([nesting_depth(x) for x in L]) if isinstance(L, list) else 0
  ```

- deal with the special case of a non-list
maximum number in nested list

Use the built-in max much like sum

- how would you find the max of non-nested list?
  \[
  \text{max}(\ldots)
  \]

- how would you build that list using a comprehension?
  \[
  \text{max}(\ldots]\]

- what would you do with list items that were themselves lists?
  \[
  \text{max}(\text{[rec\_max(x) \ldots]})
  \]

- get some intuition by tracing through flat lists, lists nested one deep, then two deep...
before and after coding:

Test your docstring examples automatically:

```python
if __name__ == '__main__':
    import doctest
doctest.testmod()
```

For more thorough testing, use `unittest`