CSC148 winter 2015

stacks and sorts

week 5

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~csc148h/winter/
416-978-5899

February 2, 2015

Computer Science
UNIVERSITY OF TORONTO

Outline

relatively efficient sort

I want to sort `[4, 6, 2, 1, 7, 5, 3]` in ascending order.

You probably know several techniques, and could also sort seven numbers in your head without consciously applying a technique.

Here's a technique:

1. figure out where 4 goes by partitioning other numbers to its left or right
2. take the same approach with the left and right partitions...

it's recursion!

We believe the technique works for a list of seven numbers if we believe it works for the smaller lists that end up to the right and left of 4.

We also have smallest/simplest lists (base case(s)) where there are so few elements there is no need to sort. What size lists are these?

parenthesization

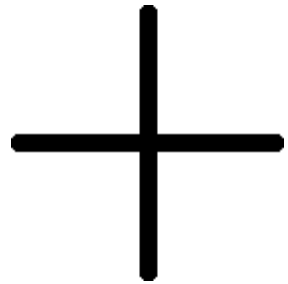
In some situations it is important that opening and closing parentheses, brackets, braces match.

'(1 + [7 - {8 / 3}])' — good

'(1 + [7 - {8 / 3]})' — bad

Remember, the computer only "sees" one character at a time.

1



7

[redacted]

8

3

}

]

]

testing...

```
def pop(self):
    ''' (Stack) -> object

    Remove and return the top item from self.
```

- How many different categories of Stack are there?
- Do you want to test them all by hand every time you change the code?
- If there were more types in the contract, would they all fit in a doctest?
- unittest!

more testing...

Use your `docstring` for testing as you develop, but use unit testing to make sure that your particular implementation remains consistent with your ADT's interface. Be sure to:

- import the module `unittest`
- subclass `unittest.Testcase` for your tests, and begin each method that carries out a test with the string `test`
- compose tests **before** and **during** implementation

even more testing...

See this week's lab **testqueue.py** for an example.