CSCA48 Exercise 9

Due: March 23, 2014. 5:00pm

A Buggy Heap

This week, we need your help. Nick and Brian wrote some code to build a Heap, and there seem to be some problems with it [1]. Your job is to fix all of the bugs in our `Heap` class, uploaded in the file `ex9_code.py`.

However, we believe that we have reverse-engineered the algorithm most students use for debugging code:

```
while(auto_mark < perfect):
    for i in range (guess_at_num_errors):
        i = Random.randint(0,num_lines_of_code - 1)
        if(code[i].doesnt_feel_right()):
           code[i] = code[i].random_change()
```

We're trying to move you towards a better algorithm. So we're instating a few rules:

- Do not look at the heap code Nick posted on his website (that wouldn't be very fun now, would it?)
- You may not add any `print` statements (we want you to get used to using the debugger, not always using the print-and-check coding.
- You can only fix a line of code once you have written a test case that exposes its error.[2]
- We will not be running the auto-marker until after the submission deadline this week.

The algorithm we're trying to get you to follow is something like:

```
while(not self.convinced(code_is_perfect)):
    while(unit_test_errors > 0):
        debugger.open()
        debugger.set_breakpoints()
        debugger.walk_through_code()
        code.find_bug()
        code.repair_bug()
    test_cases.add_new_case()
```

What to Submit

Submit the files `ex9_code.py` with your bug-free version of `Heap`, and `ex9_tests.py` with your UnitTests on MarkUs.

[1] Obviously this isn't true. Neither Nick, nor Brian would ever have bugs in their code

[2] This rule is waived for any errors that occur in the main body of the program, as it can't be tested with UnitTest. Thanks to Kenneth for pointing this out.