C M S I 3 8 6

Homework #1

Due: 2013-09-12

For the first homework assignment, you'll be warming up to Python. **Readings**: Read as many on-line tutorials or book chapters on Python as is reasonable without seriously impacting your personal life.

Produce your solutions in four files: *pythonwarmup.py*, *prefixes.py*, *lines.py*, and *pythonwarmuptest.py*. Turn in a hardcopy (8.5" × 11").

1. Write a Python function that accepts a number of U.S. cents and returns a tuple containing, respectively, the smallest number of U.S. quarters, dimes, nickels, and pennies that equal the given amount. Use the Python `divmod` built-in function.

```
>>> change(96)
(3, 2, 0, 1)
>>> change(8)
(0, 0, 1, 3)
```

2. Write a Python function that takes in a string s and returns the string which is equivalent to s but with all ASCII vowels removed. For example:

```
>>> strip_vowels("Hello, world")
"Hll, wrld"
```

3. Write a Python function that *randomly* permutes a string. By random we mean that each time you call the function for a given argument all possible permutations are equally likely (note that "random" is not the same as "arbitrary.") For example:

```
>>> scramble("Hello, world")
"w,dlroH elol"
```

4. Write a Python generator function that yields powers of two starting at 1 and going up to some limit. For example:

```
>>> for x in powers_of_two(70): print x
1
2
4
8
16
32
64
```

5. Write a Python generator function that yields powers of an arbitrary base starting at exponent 0 and going up to some limit. For example:

```
>>> for x in powers(3, 400): print x
```

```
1
3
9
27
81
243
```

6. Write a function that interleaves two lists. If the lists do not have the same length, the elements of the longer list should end up at the end of the result list. For example:

```
>>> interleave(["a", "b"], [1, 2, True, None])
["a", 1, "b", 2, True, None]
```

7. Write a function that doubles up each item in an list. For example:

```
>>> stutter([5,4,[3],9])
[5,5,4,4,[3],[3],9,9]
```

8. Write a Python script (in the file *prefixes.py*) that writes successive prefixes of its first input argument, one per line, starting with the first prefix, which is zero characters long. For example:

```
$ python prefixes.py vanrossum

v
va
van
vanr
vanro
vanros
vanross
vanrossu
vanrossum
```

9. Write a Python script (in the file *lines.py*) that reports the number of *non-blank, non-fully-commented* lines in the file named by the first argument. Blank lines are those that have either no characters or consist entirely of whitespace; commented lines are those whose first non-whitespace character is #. For example, if the file states.txt has 50 non-blank, non-comment lines:

```
$ python lines.py states.txt
50
```

10. Flesh out the unit test file for problems 1-7 that I have started here for you. (Note that you do not need to submit unit tests for problems 8 and 9.) You will be graded on how well you cover each of your functions — think zeros, negatives, extremely large numbers, etc.

```
from pythonwarmup import (change, strip_vowels, scramble, powers_of_two, powers,
    interleave, stutter)
import unittest

def anagram(s, t):
    return sorted(s) == sorted(t)

class PythonWarmupTestCase(unittest.TestCase):

    def test_change(self):
```

```
        self.assertEqual((3, 2, 0, 2), change(97))
        self.assertEqual((0, 0, 1, 3), change(8))
        self.assertEqual((10, 0, 0, 0), change(250))
        # More tests

    def test_strip_vowels(self):
        self.assertEqual('Hll, wrld', strip_vowels("Hello, world"))
        # More tests

    def test_scramble(self):
        data = ["", "a", "rat", "BSOD", "BDFL", "Python testing"]
        for s in data:
            self.assertTrue(anagram(s, scramble(s)))

    # test_powers_of_two goes here

    # test_powers goes here

    # test_interleave goes here

    # test_stutter goes here

if __name__ == '__main__':
    unittest.main()
```