Problem Statement 1: Python

Implement a web crawler for Shopping.com

Implement a robust text scraper that will connect to a page on www.shopping.com, and return a result for a given keyword. Two queries can be performed using this program. The first query is getting the total number of results for a given keyword. The second query is to find all results for a given keywords on a specified page. Handle all the exceptions gracefully any feel free to use your favorite library.

Following are the URLs
`http://www.shopping.com/products?KW=<keyword>`
`http://www.shopping.com/products~PG-<number>?KW=<keyword>"`

Queries:
Query 1: (requires a single argument)
`your_program <keyword>`

Query 2: (requires two arguments)
`your_program <keyword> <page number>`

The focus of this problem is on correctness and design.

Problem Statement 2: Python

In Python, write a class or module with a bunch of functions for manipulating a URI. For this exercise, pretend that the urllib, urllib2, and urlparse modules don't exist. You can use other standard Python modules, such as re, for this. The focus of the class or module you write should be around usage on the web, so you'll want to have things that make it easier to update or append a querystring var, get the scheme for a URI, etc., and you may want to include ways to figure out the domain for a URL (british-site.co.uk, us-site.com, mailto: yourname@example.com, etc.)

We're looking for correctness, and elegance of your API (does it let you do the things you commonly want to do with URIs in a really straightforward way?,) as well as coding style. Your code should be well-commented and documented and conform to the guidelines in the PEP 8 Style Guide for Python Code. Include some instructions and examples of usage in your documentation. You may also want to write unit tests.

Thinks to keep in mind:
- The code should be of production quality
- Follow test driven development. Your code must reflect quality
- Ensure code follows a consistent convention