Authors:
Spencer Moran
Wei Gao

1. Explain testIsValid Function of UrlValidator test code.

- Parameters: an array of test objects, and a number representing the options for the function.
    a. Function instantiates a member of the UrlValidator class passing in the options to the constructor.
    b. The program then calls the isValid function from the UrlValidator class to verify that function works correctly. It uses the google front page for this, which is a known good URL.
    c. Using a do while loop, the function then iterates through the testobjects array, that was passed into the function.
    d. In each loop it:
        i. Initializes a stringbuffer called 'testbuffer'
        ii. Validates the parts of each object in the testObjects array in a nested for loop.
        iii. Compares the expected outcome of each part constructed using the parts in the testObjects array.
        iv. Creates a new variable called url, with the contents of the testbuffer.
        v. Stores the result of a UrlValidator call to its member function isValid(), passing in the url that was just created.
        vi. If the result is true, it prints the url to the system and afterward runs an assertion ensuring that the result, and expected result are all valid for the url.
        vii. The system then reports for that iteration whether the url passed or failed the validation check using either a period ('.') to indicate a pass, or a capital X to indicate a failure.
        viii. This process continues until all the objects in the array have been processed.

2. How many total number of the URLs it is testing.

The total number of URLs varies based on the input. The result is some function of all possible permutations of the input objects. The total number of URLs will be the permutation of differents portions of URL. There are 9 schemes, 19 authorities, 7 ports, 10 paths and 3 queries. Thus, there are 9*19*7*10*3 = 35910 URLs.

3. Explain how it is building all the URLs.

Construction happens in the nested for loop near the beginning of the code. This for loop sets an index to the contents of the testPartsIndex on the current iteration of the testPartsIndexIndex (confusing, I know), casts the object currently referenced in the testObjects array to a result pair, appends the resulting part to the testing buffer, and then performs a bitwise assignment of the result of a call to the valid function of the generated part on that particular index.

Authors:
Spencer Moran
Wei Gao

4. Give an example of valid URL being tested and an invalid URL being tested by testIsValid() method.
   Valid URL: http://www.google.com:80/test1?action=view
   Invalid URL: 3ht://256.256.256.256:-1/..?action=view

5. Do you think a real world test (URL Validator's testIsValid() test in this case) is very different than the unit tests that we wrote (in terms of concepts & complexity)?
   Yes. The authors of the testIsValid function demonstrate a deep knowledge of the Java language and their chosen field. They use shortcuts that most rookie software engineers are probably unaware of. In addition they do not extensively comment out their code, but seem to adhere to the expectation that whoever is reading their code has the same level of experience that they do, and can follow the logic without needing instruction. While the logic of the function itself may be simple, it is highly condensed and self contained. Structurally, the logic is similar to the exercises we have been building in class, but in execution the logic is more elegant.