CPE 202

Lab 2: ADT Stack Implementations

The goal of this lab is to implement the Stack Abstract Data Type using two different implementations:

1) The built in List construct in Python
2) A simple linked data structure covered in class

As part of the Stack definition for this lab, we will specify a capacity for the stack. In the case of the Python List implementation, you will allocate a list of size capacity and use this to store the items in the stack. Since Lists in Python expand when more storage is needed, you must avoid using any functions that would cause a List to expand (see the list (haha) of forbidden methods/operations below). This prevents the stack from growing if the user accesses the List through the given interface. (This prevents the stack from using more space than a user might want. Think of this as a requirement for an application on a small device that has very limited storage.) For consistency, in the simple linked data structure implementation, we will also have a capacity attribute for the stack.

You are NOT allowed to use the following Python List operations (we will check!):

- `all built-in list methods including:`
  - `append()`
  - `insert()`
  - `extend()`
  - `remove()`
  - `pop()`
- `== and != between lists`
- `del`
- `in`
- `len`
- + (concatenation of lists)
- List slicing (e.g. `some_list[2:9]`)

Additional Requirements:

- All stack operations must have O(1) performance
- Your stack implementations must be able to hold values of None as valid data

The following starter files are available on GitHub:

- **`stack_array.py`**: Contains an array (Python List) based implementation of the **Stack** class
- **`stack_linked.py`**: Contains a linked based implementation of the **Stack** class
- **`stack_tests.py`:** Contains your set of thorough tests to ensure your implementations work correctly. These tests must run correctly on **ANY implementation** that follows the specification.

(Note that the class in each stack implementation is named **Stack**, and both implementations must follow the same specification. This allows one set of tests in **stack_tests.py** that can be used for both implementations by just changing which file is used when importing Stack.)