Stacks

A stack is a collection of objects where:

- If we want to take an item, we take it from the top.
- If we want to add an item, we add it to the top.

A stack is last-in, first-out (LIFO).

Uses

Stacks are useful:

- A stack of actions in a text editor allows an undo option.
- Similarly, the back button in a web browser can use a stack.
- Many languages (including Python) are implemented using a stack of active function calls.
- Evaluating arithmetic expressions without the need for brackets (via postfix notation) can be done with a stack.
- A stack can provide a quick method for reversing a sequence of input characters.

ADT (Abstract Data Type)

ADTs are a way of specifying data structures (such as stacks) precisely, without going into details of how they are implemented underneath.

Stack ADT

- `push` – place an element onto the top of the stack
- `pop` – get the first element off the top of the stack (and remove it from the stack)

- `top` – report the top element of the stack (but don't remove it)
- `length` – report how many elements are in the stack
- `is empty` – report whether or not the stack is empty

Implementation

The elements in the stack have an order to them, so using a sequence makes sense. We can use a Python list. Due to the way Python manages memory for lists, we should add and delete at the end of the list to do it most efficiently.

Code

```
class Stack(object):
    def__init__(self):
        self._alist = []

    def push(self, element):
        self._alist.append(element)

    def pop(self):
        if len(self._alist) == 0:
            return None
        return self._alist.pop()

    def top(self):
        if len(self._alist) == 0:
            return None
        return self._alist[-1]

    def length(self):
        return len(self._alist)

    def is_empty(self):
        return len(self._alist) == 0
```

Complexity

- `List.append` and `List.pop` are O(1) on average (due to Python's memory reshuffling).
    - Our `push` and `pop` methods are O(1) on average.
- List index lookup is O(1).
    - Our `top` method is O(1).
- List length is O(1).

  - Our `length` method is O(1).
  - Our `is_empty` method is O(1).