Chapter 6: Stacks/Queues

Stack: collection of objects that are inserted and removed using LIFO
→ can insert anytime but only access/remove most recent

S.push(e): add element e to the top of stack S
S.pop(): remove/return top element from stack S
S.top(): return reference w/o popping it off

Adapter Pattern

→ applies to any context where we want to modify an existing class so its methods may match those of a different (but related) class/interface
- general way to apply the pattern is to define a new class which contains an instance of the existing class as a hidden field, then implement each method of new class using hidden instance variable

→ Example using ArrayStack w/ underlying Python list for storage

```
class ArrayStack:
  def __init__(self):
    self._data = []
  def __len__(self):
    return len(self._data)
  def is_empty(self):
    return len(self._data) == 0
  def push(self, e):
    self._data.append(e)
  def top(self):
    if self.is_empty()
      raise Empty
    return self._data[-1]
  def pop(self):
    if self.is_empty
      raise Empty
    return self._data.pop()
```