Project 1: Stack Implementation

The goal of this project is to implement the Stack Abstract Data Type using the built in List construct in Python and the Stack ADT using the simple linked data structure covered in class.

As discussed in class you are to allocate a list of size stack_capacity and use this to store the items in the stack. Since Lists in python expand when more storage is needed you will have to provide a mechanism to prevent the list *inside* your stack from growing. This really prevents the stack from growing if the user only accesses it through the given interface but that is fine for the purposes of this exercise. (This prevents the stack from using more space that a user might want. Think of this as a requirement for an application on a small device than has very limited storage.) In the case when a user attempts to push an item on to a full stack, your push function (method) should raise an IndexError. Similarly if a user tries to pop an empty, your pop function (method) should raise an IndexError.

PolyLearn has a file **starter stacks.py** which provides the following as a starting point for **stacks.py**:

stacks.py

```
class StackArray:
    """Implements an efficient last-in first-out Abstract Data Type using a Python List"""

    def __init__(self, capacity):
        """Creates and empty stack with a capacity"""
        self.capacity = capacity               # Capacity of your stack
        self.items = [None]*capacity           # initializing the stack
        self.num_items = 0                     # number of elements in the stack

    def is_empty(self):
        """Returns true if the stack self is empty and false otherwise"""

    def is_full(self):
        """Returns true if the stack self is full and false otherwise"""

    def push(self, item):

    def pop(self):

    def peek(self):

    def size(self):
       """Returns the number of elements currently in the stack, not the capacity"""
```

Submit to PolyLearn two files:

- **stacks.py** containing a list based implementation of stack and a linked implementation of stack. The classes must be called: **StackArray** and **StackLinked** . Both implementations should follow the above specification and be thoroughly tested.
- **test_stacks.py** contains your set of tests to ensure you classes work correctly

Make sure that you follow the design recipe. (No need for template)

Note: Your class names, function names and files name must follow the spec of the project.