Project (Week 3)

1. Modify the ArrayStack implementation so that the stack’s capacity is limited to maxlen elements, where maxlen is an optional parameter to the constructor (that defaults to None). If push is called when the stack is at full capacity, throw a Full exception (defined similarly to Empty).
2. Stacks are often used to provide “undo” support in applications like a Web browser or text editor. While support for undo can be implemented with an unbounded stack, many applications provide only limited support for such an undo history, with a fixed-capacity stack. When push is invoked with the stack at full capacity, rather than throwing a Full exception above, a more typical semantic is to accept the pushed element at the top while “leaking” the oldest element from the bottom of the stack to make room.

Give an implementation of such a LeakyStack abstraction, using a circular array (or just the Python list in the above ArrayStack) with appropriate storage capacity. This class should have a public interface similar to the bounded-capacity stack in the first half of this project, but with the desired leaky semantics when full.

**Submit your code together with the run results.**