Daily Coding Problem #154

Problem

This problem was asked by Amazon.

Implement a stack API using only a heap. A stack implements the following methods:

- `push(item)`, which adds an element to the stack
- `pop()`, which removes and returns the most recently added element (or throws an error if there is nothing on the stack)

Recall that a heap has the following operations:

- `push(item)`, which adds a new key to the heap
- `pop()`, which removes and returns the max value of the heap

Solution

One way to solve this problem is to store a timestamp as the keys to a max-heap. This way, the most recently added item will always be at the top, and we can extract it by calling `pop()` on the heap. This should cause the heap to bubble-down and bring the next most recently added item to the top.

```
from time import time

class Stack:
    def __init__(self):
        self.max_heap = MaxHeap()

    def push(self, item):
        t = time()
        self.max_heap.push(item, t)
```

```
    def pop(self):
        item, _ = self.max_heap.pop()
        return item


import heapq

class MaxHeap:
    def __init__(self):
        self._heap = []

    def push(self, item, priority):
        heapq.heappush(self._heap, (-priority, item))

    def pop(self):
        _, item = heapq.heappop(self._heap)
        return item
```

The stack operations will take O(log n) time and O(n) space.