Binary Heap Operations

The basic operations we will implement for our binary heap are as follows:

- `BinaryHeap()` creates a new, empty, binary heap.
- `insert(k)` adds a new item to the heap.
- `findMin()` returns the item with the minimum key value, leaving item in the heap.
- `delMin()` returns the item with the minimum key value, removing the item from the heap.
- `isEmpty()` returns true if the heap is empty, false otherwise.
- `size()` returns the number of items in the heap.
- `buildHeap(list)` builds a new heap from a list of keys.

ActiveCode 1 demonstrates the use of some of the binary heap methods. Notice that no matter the order that we add items to the heap, the smallest is removed each time. We will now turn our attention to creating an implementation for this idea.

Run Save Load

```
1 from pythonds.trees.binheap import BinHeap
2
3 bh = BinHeap()
4 bh.insert(5)
5 bh.insert(7)
6 bh.insert(3)
7 bh.insert(11)
8
9 print(bh.delMin())
10
11 print(bh.delMin())
12
13 print(bh.delMin())
14
15 print(bh.delMin())
16
```

ActiveCode: 1 Using the Binary Heap (heap1)

❮ (PriorityQueueswithBinaryHeaps.html) ❯ (BinaryHeapImplementation.html)

Mark as completed

(QueueswithBinaryHeaps.html)

Next Section - Binary Heap Implementation (Bina