1 Min-nie Mouse and Max-xie Mouse

Use or modify a data structure or set of data structures to do the following operations in the provided time:

- insert: O(log(N))
- getSmallest: O(1)
- getLargest: O(1)
- deleteSmallest: O(log(N))
- deleteLargest: O(log(N))

Solution:
We will fulfill all of our requirements by using a basic Balanced Search Tree. By construction, Balanced Search Trees insert in O(log(N)) time and can also delete in O(log(N)) time. As a result, the only operations that we need to account for is getSmallest and getLargest. We can trivially keep a pointer that keeps track of the smallest element and the largest element. Whenever we delete the smallest element or update the Balanced Search Tree to have a smallest element, we will update our pointer. We can do a similar process for the largest element. The reason why 2-3 trees are not used in the place of normal min-heaps or max-heap are really easy to implement, as they can be visualized using arrays. This also saves a lot of memory compared to Balanced Search Trees which can tend to be a bit expensive.
Explanation: