性质和排序方式：

堆： 堆是一个完全二叉树，分为最大堆和最小堆。在最大堆中，每个节点的值都大于或等于其子节点的值；在最小堆中，每个节点的值都小于或等于其子节点的值。堆主要用于实现优先队列。

二叉搜索树： BST是一棵二叉树，其中每个节点的值大于其左子树中的所有节点的值，小于其右子树中的所有节点的值。BST用于实现有序集合，支持高效的查找、插入和删除操作。

平衡性：

堆： 堆不要求有序性，只要满足堆的性质即可。堆中的节点相对于它们的兄弟节点没有大小顺序关系，因此堆并不保证左右子树的平衡。

二叉搜索树： BST要求有序性，左子树小于根节点，右子树大于根节点。为了保持平衡性，通常会使用平衡二叉搜索树（如 AVL 树、红黑树）。

操作复杂度：

堆： 堆主要用于高效地找到和移除具有最大或最小关键字的元素，插入操作相对简单。堆排序是一种基于堆的排序算法，其时间复杂度为 O(n log n)。

二叉搜索树： 在平均情况下，BST的查找、插入和删除操作的时间复杂度为 O(log n)，但在最坏情况下可能达到 O(n)。为了避免最坏情况，可以使用平衡二叉搜索树。

应用场景：

堆： 适用于优先队列、堆排序等场景，其中对最大或最小值的快速访问非常重要。

二叉搜索树： 适用于需要支持高效查找、插入和删除操作的场景，如有序集合的实现。