1. **Trie**: This is a tree-like data structure used for efficient retrieval of a key in a large dataset of strings.
2. **AVL Tree**: It's a self-balancing binary search tree where the heights of the two child subtrees of any node differ by at most one.
3. **Red-Black Tree**: Another self-balancing binary search tree where each node has an extra bit for denoting the color (red or black).
4. **B-Tree**: A self-balancing tree data structure that maintains sorted data and allows searches, sequential access, insertions, and deletions in logarithmic time.
5. **Heap**: A specialized tree-based data structure that satisfies the heap property. It could be a max heap or a min heap.

As for sorting algorithms, you might come across:

1. **Merge Sort**: A divide and conquer algorithm that divides the input array into two halves, sorts each half, and then merges them.
2. **Quick Sort**: Another divide and conquer algorithm that picks an element as a pivot and partitions the array around the pivot.
3. **Heap Sort**: It's a comparison-based sorting algorithm that uses a binary heap data structure.
4. **Counting Sort**: An integer sorting algorithm that operates by counting the number of occurrences of each unique element in the input array.
5. **Radix Sort**: A non-comparison-based sorting algorithm that sorts integers by grouping them by individual digits.

**Instructions**

* Test and try the interactive schemes of each data structure & sorting algorithms
* List the name of the data structures  & sorting algorithms that you are not  familiar yet and why you still not mastering .
* Share with your instructor.