A B-tree of order N is a tree with:

1. Except for (possibly) the root, each node has at least N data keys and at most 2N data keys. Every internal node has one more link than the number of data items. I.e., internal nodes have at least N+1 links to child nodes and at most 2N+1 links to child nodes.

   A 2-3 tree is a B-tree of order 1. Each tree node has at most 2 data items and at most 3 links to child nodes, thus the designation 2-3.

   We can consider B-trees of higher orders. For example, a node of a B-tree of order 2 will have at least 2 data items in each node (except possibly the root) and at most 4 data items.

2. All the leaves are on the same level in a B-tree. The tree is always perfectly balanced.

3. A B-tree has an extended search property, analogous to binary search trees. I.e., the data keys are ordered within each node, and (conceptually) the links to sub-trees are positioned between the data keys.

   If a node contains a sequence D1 P D2, where D1, and D2 are data keys, and P is a pointer to a sub-tree, then all the data found in the subtree P will be between the key values D1 and D2.

4. Searching a B-tree proceeds by recursively examining at most 2N data items in the current node. Search starts at the root node. As we process each node encountered, exactly one of the following must occur:

   a. The B-tree is empty (link is NULL). The search target is not in the tree, OR

   b. the search target will be found among the 2N data items, OR

   c. the search target will be either:
      i) before the first key
      ii) between two data
      iii) after the last data key

      In any of i), ii), or iii), a link to a sub-tree is identified, and the search can proceed by following the link to the sub-tree.

5. In higher order B-trees, there are many more sub-trees branching off of each node than just the two sub-trees that we have in binary search trees (or AVL trees). As a result, higher order B-trees tend to be broad and shallow. B-trees are especially useful in file systems, since when accessing disk:

   a. It is time consuming to position the read head and wait for the rotational delay for each read operation. Positioning the read head, and awaiting the rotational delay is known as a ``disk seek'', or just ``seek'' operation.

   b. After the correct position on the disk is located, a read operation can quickly obtain a significant amount of data, e.g., at least an entire B-tree node.