Our implementation of the B+ Tree Index makes it easier to insert and search through a large sample of data. In this project, we assume that all attributes are integers. Recursion is used to improve efficiency.

Each node in the tree is a page. All nodes are of two types: non-leaf and leaf. Leaves contain the actual data, and non-leaves contain keys to assist with searching. The very top non-leaf is the root node. Each node, non-leaf or leaf, has a field called parent, which is the node that points to it. The parent field of the root node is 0. Each node, non-leaf of leaf has a field called key\_count, which stores the number of keys currently in the node. During insertion of an attribute, the algorithm first recursively finds the correct place for the attribute in the tree, and if the node is full, the algorithm recursively splits nodes according the parent field. Each page is pinned when any information or modification of that page is needed, and is unpinned after the program done with using or modifying the page. The efficiency of our implementation is decent. The algorithm uses recursion based on key values, and therefore unnecessary traversing is avoided, so a huge amount of time is saved.