Assignment 7

Design and Implementation of an Order-m B-tree

Manan Patel

1. Tree Design

a. Class `BTreeNode`

This class represents a node within the B-tree and includes the following key variables:

- `self.keys`: A list storing the keys in the node, with a maximum of *m-1* keys per node.
- `self.children`: A list of references to child nodes, with up to m children per node.
- `self.t`: The order of the B-tree, defining the range for the number of keys in each node.
- `self.leaf`: A boolean indicating whether the node is a leaf (contains no child nodes).

b. Class `BTree`

This class represents the B-tree structure and includes the following main variables:

- `self.root`: The root node of the B-tree, initialized as a `BTreeNode`.
- `self.t`: The order of the B-tree, which is passed down to nodes when they are created.

2. Main Operations

Insertion

- Inserts a key into a non-full node.
- If the node is a leaf, the key is directly inserted.
- If a split occurs during the insertion, uses the `splitChild` method.

Deletion

- Removes a specified key from the B-tree.
- Calls the `fill` method if a node has fewer than the minimum required keys.
- Balances the tree by borrowing from a sibling or merging nodes if necessary.

Display

- Outputs the structure of the B-tree level by level.

3. Testing

Testing was performed as per assignment requirements:

- Used orders $m = 4$ and $m = 5$.
- Inserted integers sequentially from 1 to 20.
- Deleted even integers from 2 to 20 and verified the structure.