COP 5536: B+ Tree Report

Name: Preetham Dasari
Email:preethamdasari@ufl.edu
UFID: 69698425

Introduction:

A **B+-tree** is a self-balancing tree data structure that maintains sorted data and allows searches, sequential access, insertions, and deletions in logarithmic time. The B-tree is a generalization of a binary search tree in that a node can have more than two children. Unlike other self-balancing binary search trees, the B-tree is well suited for storage systems that read and write relatively large blocks of data, such as discs. It is commonly used in databases and file systems.

Code Explanation:

I have created two classes BNode and BTree. The BTree object consists of the root pointer to the B+ Tree and the order of the B+Tree. And the BNode consists of the (key, value) pairs, pointers to the children, left sibling, right sibling, and the parent node. The Key-value pairs are stored in a structure called KVPair which consists of key of integer type and value of double type.

Installation:

Run make in the directory to create bplustree executable. Provide bplustree the path to input file with queries and it will dump output into output.txt in the same directory. Supported types for (key, value) pairs in this B+ tree are (integer, double)

Prototypes of Functions:

BNode Class:

```
class BNode{
    public:
        int degree;
        bool root;
        bool children;
        bool leaf;
        bool element;
        vector<KVPair> keys;
        vector<BNode*> pointers;
        BNode *LEFT;
        BNode *RIGHT;
        BNode *parent;

       #Methods

       BNode(int m) /*Constructor, m is the order of the B+ tree*/
       BNode* getPointer(int key)
                void pprint()
       bool isfull()
       void insert_element(KVPair a)
       void insert_pointer(BNode *a)
       void makeRoot()
       void makeLeaf()
       void makeElement()
       bool isRoot()bool isLeaf()
       bool hasChildren()bool isLeaf()
       bool isElement(){
```

```
       bool isLeaf()
       }
```

BTree Class:

```
class BTree{
    public:
    int order;
    BNode *root;

    BTree(int order)
    void clear_order()
    void set_order(int order
    void tester(KVPair x)
    string search(int k)
    string rsearch(int k1,int k2)
    void non_leaf_split(BNode *a)
    void leaf_split(BNode *a)
    BNode* getLeaf(BNode* root, KVPair x)
    void insert(KVPair x)
    void delete_N(KVPair x)
}
```

Structure for Key-Value Pair:

```
struct KVPair{
    int key;
    double value;
    KVPair(){}
    KVPair(int a,double b)

};
```