2023 秋“大数据软件设计与实践”课程报告

姓名：　刘俊杉
学号：　2021112078

1.　软件功能

实现 B+树的基本操作，包括点查找索引项、区间查找、插入索引项、删除索引项。实现 B+树的可视化展示。

2.　设计方案

1. 定义 B+树的数据结构。
2. 实现 B+树的基本操作，包括点查找索引项、区间查找、插入索引项、删除索引项。
3. 实现 B+树的可视化展示。

3.　软件实现

插入节点

```
def search(self, element: int):
    is_leaf = len(self.children) == 0
    idx = 0

    for key in self.keys:
        if element == key and is_leaf:
            return key
        elif element < key:
            break
        elif element == key:
            idx += 1
            break
        idx += 1

```

```python
    if self.children:
        return self.children[idx].search(element)
    else:
        return -1
```

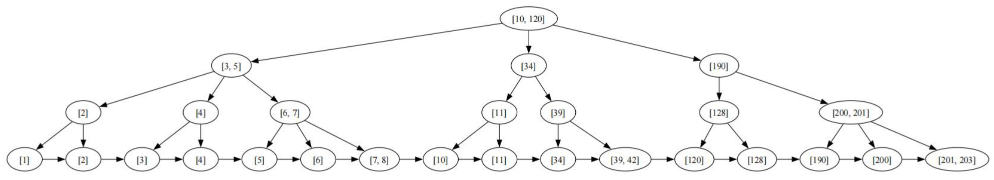
插入节点

```python
def insert(self, element: int):
    insert_idx = 0

    for key in self.keys:
        if element < key:
            break
        insert_idx += 1

    if self.children:
        self.children[insert_idx].insert(element)
    else:
        self.keys = self.keys[:insert_idx] + [element] + self.keys[insert_idx:]

        if len(self.keys) == self.order:
            self.promote()
```

融合节点

```python
def promote(self):
    is_leaf = len(self.children) == 0

    middle_idx = math.floor((len(self.keys) - 1) / 2)
    middle_key = self.keys[middle_idx]
    copy_idx = middle_idx if is_leaf else middle_idx + 1

    right_tree = BPlusTreeNode(self.order)
    right_tree.keys = self.keys[copy_idx:]
    right_tree.children = self.children[copy_idx:]
    # Update parents on the right tree
    for c in right_tree.children:
        c.parent = right_tree

    self.keys = self.keys[:middle_idx]
    self.children = self.children[: copy_idx]

    if is_leaf:
        right_tree.next_node = self.next_node
        self.next_node = right_tree
    else:
        self.next_node = None

    if not self.parent:
        self.parent = BPlusTreeNode(self.order)
        self.parent.children = [self]

    right_tree.parent = self.parent

    insert_idx = self.parent.children.index(self)
    self.parent.keys = (
            self.parent.keys[:insert_idx] + [middle_key] + self.parent.keys[insert_idx:]
    )
    self.parent.children = (
            self.parent.children[: insert_idx + 1]
            + [right_tree]
            + self.parent.children[insert_idx + 1:]
    )

    # Parentes são promovidos de forma recursiva
    if len(self.parent.keys) == self.order:
        self.parent.promote()
```

4. 软件界面


[10, 120]
[3, 5]
[34]
[190]
[2]
[4]
[6, 7]
[11]
[39]
[128]
[200, 201]
[1]
[2]
[3]
[4]
[5]
[6]
[7, 8]
[10]
[11]
[34]
[39, 42]
[120]
[128]
[190]
[200]
[201, 203]