2-3 trees

DATE: PAGE:

```
Class Node:
    def __init__(self, data, par=None)
    self.data = list(data)
    self.par = par
    self.child = list()

    def add(self, newnode):
        self.data.extend(newnode.data)
        self.data.sort()
        self.child.extend(newnode.child)
        if len(self.child) > 1:
            self.child.sort()
        if len(self.data) > 2:
            self.split()

    def insert(self, node)

        if isleaf():
            self.add(newnode)
        elif newnode.data[0] > self.data[-1]
            self.child[-1].insert(newnode)
        else
            for i in range(0, len(self.data)):
            if node.data[0] < self.data[i]:
            self.child[i].insert(node)
            break.

    def split(self, item):
        if item in
        left = Node(self.data[0])
        right = Node(self.data[2])
```

DATE: PAGE:

```
if self.child:
    self.child[0].parent = leftchild
    self.child[1].parent = left
    self.child[2].parent = right
    self.child[3].parent = right

self.child = [leftchild]
self.child.append(rightchild)
self.data = [self.data[1]]

def delete(self, element):
    pos = self.find(element)
    if pos == False:
        return
    if pos.isleaf():
        pos.data.remove(element)
    elif len(pos.data) == 1:
        if len(pos.child) == 2 & pos.child[1] != None:
            val = pos.child[0].data
            pos.child[0] = None
            pos.data.extend(val)
            pos.data.remove(element)
            pos.data.sort()
            return

    elif len(pos.child) == 2 and pos.child[1] != None:
        val = pos.child[1].data
        pos.data.extend(val)
        pos.child[1] = None
        if len(pos.data) > 2
            pos.split()
```