Project 03 : Trees, Trees
and more Trees
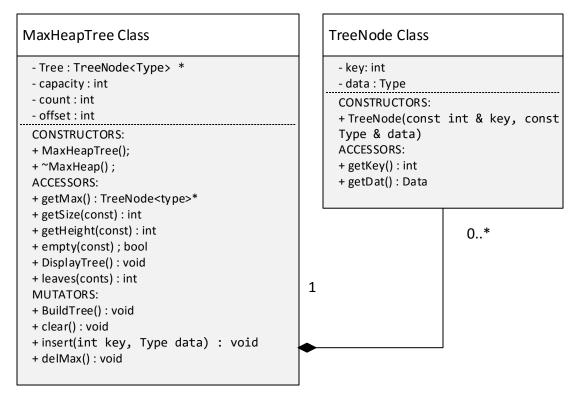
General Tree UML
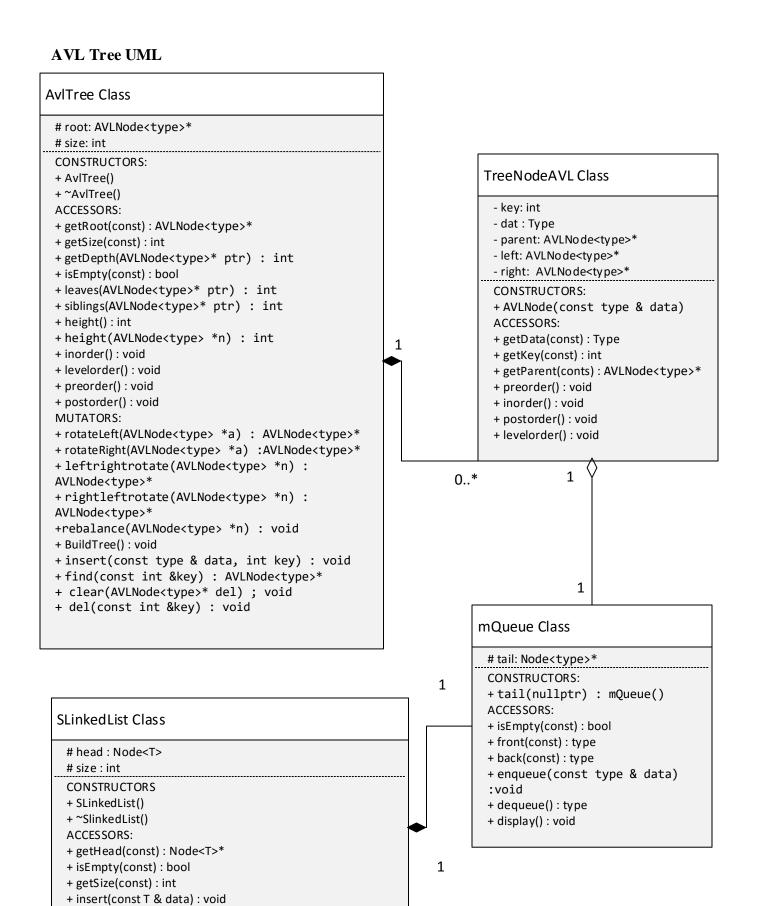
LinkedTree Class

root : TreeNodeG<type>*
visit : TreeNode<void>
arr_data : TreeNodeG<type>*

CONSTRUCTORS:
+ LinkedTree() ;
+ ~LinkedTree();
ACCESSORS:
+ getRoot() : TreeNodeG<type>*
+ getSize() : int
+ getHeight() : int
+ getHeight(TreeNodeG<type>* ptr, int h) : int
+ getDepth(TreeNodeG<type>* ptr , int h) : int
+ leaves(TreeNodeG<type>* ptr, int h) : int
+ siblings(TreeNodeG<type>* ptr) : int
+ findCommonAncestor(TreeNodeG<type>* ptr1, TreeNodeG<type>* ptr2) : TreeNodeG<type>*
+ findParent(int key_num) : TreeNodeG<type>*
+ findNote(type data) : TreeNodeG<type>*
+ preorder(TreeNodeG<type>* ptr) : void
+ postorder(TreeNodeG<type>* ptr) : void
+ levelorder(TreeNodeG<type> *ptr) : void
MUTATORS:
+ insert(const type & data, TreeNodeG<type> *parent) ; void
+ clear() : void
+ del(type data) ; void

Node Class

+ data : Type
+ next : Node
+ id: int

CONSTRUCTORS:
+ Node(const Type & data, TreeNodeG * parent)

0..*

1

SLinkedList Class

head : Node<T>
size : int

CONSTRUCTORS:
+ SLinkedList()
+~SlinkedList()
ACCESSORS:
+ getHead(const) : Node<T>*
+ isEmpty(const) : bool
+ getSize(const) : int
+ insert(const T & data) : void
+ printList(const) : void
MUTATORS:
+ del(const T & data) : void

1

0..*

0..*

TreeNodeG Class

key : int
data : type
parent : TreeNodeG*
children : SLinkedList<TreeNodeG*>

CONSTRUCTORS:
+ TreeNodeG(const Type & data, TreeNodeG * parent)
ACCESSORS:
+ getKey(const) : int
+ getData(const) : type
+ getParent(const) : TreeNodeG*

1

<<Team 22>> **Mayra Lanza**
Giovany Gonzalez
Andrew Walton

Heap Tree UML

MaxHeapTree Class

- Tree : TreeNode<Type> *
- capacity : int
- count : int
- offset : int

CONSTRUCTORS:
+ MaxHeapTree();
+ ~MaxHeap() ;
ACCESSORS:
+ getMax() : TreeNode<type>*
+ getSize(const) : int
+ getHeight(const) : int
+ empty(const) ; bool
+ DisplayTree() : void
+ leaves(conts) : int
MUTATORS:
+ BuildTree() : void
+ clear() : void
+ insert(int key, Type data) : void
+ delMax() : void

TreeNode Class

- key: int
- data : Type

CONSTRUCTORS:
+ TreeNode(const int & key, const Type & data)
ACCESSORS:
+ getKey() : int
+ getDat() : Data

0..*

1

AVL Tree UML

AvlTree Class

root: AVLNode<type>*
size: int

CONSTRUCTORS:
+ AvlTree()
+ ~AvlTree()
ACCESSORS:
+ getRoot(const) : AVLNode<type>*
+ getSize(const) : int
+ getDepth(AVLNode<type>* ptr) : int
+ isEmpty(const): bool
+ leaves(AVLNode<type>* ptr) : int
+ siblings(AVLNode<type>* ptr) : int
+ height() : int
+ height(AVLNode<type> *n) : int
+ inorder() : void
+ levelorder() : void
+ preorder() : void
+ postorder() : void
MUTATORS:
+ rotateLeft(AVLNode<type> *a) : AVLNode<type>*
+ rotateRight(AVLNode<type> *a) :AVLNode<type>*
+ leftrightrotate(AVLNode<type> *n) : AVLNode<type>*
+ rightleftrotate(AVLNode<type> *n) : AVLNode<type>*
+rebalance(AVLNode<type> *n) : void
+ BuildTree() : void
+ insert(const type & data, int key) : void
+ find(const int &key) : AVLNode<type>*
+ clear(AVLNode<type>* del) ; void
+ del(const int &key) : void

TreeNodeAVL Class

- key: int
- dat : Type
- parent: AVLNode<type>*
- left: AVLNode<type>*
- right: AVLNode<type>*

CONSTRUCTORS:
+ AVLNode(const type & data)
ACCESSORS:
+ getData(const) : Type
+ getKey(const) : int
+ getParent(conts) : AVLNode<type>*
+ preorder() : void
+ inorder() : void
+ postorder() : void
+ levelorder() : void

1

0..*

1

1

1

mQueue Class

tail: Node<type>*

CONSTRUCTORS:
+ tail(nullptr) : mQueue()
ACCESSORS:
+ isEmpty(const) : bool
+ front(const) : type
+ back(const) : type
+ enqueue (const type & data) :void
+ dequeue() : type
+ display() : void

1

SLinkedList Class

head : Node<T>
size : int

CONSTRUCTORS
+ SLinkedList()
+ ~SlinkedList()
ACCESSORS:
+ getHead(const) : Node<T>*
+ isEmpty(const) : bool
+ getSize(const) : int
+ insert(const T & data) : void
+ printList(const) : void
MUTATORS:
+ del(const T & data) : void

1

1