Class WAVLTree

java.lang.Object
    WAVLTree

```
public class WAVLTree
extends java.lang.Object
```

WAVLTree An implementation of a WAVL Tree. (Haupler, Sen and Tarajan '15)

Nested Class Summary

Nested Classes

Modifier and Type	Class and Description
static class	**WAVLTree.WAVLNode** public class WAVLNode

Constructor Summary

Constructors

Constructor and Description
WAVLTree() CTOR

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
int	**delete**(int k) public int delete(int k) deletes an item with key k from the binary tree, if it is there; the tree must remain valid (keep its invariants).
boolean	**empty**() public boolean empty()
WAVLTree.WAVLNode	**getRoot**()

	public WAVLNode getRoot() Returns the root WAVL node, or null if the tree is empty O(1)
java.lang.String[]	**infoToArray**() public String[] infoToArray() Returns an array which contains all info in the tree, sorted by their respective keys, or an empty array if the tree is empty.
int	**insert**(int k, java.lang.String value) inserts an item with key k and info i to the WAVL tree.
int[]	**keysToArray**() Returns a sorted array which contains all keys in the tree, or an empty array if the tree is empty.
java.lang.String	**max**() public String max() Returns the info of the item with the largest key in the tree, or null if the tree is empty O(log n)
java.lang.String	**min**() public String min() Returns the info of the item with the smallest key in the tree, or null if the tree is empty O(log n)
int	**minKey**()
java.lang.String	**search**(int key) public String search(int k)
java.lang.String	**select**(int i) public int select(int i) Returns the value of the i'th smallest key (return -1 if tree is empty) Example 1: select(1) returns the value of the node with minimal key Example 2: select(size()) returns the value of the node with maximal key Example 3: select(2) returns the value 2nd smallest minimal node, i.e the value of the node minimal node's successor
int	**size**() public int size() Returns the number of nodes in the tree.
java.lang.String	**toString**()

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

WAVLTree

public WAVLTree()

CTOR

Method Detail

empty

```
public boolean empty()
```

public boolean empty()

search

```
public java.lang.String search(int key)
```

public String search(int k)

insert

```
public int insert(int k,
                  java.lang.String value)
```

inserts an item with key k and info i to the WAVL tree. the tree must remain valid (keep its invariants). returns the number of rebalancing operations, or 0 if no rebalancing operations were necessary. returns -1 if an item with key k already exists in the tree.

Parameters:

k -

value -

Returns:

rebalnce O (log n)

delete

```
public int delete(int k)
```

public int delete(int k) deletes an item with key k from the binary tree, if it is there; the tree must remain valid (keep its invariants). returns the number of rebalancing operations, or 0 if no rebalancing operations were needed. returns -1 if an item with key k was not found in the tree. O(log n)

Parameters:

k - - key

Returns:

rebalncing

min

```
public java.lang.String min()
```

public String min() Returns the info of the item with the smallest key in the tree, or null if the tree is empty O(log n)

Returns:

minKey

```
public int minKey()
```

Returns:

min key O (log n)

max

```
public java.lang.String max()
```

public String max() Returns the info of the item with the largest key in the tree, or null if the tree is empty O(log n)

Returns:

keysToArray

```
public int[] keysToArray()
```

Returns a sorted array which contains all keys in the tree, or an empty array if the tree is empty. O(n)

Returns:

array contain the keys

infoToArray

```
public java.lang.String[] infoToArray()
```

public String[] infoToArray() Returns an array which contains all info in the tree, sorted by their respective keys, or an empty array if the tree is empty. O(n)

Returns:

array contain the values

size

```
public int size()
```

public int size() Returns the number of nodes in the tree. O(1)

Returns:

`size`

getRoot

```
public WAVLTree.WAVLNode getRoot()
```

public WAVLNode getRoot() Returns the root WAVL node, or null if the tree is empty O(1)

Returns:

`root`

select

```
public java.lang.String select(int i)
```

public int select(int i) Returns the value of the i'th smallest key (return -1 if tree is empty) Example 1: select(1) returns the value of the node with minimal key Example 2: select(size()) returns the value of the node with maximal key Example 3: select(2) returns the value 2nd smallest minimal node, i.e the value of the node minimal node's successor

Parameters:

`i - O(n) - we can do it better by const`

Returns:

`value`

toString

```
public java.lang.String toString()
```

Overrides:

`toString in class java.lang.Object`