Node Class Reference

Public Member Functions

	Node () Default Constructor
	Node (**Node** *c, **Node** *s, int k, int d, **Node** *p=nullptr) Constructor Used to create a node for the Binomial Heap More...
	Node (int k) Constructor to create a node with key k More...
	Node (**Node** ***child**, **Node** ***left**, **Node** ***right**, int value, int **degree**, bool **marked**=false, **Node** ***parent**=nullptr)
void	**setParent** (**Node** *n) Setter for parent More...
Node *	**getParent** () Getter for Parent More...
void	**setChild** (**Node** *n) Setter for the child of the node More...
Node *	**getChild** () Getter for the child of the node More...
void	**setSibling** (**Node** *n) Setter for the Sibling More...
Node *	**getSibling** () Getter for the Sibling More...
void	**setKey** (int k) Setter to set the key of the node More...
int	**getKey** () Getter for the key attribute More...
void	**setDegree** (int d) Setter for the degree More...
int	**getDegree** () Getter for the degree of the node More...
bool	**getMarked** () gets the mark of a node (for Fibonacci Heaps) More...
Node *	**getRight** () returns the node to the right of node x (for Fibonacci Heaps) More...
Node *	**getLeft** () returns the node to the left of node x (for Fibonacci Heaps) More...
void	**setRight** (**Node** *) Sets the right of node x to a new node n More...
void	**setLeft** (**Node** *) Sets the left of node x to a new node n More...
void	**setMarked** (bool) Sets mark of node x More...
bool	**operator<** (const **Node** &a)

Overloading Operator '<' to be able to compare the keys of two nodes without using getters. It returns true if key of node x is less than key of node a, else it returns false More...

bool **operator>** (const **Node** &a)

Private Attributes

Node * **parent**
The parent of the node

Node * **child**
The child of the node

Node * **sibling**
The sibling of the node

int **key**
The key of the node

int **degree**
The degree of the node

Node * **right**
Node right to node x

Node * **left**
Node left to node x

bool **marked**
Mark of node x

Constructor & Destructor Documentation

◆ Node() [1/2]

```
Node::Node ( Node * c,
             Node * s,
             int    k,
             int    d,
             Node * p = nullptr
           )
```

Constructor Used to create a node for the Binomial Heap

Parameters

c Child of the node

s Sibling of the node

k Key of the node

d Degree

p **Node**'s Parent

◆ Node() [2/2]

Node::Node (int k)

Constructor to create a node with key k

Parameters

k Key

Member Function Documentation

◆ getChild()

Node * Node::getChild ( )

Getter for the child of the node

Returns

A pointer to the child of the current node

◆ getDegree()

int Node::getDegree ( )

Getter for the degree of the node

Returns

the degree of the node

◆ getKey()

int Node::getKey ( )

Getter for the key attribute

Returns

the key of the node

◆ getLeft()

Node * Node::getLeft ( )

returns the node to the left of node x (for Fibonacci Heaps)

Returns

◆ getMarked()

bool Node::getMarked ( )

gets the mark of a node (for Fibonacci Heaps)

Returns

◆ getParent()

Node * Node::getParent ( )

Getter for Parent

Returns

A pointer to the parent node of the current node

◆ getRight()

Node * Node::getRight ( )

returns the node to the right of node x (for Fibonacci Heaps)

Returns

◆ getSibling()

Node * Node::getSibling ( )

Getter for the Sibling

Returns

A pointer to the sibling of the current node

◆ operator<()

bool Node::operator< (const **Node** & a)

Overloading Operator '<' to be able to compare the keys of two nodes without using getters. It returns true if key of node x is less than key of node a, else it returns false

Parameters

a **Node** a, whose key is to be compared with the key of node x

Returns

◆ operator>()

bool Node::operator> (const **Node** & a)

Overloading Operator '>' to be able to compare the keys of two nodes without using getters It returns true if key of node x is greater than key of node a, else it returns false

Parameters

a **Node** a, whose key is to be compared with the key of node x

Returns

◆ setChild()

void Node::setChild (**Node** * n)

Setter for the child of the node

Parameters

n node n to be set as child to the current node

◆ setDegree()

void Node::setDegree (int d)

Setter for the degree

Parameters

d degree to be set

◆ setKey()

void Node::setKey (int k)

Setter to set the key of the node

Parameters

k Key to be set for the current node

◆ setLeft()

void Node::setLeft (**Node** *)

Sets the left of node x to a new node n

Parameters

n **Node** set as the left node of node x

◆ setMarked()

void Node::setMarked (bool)

Sets mark of node x

Parameters

mark a boolean value to either set or reset the mark of a node

◆ setParent()

void Node::setParent (**Node** * n)

Setter for parent

Parameters

n **Node** to be set as parent of the current node

◆ setRight()

void Node::setRight (**Node** *)

Sets the right of node x to a new node n

Parameters

n **Node** set as the right node of node x

◆ setSibling()

void Node::setSibling (**Node** * n)

Setter for the Sibling

Parameters

n **Node** n to be set as the sibling of the current node

The documentation for this class was generated from the following file:

- **node.hpp**