Splay Trees

Final presentation - Algorithms and Data Structures

Juan Esteban Ladino & Joseph Mancera

May, 2019

Motivation

What is a splay tree?

- Is a type of binary tree.
- Is a type of BST with additional properties.
- Main Feature: last access node move to **root**.

Motivation

Who and how?

- The splay tree was invented by Daniel Sleator and Robert Tarjan in 1985
- Wireless Multimedia Sensor Network (Zhang, et. al. , 2012)
- System Simulator and Memory-Aware Splay Tree for In-Memory Databases in Hybrid Memory Systems (SAP SE , 2018)
- Predictive Modeling In Event Processing Systems For Big Data Processing In Cloud (Hybris AG , 2019)

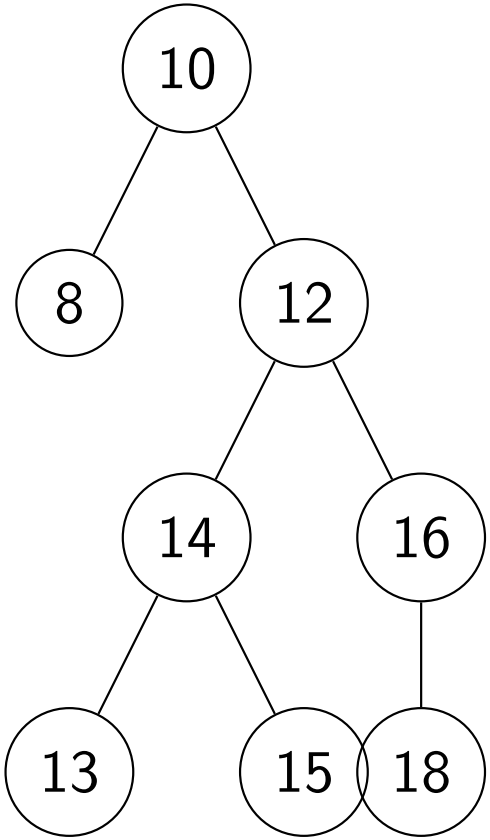
Theoretical Framework

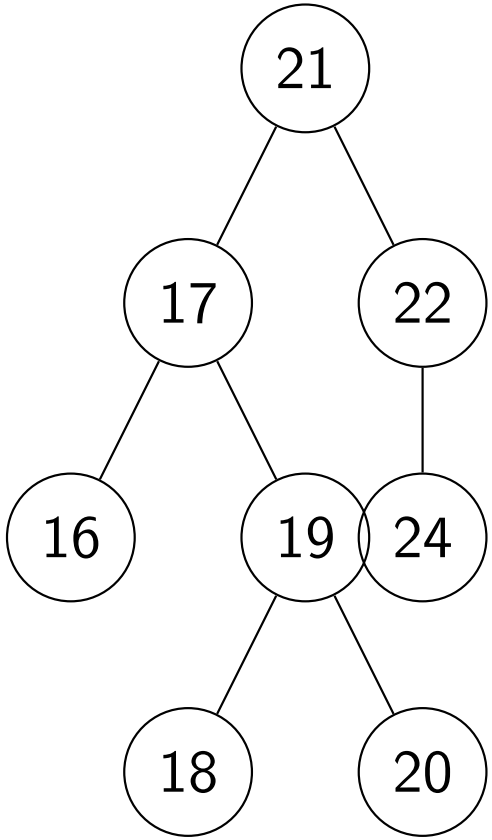
Concepts

- What is a Data Structure ?
  - The form to storing data in computer
- What is a Tree Structure ?
  - Is a collections of nodes with the properties.
    - ★ Have a node called **root** at the top of the tree.
    - ★ Other nodes are connected to the root via a single line path.
- What is Binary Search Tree?
  - Binary tree
  - For all node S the left Sub-Tree contains lower values than S.
  - For all node S the right Sub-Tree contains higher values than S.

Theoretical Framework

Binary Search Tree





Theoretical Framework

Splay trees

- Retaking the definition ....

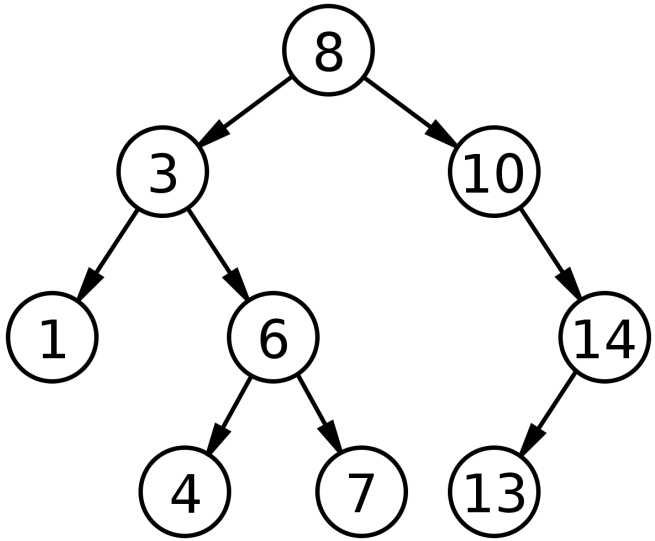


Figure: Caption

Implementation

Generalities

- It was implemented with programming language C++.
- Class Prototype: **template** <**typename** dataType>
  ——————— **class** SplayT
- Public Methods of the class: **find, insert, Remove, empty, size**

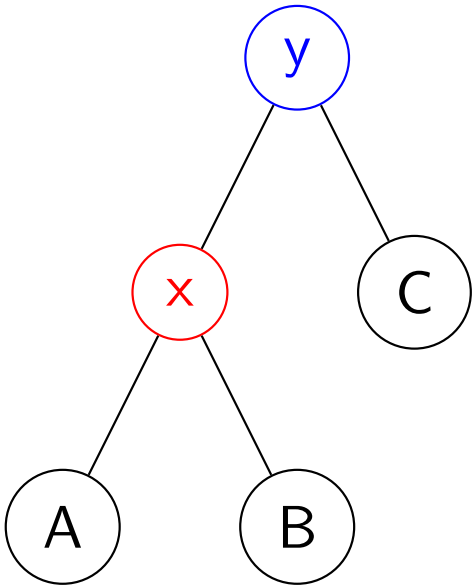
Implementation

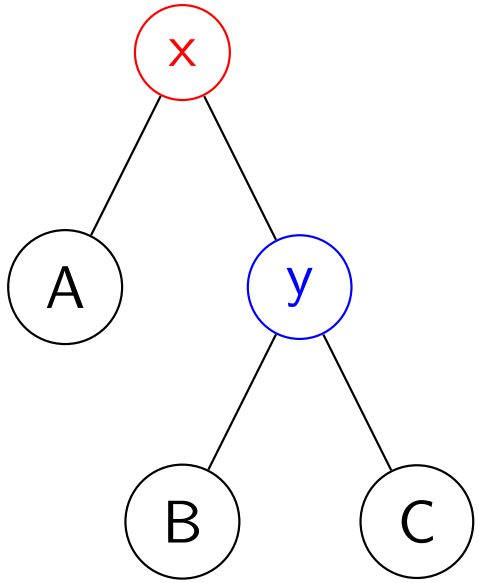
Splaying Function

- Splaying Prototype: void splaying(node **x**)
  - Is a private method.
- Be R a not trial tree and x a node in R.
  splaying(x) move x to the root.
- Moves node x to the root through three rotations.
  Zig-rotation,ZigZig rotation, ZigZag-rotation

Implementation

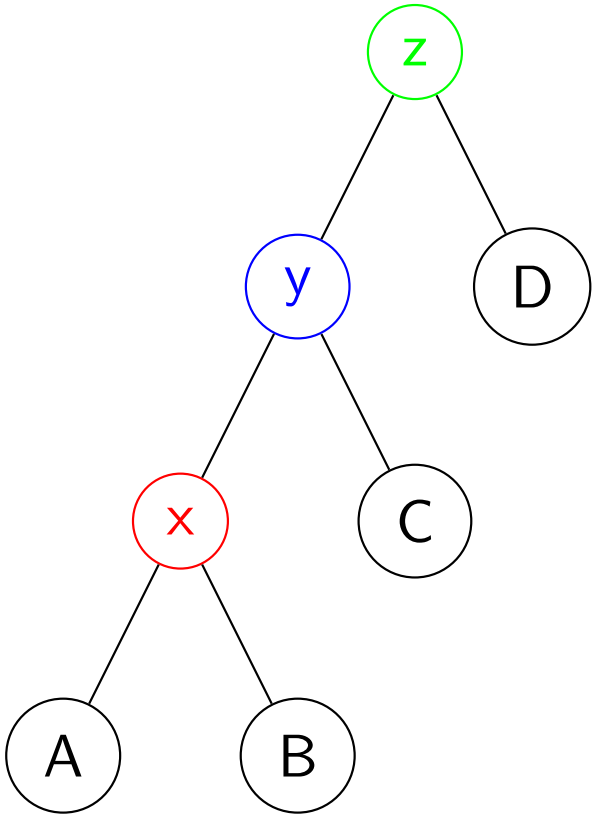
Zig-Rotation

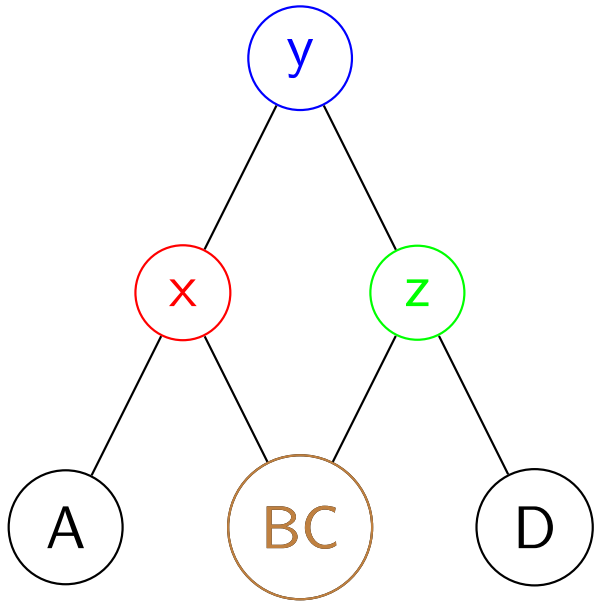




Implementation

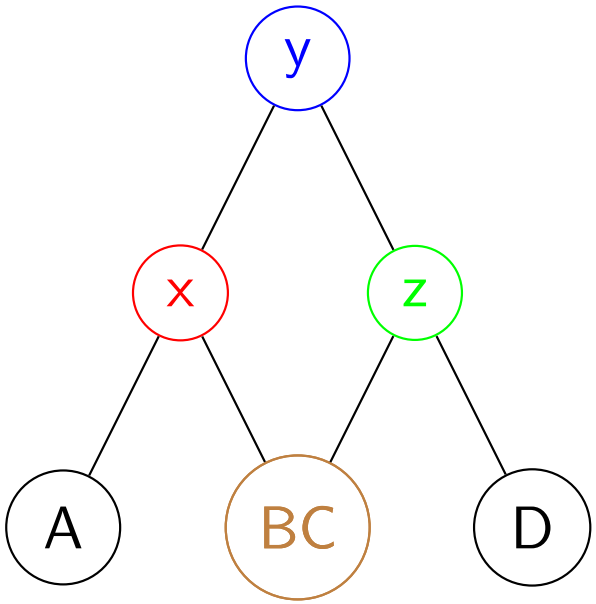
ZigZig-Rotation

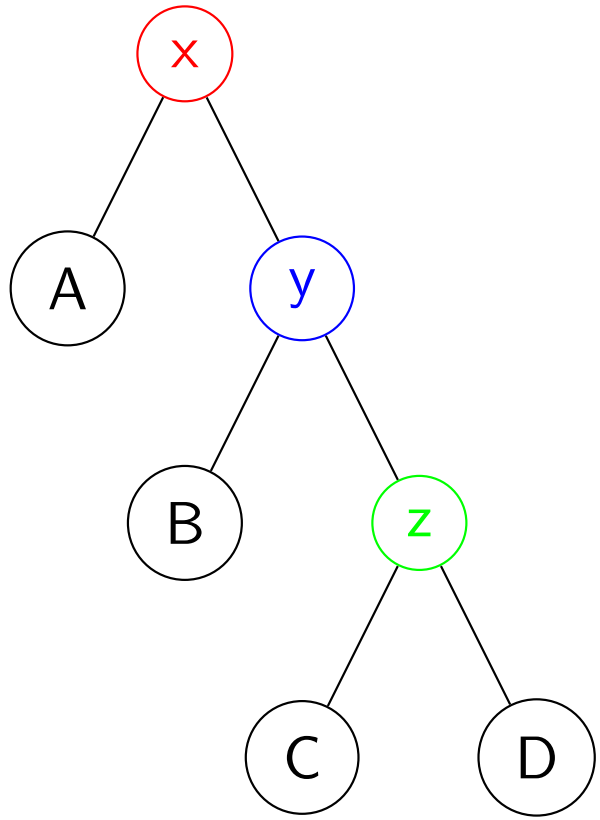




Implementation

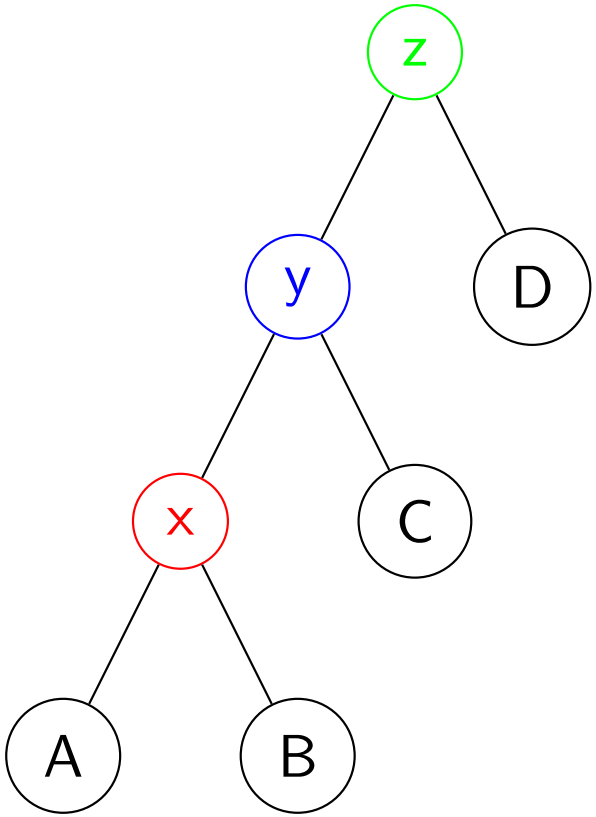
ZigZig-Rotation

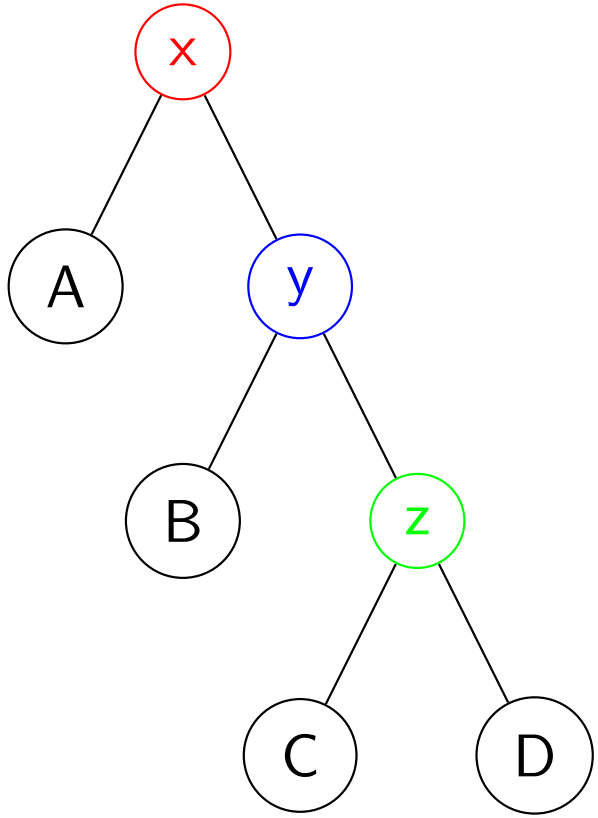




Implementation

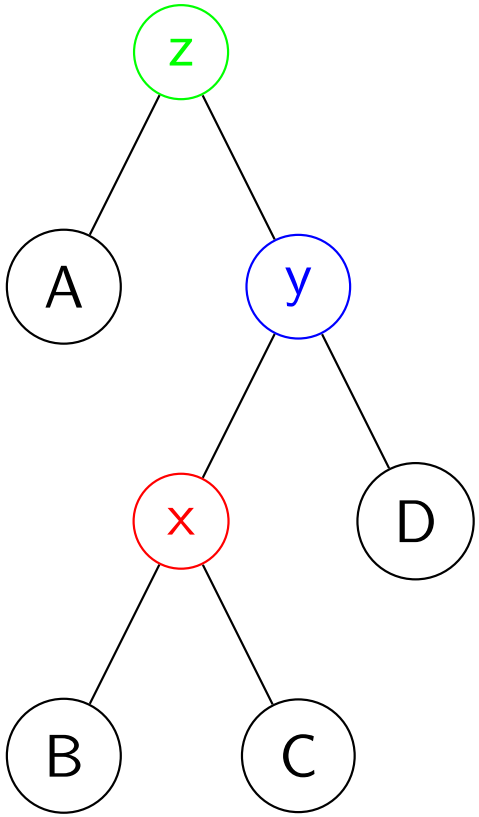
ZigZig-Rotation

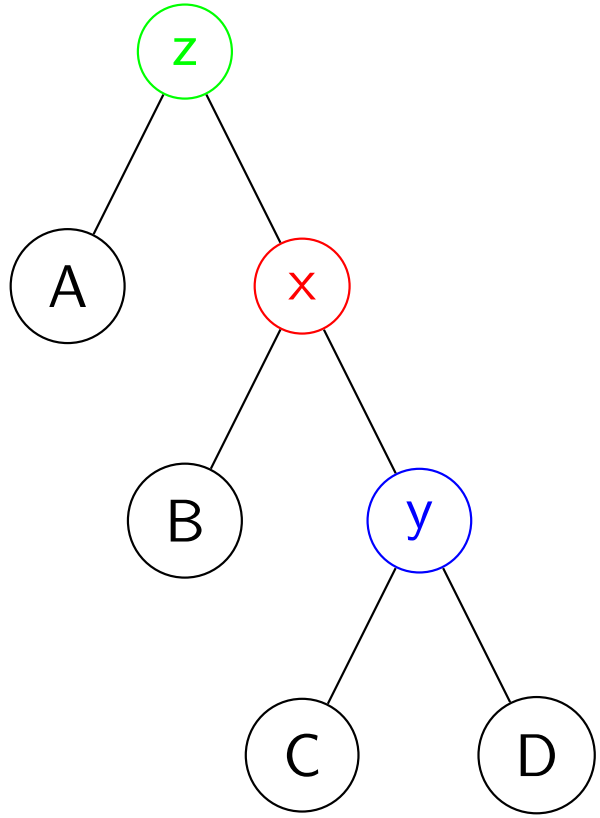




Implementation

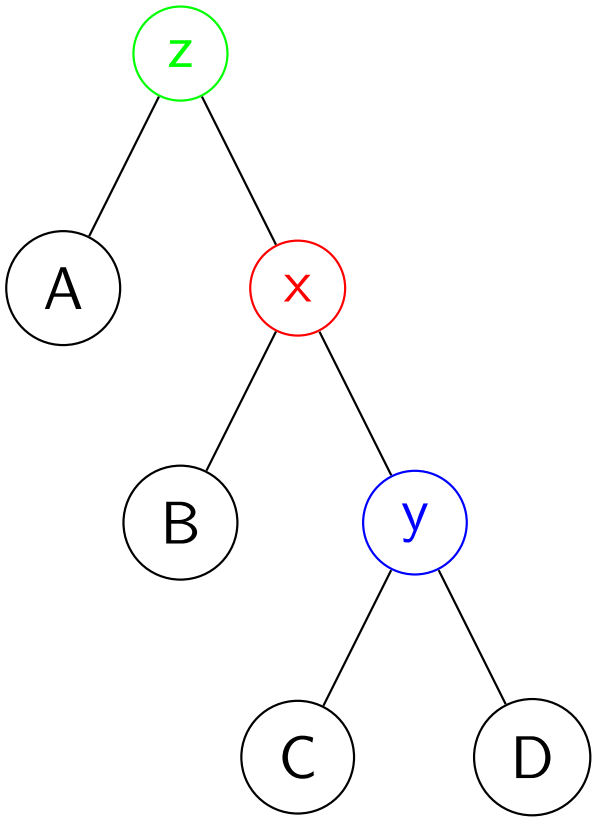
ZigZag-Rotation

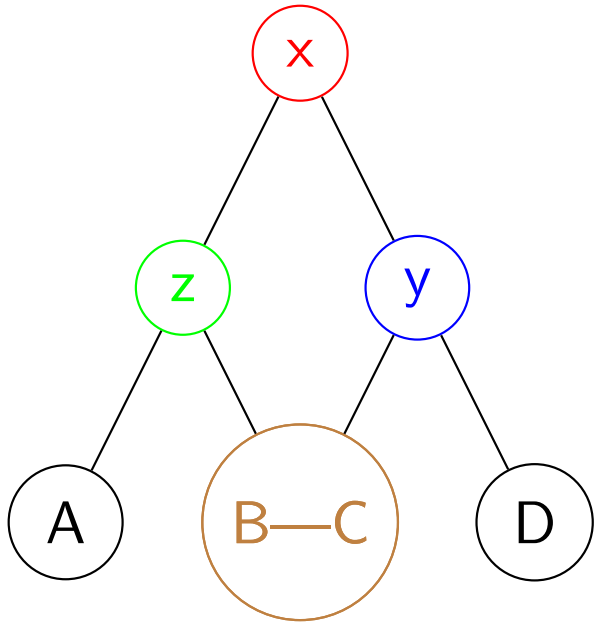




Implementation

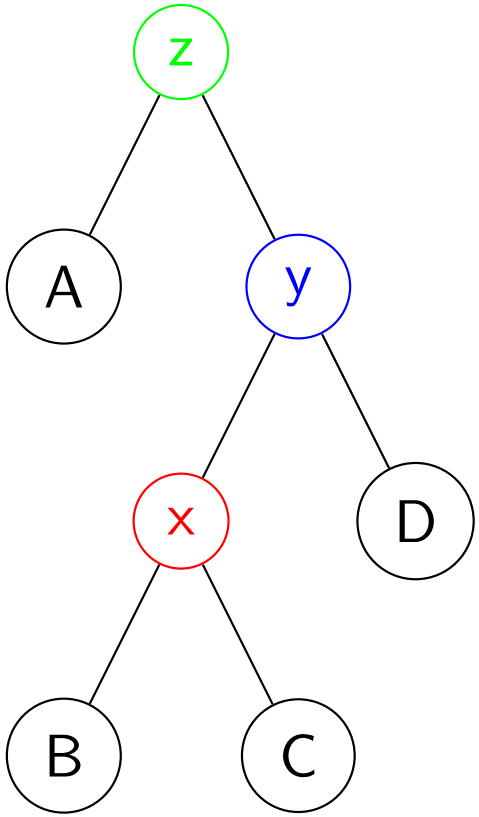
ZigZag-Rotation

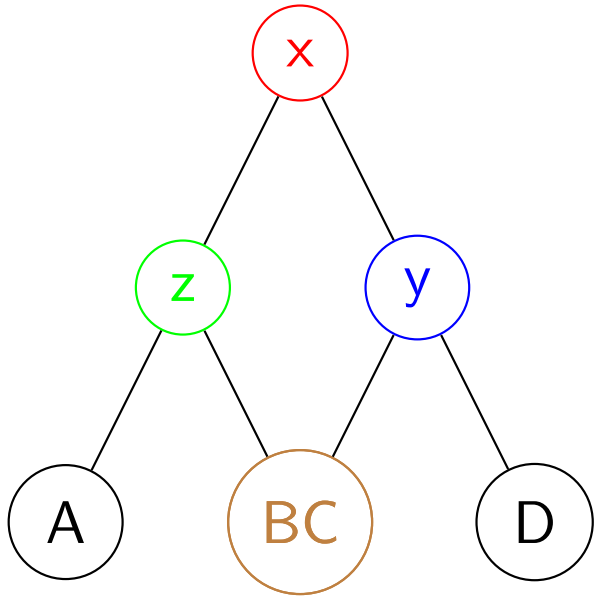




Implementation

ZigZag-Rotation

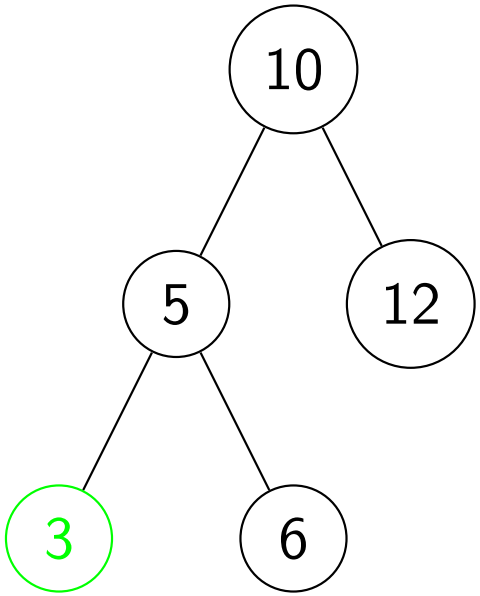


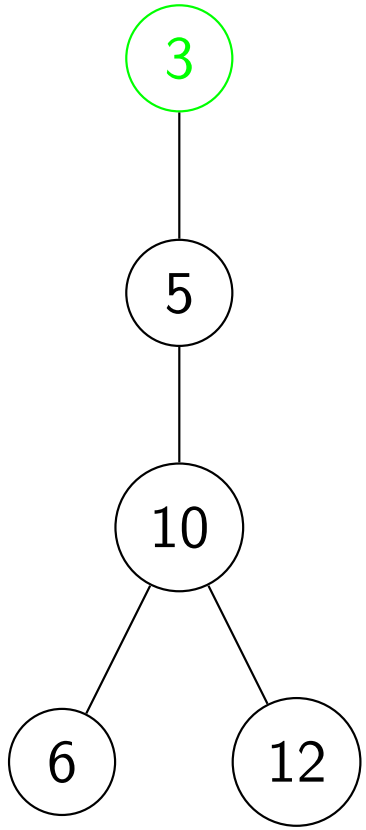


Implementation

Find Method

- Prototype: **bool SplayT::find(dataType key)**

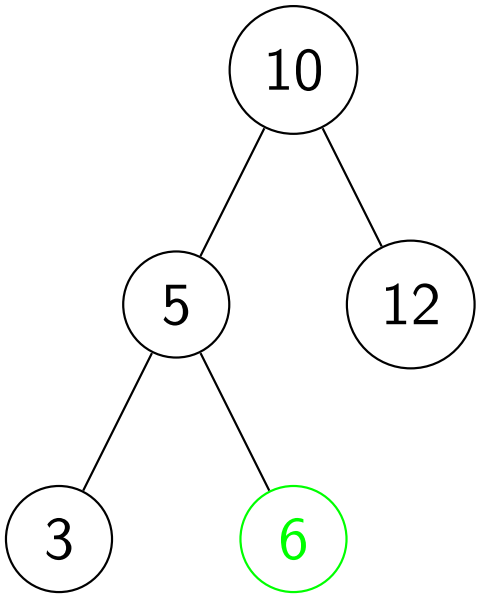


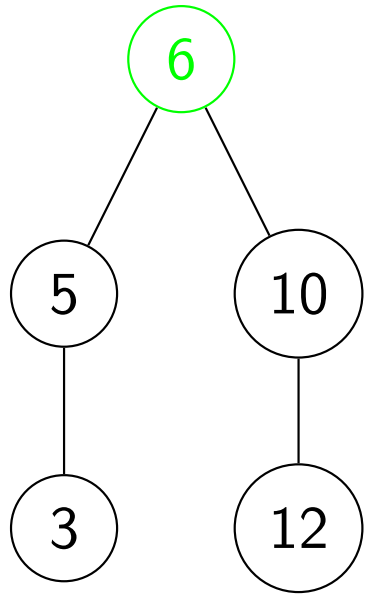


Implementation

Insert Method

- Prototype: **void SplayT::insert(dataType key)**

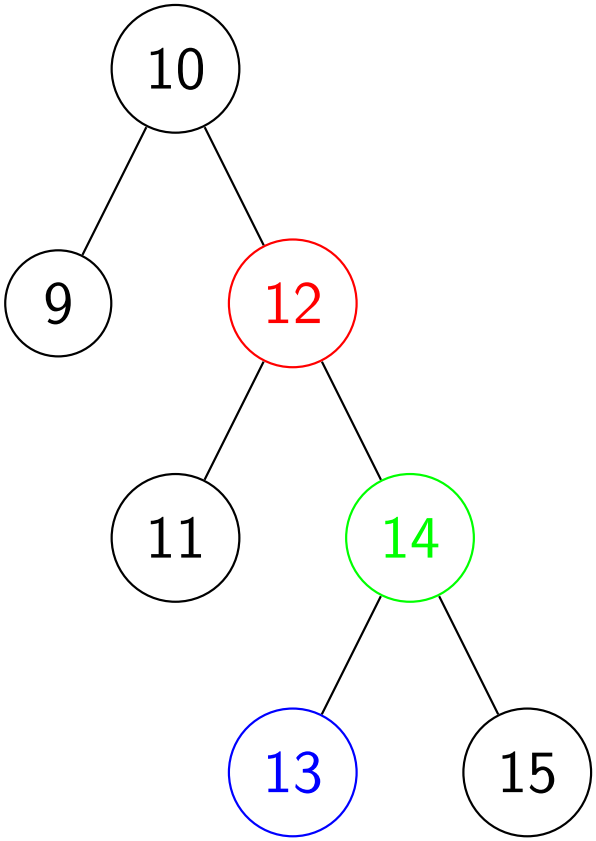


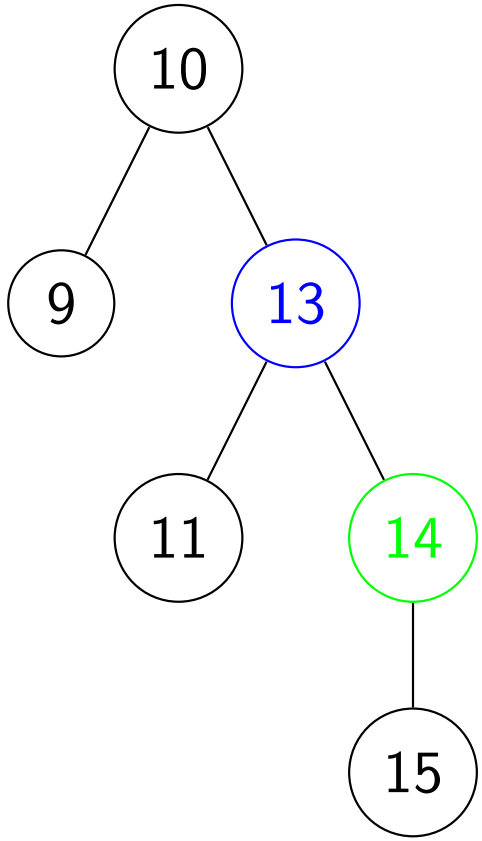


Implementation

Remove Method

- Prototype: **void SplayT::remove(dataType)**

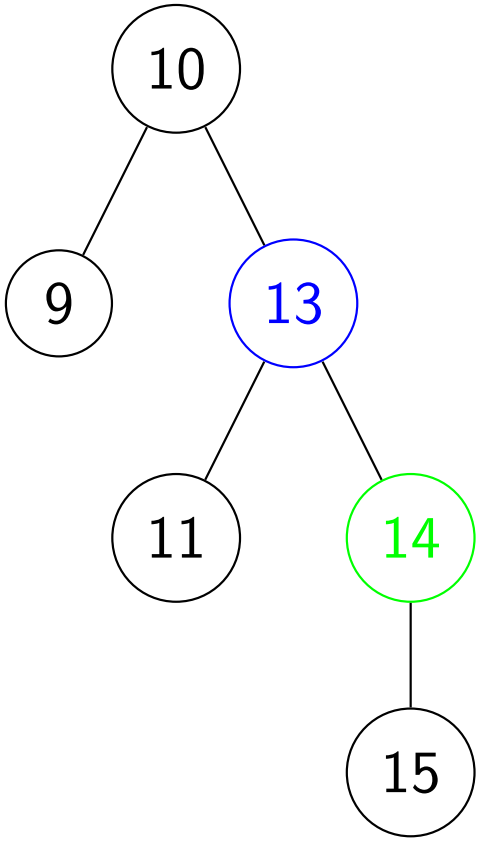


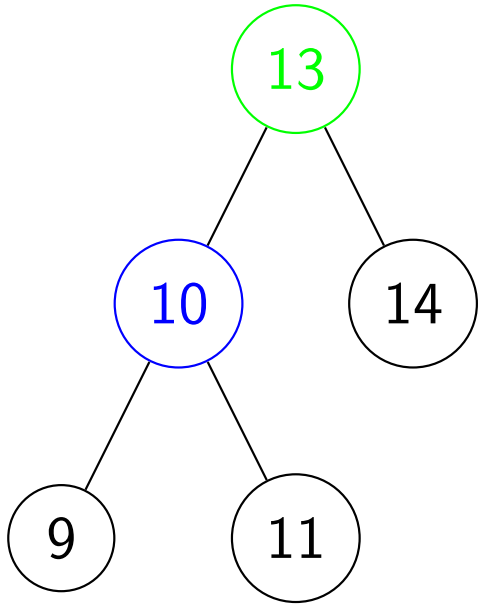


Implementation

Remove Method

- Prototype: **void SplayT::remove(dataType)**





Time Complexity

Splay Tree vs. BST

- Splay Tree Complexity

Time Complexity in big O notation		
Algorithm	**Average**	**Worst case**
Space	O(N)	O(N)
Find	O(log N)	Amortized O(log N)
Delete	O(log N)	Amortized O(log N)
Insert	O(log N)	Amortized O(log N)

Figure: Caption

- BST Complexity

Time Complexity in big O notation		
Algorithm	**Average**	**Worst case**
Space	O(N)	O(N)
Find	O(log N)	O(N)
Delete	O(log N)	O(N)
Insert	O(log N)	O(log N)

Figure: Caption

Conclusions

- The advantages of splay tree with respect to BST.
- The time complexity of the Splay Tree is more efficient compared to BST and other data tree structures.