Faculty of computer science

Data Science and Business Analytics

Moscow
2023

Software project

Data Structure Visualization

Визуализация структур данных

Author: Spirina Mayya Alexandrovna Bpad211
Project supervisor: associate professor, Trushin Dmitrii Vitalievich

Relevance

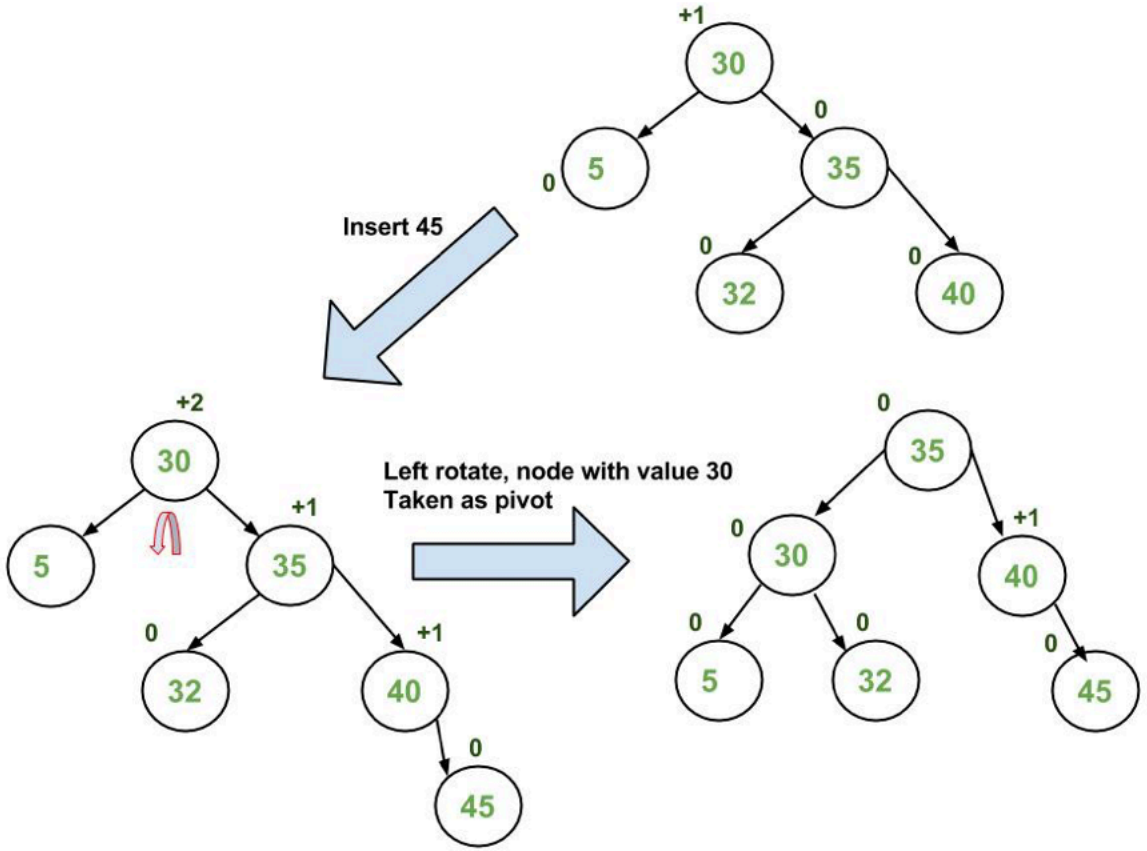
Data Structure Visualization = Better Understanding

AVL Tree

AVL Tree — a self-balancing binary search tree.

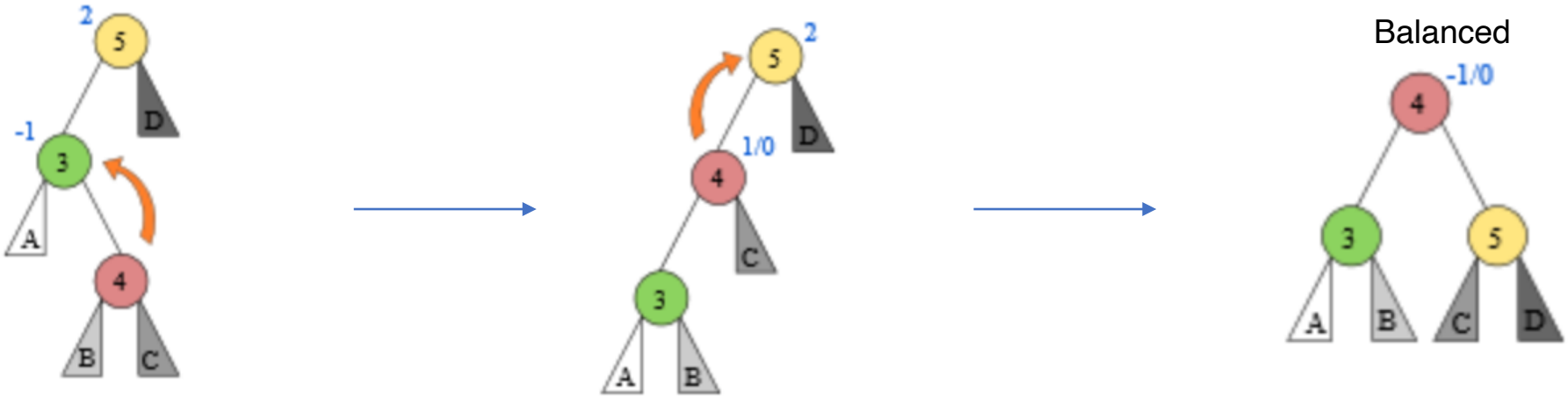
Invariant = Balancing factor =
= Height(left subtree) - Height(right subtree) = {-1 0, 1}

Operations O(log n):
- insertion
- deletion
- search



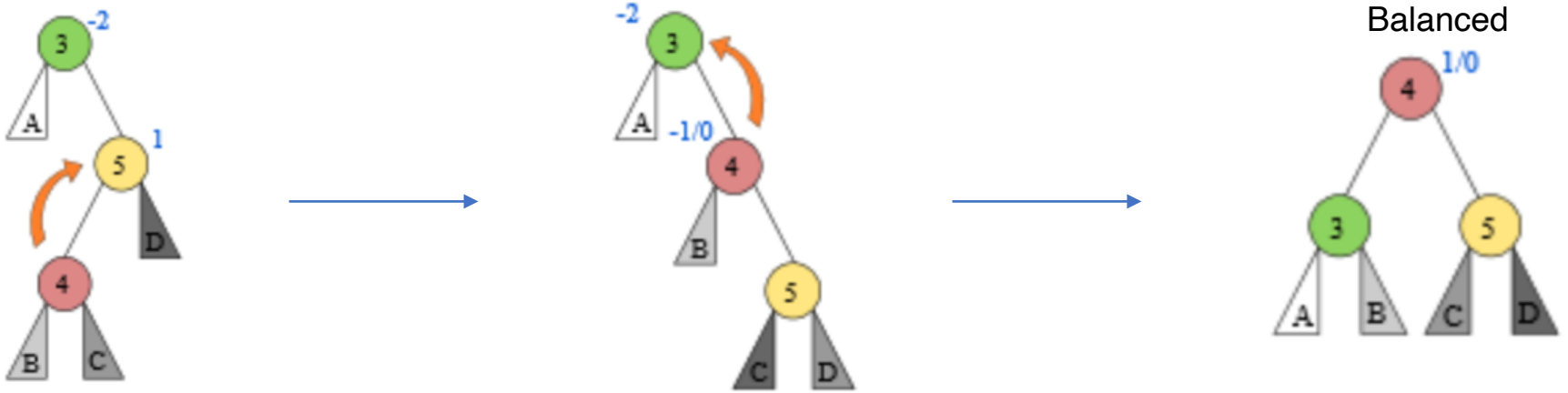
Rotation operations

Left Right Case



Left Left Case

Right Left Case



Right Right Case

Goal and tasks

Goal: Learn how to create **interactive graphical** applications in c++

Initial tasks

- Implement AVL-tree
- Examine Observer pattern
- Examine MVC pattern
- Examine Qt framework
- Implement an interactive Application:
    I. Visualize AVL tree
    II. Visualize all operations
    III. Animate intermediate states of the tree
- Write supporting documentation

Additional tasks fulfilled

- Implement traversal visualization
- Add step back option
- Load from file

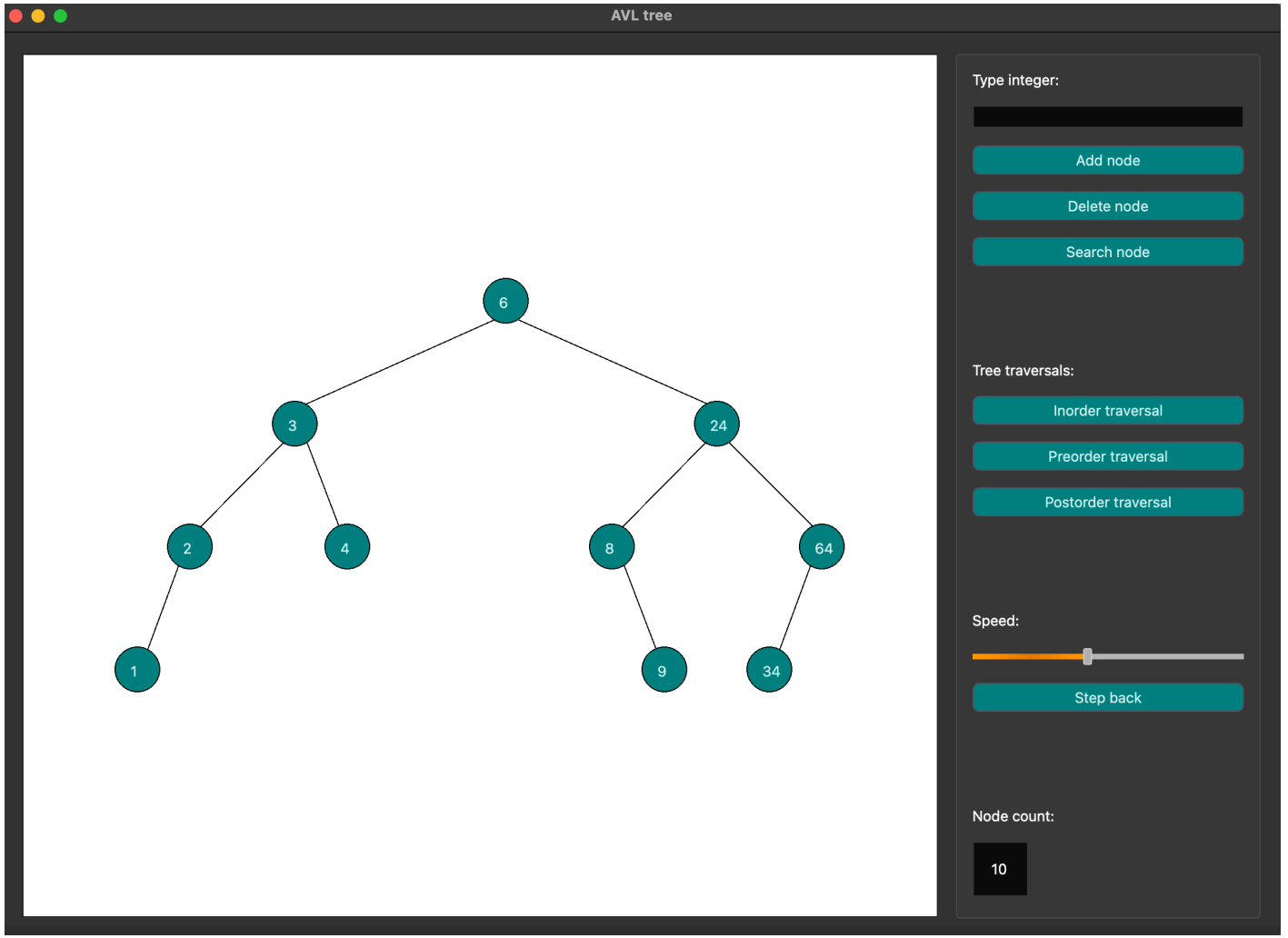
All tasks were fulfilled!

Comparative analysis of analogues

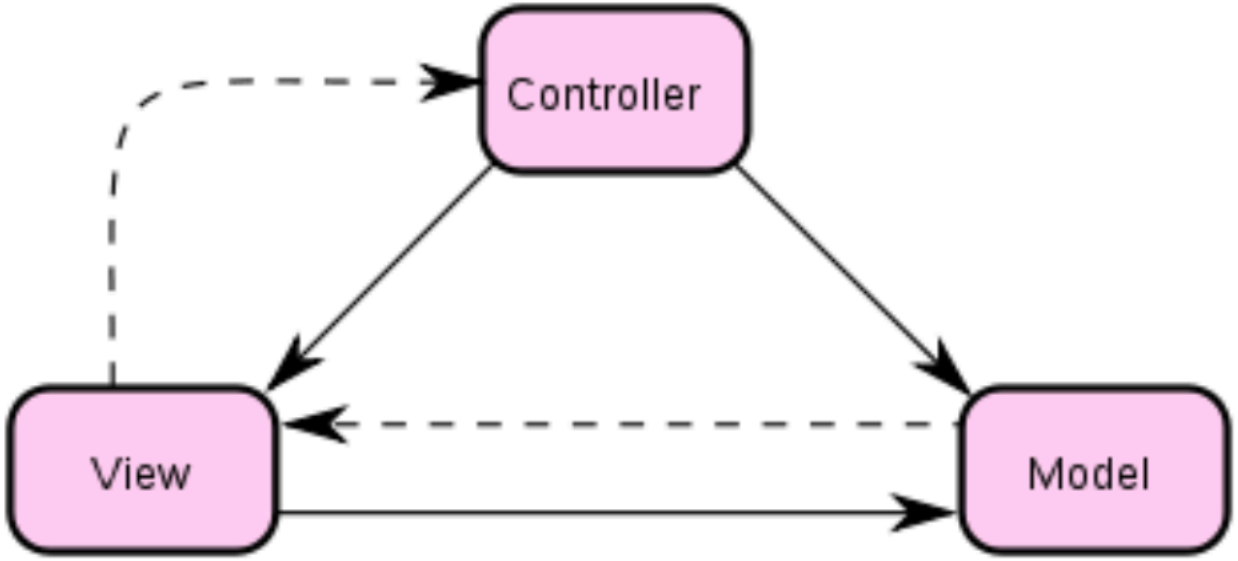
Function	AVL Tree application	usfca.edu	visualgo.net
Wide range of data structures and algorithms	-	+	+
Interaction with nodes by click	+	-	-
Load from file	+	-	-
Pause visualisation	-	+	+
Step-by-step visualisation	+	+	+

Application features

1. Supports operations
   - insert
   - search
   - delete
2. Key selection by node clicking
3. Animation of:
   - inorder traversal
   - preorder traversal
   - postorder traversal
4. Change animation speed
5. Cancel previous operation:
   - insertion
   - deletion
6. Shows statistics (number of nodes)
7. Load tree from a file
8. Error notifications



Methods: Model-View-Controller



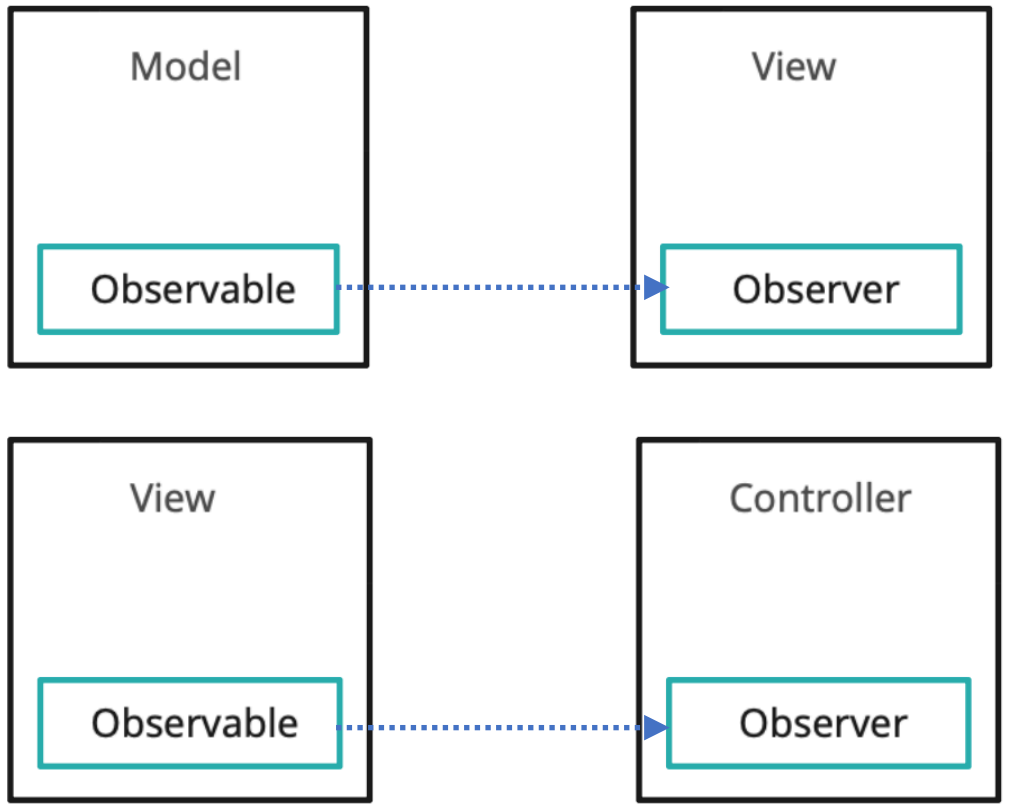
Methods: Observer pattern

The role: safe connection between objects

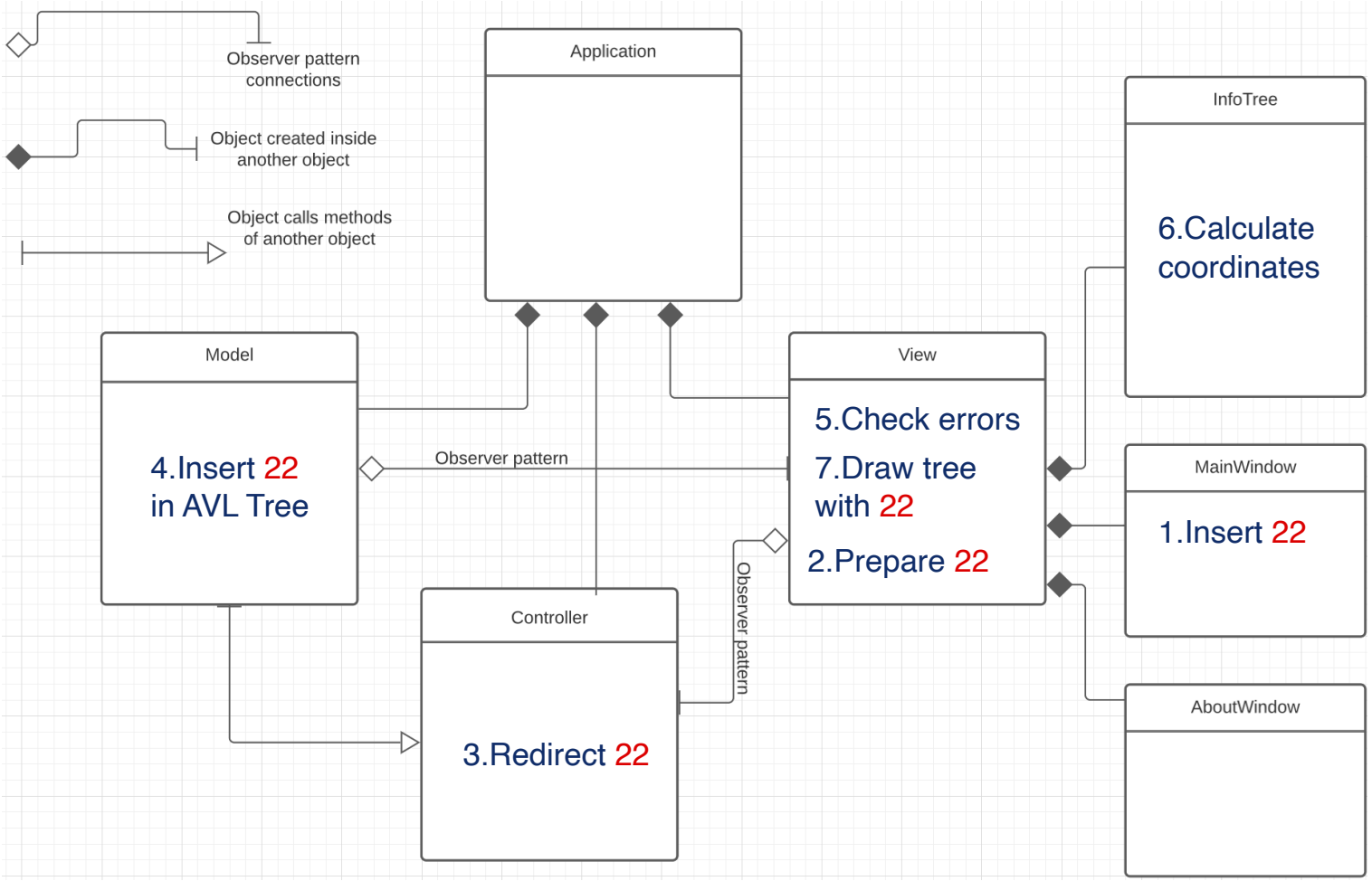
Actors:
- Observable — sends information
- Observer — receives information

Do not use interfaces.
Observable and Observer = Class Data

Observable notifies Observer



Application architecture



Implementation details

1) Tree visualization = Store Additional Tree

Additional Tree nodes = Tree nodes + Extra information

Extra information:
- coordinates (x, y)
- width of subtree

2) Choosing node by clicking
Implemented as search in Additional Tree by coordinates

3) Reading from a file
Insert keys one-by-one

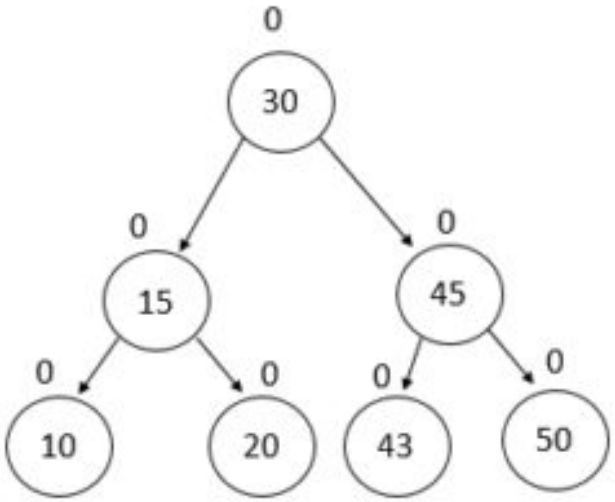
4) Step back
Store previous operation info.

Results

Implemented

1) An interactive GUI Application
2) Visualization of AVL tree using MVC pattern
3) MVC pattern using observer pattern
4) GUI using Qt.
5) Animation of operations
6*) Reading trees from file
7*) Step back operation
8*) Choosing nodes by clicking
9*) Measured time of node insertion

2211 lines of code



- https://github.com/spirinamayya/avltree/tree/master