Lab 8.4.1 Operators: complex classes

Objectives

Familiarize the student with:

- writing operator code to class hierarchy;
- writing iostream operator;

Scenario

Prepare a base container class for a binary tree structure. The binary tree is a tree data structure where each node has zero, one or two child nodes. These child nodes are referred to as the left child and the right child. This class consists of three fields:

- a field with a value;
- a pointer to the left child;
- a pointer to the right child.

Implement a method to add a value to a tree, test it with some values (you can hard-code the values – testing should be easier and faster). Implement three derived classes – each one of these classes must have two operators overloaded: << and []. The first operator (<<) should print the tree in the adequate traversal method. The second operator ([]) should return an element of the tree in the appropriate order, i.e. if a tree has three elements, then tree[1] should return a root element in the inorder class: The three classes are:

- inorder;
- postorder;
- preorder.

A simple description of each of the tree traversal methods:

- inorder is a tree traversal method where you first traverse the left child inorder, then print the value of the current node, and then traverse the right child inorder;
- postorder is a tree traversal method where you first traverse the left child postorder, and then traverse the right child postorder, and then print the value of the current node;
- preorder is a tree traversal method where you first print the value of the current node, then traverse the left child preorder, and then traverse the right child preorder.

Overload the operator in each of the derived classes (preorder, inorder, postorder). Test all the classes with different values.

Example input

```
3 5 6 1
```

Example output

```
Inorder: 1 3 5 6
Postrder: 1 6 5 3
Preorder: 3 1 5 6
```