

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ФАКУЛЬТЕТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК ИМЕНИ
ПРОФЕССОРА Н.И.ЧЕРВЯКОВА

ЛАБОРАТОРНАЯ РАБОТА №17

Алгоритмизация и программирование

Вариант 9

Выполнил студент:

Сивко Иван Андреевич

студент 2 курса

группа ПМИ-б-о-23-2,

направление подготовки 01.03.02

Проверил:

Ассистент кафедры вычислительной
математики и кибернетики, к.ф.-м.н.,

Черкашина Анастасия Андреевна

Вариант 9

Цель:

- Совершенствование навыков разработки программ в среде программирования
- Совершенствование навыков в программировании с использованием указателей
- Исследование процесса формирования элементов простейшей динамической структуры данных – двоичное дерево
- Исследование операций при работе с двоичными деревьями

Задание 1

В соответствии с вариантом написать и отладить программу, используя динамическую структуру данных: Двоичное дерево.

1 Условие

Разработать программу, которая строит двоичное дерево Т. Описать процедуру, заменяющую в дереве Т все отрицательные элементы на их абсолютные значения.

2 Алгоритм / Математическая модель

Программа предназначена для работы с бинарным деревом поиска (Binary Search Tree, BST) и выполняет операции добавления, отображения и модификации данных. Алгоритм выполнения программы следующий:

1. Ввод чисел:

- Пользователь вводит целые числа построчно.
- Ввод завершается пустой строкой.
- Числа добавляются в бинарное дерево поиска.

2. Печать дерева:

- Дерево выводится на экран в упорядоченном (in-order) порядке.
- Если дерево пустое, вывод на экран не выполняется.

3. Замена отрицательных значений:

- В каждом узле дерева отрицательные значения заменяются их модулями.

4. Повторная печать дерева:

- После преобразования дерево снова выводится в упорядоченном виде.

5. Выход: Программа завершает свою работу.

2.1 Математическая модель

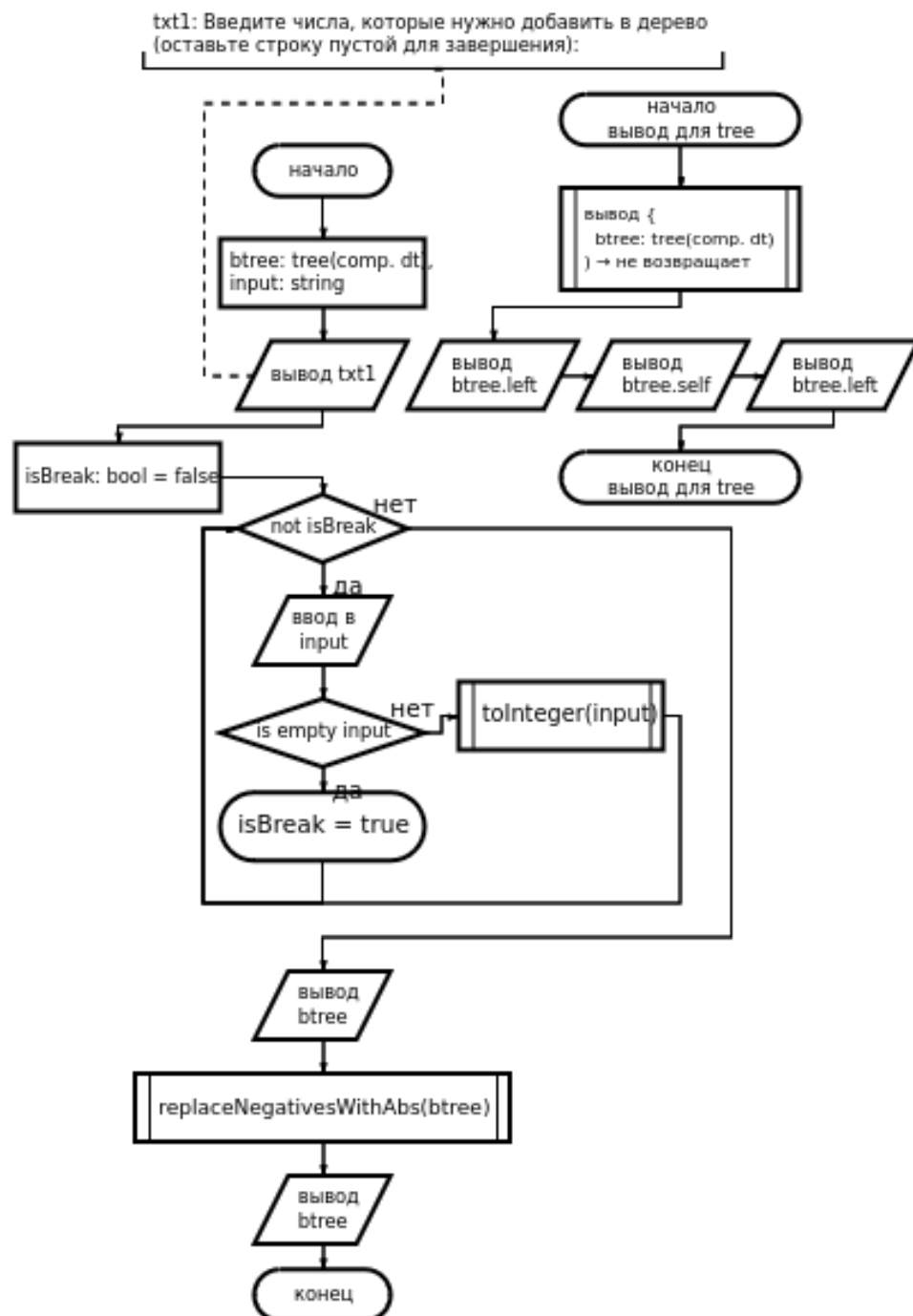
Бинарное дерево поиска (BST): Дерево представлено как тройка:

$$T = (N, L, R)$$

где N — значение узла, L — левое поддерево, R — правое поддерево. Для любого узла выполняется:

$$\forall x \in L, x < N; \quad \forall y \in R, y \geq N.$$

3 Диаграмма:



4 Код:

```
#include <cstdlib>
#include <cstdlib>
#include <initializer_list>
#include <string>
#include <iostream>

template <typename T>
struct Node {
    T data;
    Node<T> *left;
    Node<T> *right;

    Node(T data, Node<T> *left=nullptr, Node<T> *right=nullptr)
        : data{ data }, left{ left }, right{ right }
    { }
};

template <typename T>
class BinaryTree {
    Node<T> *root;

    static void insert(Node<T> *&node, T data) {
        if (!node) {
            node = new Node{ data };
            return;
        }
        if (data < node->data)
            insert(node->left, data);
        else
            insert(node->right, data);
    }

    static void display(std::ostream& out, const Node<T> *node) {
        if (!node)
            return;
        display(out, node->left);
        out << node->data << ' ';
        display(out, node->right);
    }

    static void recReplaceWithOposite(Node<T> *node) {
        if (!node)
            return ;
        recReplaceWithOposite(node->left);
        if (node->data < 0)
            node->data *= -1;
        recReplaceWithOposite(node->right);
    }
public:
```

```

BinaryTree()
    : root{ nullptr }
{ }

BinaryTree(std::initializer_list<T> values)
    : root{ nullptr }
{
    for (T value : values)
        insert(value);
}

void insert(int32_t data) {
    insert(root, data);
}

// заменяет все отрицательные элементы их абсолютными значениями
void replaceNegativesWithAbs() {
    Node<T> *firstNegative{ root };
    while (firstNegative && firstNegative->data >= 0)
        firstNegative = firstNegative->left;
    if (!firstNegative)
        return ;
    recReplaceWithOposite(firstNegative);
}

friend std::ostream& operator<<(std::ostream& out, const BinaryTree& btree) {
    BinaryTree::display(out, btree.root);
    return out;
}

};

int main() {
    BinaryTree<int32_t> btree{};
    std::string input{};
    std::cout << "Enter the numbers to add to the tree (leave the line blank to com
while (true) {
    std::getline(std::cin, input);
    if (input.empty())
        break;
    try {
        int32_t value = std::stoi(input);
        btree.insert(value);
    } catch (const std::invalid_argument& e) {
        std::cout << "Please enter a valid integer.\n";
    } catch (const std::out_of_range& e) {
        std::cout << "The number is out of range.\n";
    }
}

std::cout << btree << '\n';
btree.replaceNegativesWithAbs();

```

```
    std::cout << btree << '\n';  
    return EXIT_SUCCESS;  
}
```

source code

5 Результат работы программы:

```
[john@arch cpp]$ ./a.out  
Enter the numbers to add to the tree (leave the line blank to complete):  
-8  
2  
0  
-5  
-4  
-7  
9  
3  
-1  
  
-8 -7 -5 -4 -1 0 2 3 9  
8 7 5 4 1 0 2 3 9
```