МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Отчет к лабораторной работе № 5

Тема: **Объектно-ориентированный подход. Шаблоны классов**

Дисциплина: **«Информационные технологии и программирование»**

**Выполнил:**

Студент 1 курса группы ПИЖ-б-о-23-1(2) направление подготовки: 09.03.04 «Программная инженерия»

Лаб прог 5

**Проверила:**

Новикова Е. Н.

Ставрополь 2024

**Вариант 8**

**Задание 1:**

#include <iostream>

#include <stdexcept>

template <typename T>

class BinaryTree {

private:

struct Node {

T data;

Node\* left;

Node\* right;

Node(const T& data) : data(data), left(nullptr), right(nullptr) {}

};

Node\* root;

void insert(Node\*& node, const T& data) {

if (node == nullptr) {

node = new Node(data);

}

else if (data < node->data) {

insert(node->left, data);

}

else {

insert(node->right, data);

}

}

void inorder(Node\* node) const {

if (node != nullptr) {

inorder(node->left);

std::cout << node->data << " ";

inorder(node->right);

}

}

void destroy(Node\* node) {

if (node != nullptr) {

destroy(node->left);

destroy(node->right);

delete node;

}

}

public:

BinaryTree() : root(nullptr) {}

~BinaryTree() {

destroy(root);

}

void insert(const T& data) {

insert(root, data);

}

void inorder() const {

inorder(root);

std::cout << std::endl;

}

};

int main() {

try {

BinaryTree<int> tree;

tree.insert(10);

tree.insert(5);

tree.insert(15);

tree.insert(3);

tree.insert(7);

std::cout << "Inorder traversal: ";

tree.inorder();

}

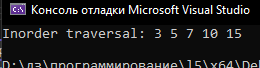
catch (const std::exception& e) {

std::cerr << "Exception: " << e.what() << std::endl;

}

return 0;

}

****

**Задание 2:** Создать шаблон класса «бинарное дерево». Использовать его для сортировки целых чисел и строк, задаваемых с клавиатуры или из файла.

#include <iostream>

#include <string>

template <typename T>

class BinaryTree {

private:

struct Node {

T data;

Node\* left;

Node\* right;

Node(T value) : data(value), left(nullptr), right(nullptr) {}

};

Node\* root;

void insert(Node\*& node, T value) {

if (!node) {

node = new Node(value);

}

else if (value < node->data) {

insert(node->left, value);

}

else {

insert(node->right, value);

}

}

void inorder(Node\* node) const {

if (node) {

inorder(node->left);

std::cout << node->data << " ";

inorder(node->right);

}

}

void clear(Node\* node) {

if (node) {

clear(node->left);

clear(node->right);

delete node;

}

}

public:

BinaryTree() : root(nullptr) {}

~BinaryTree() {

clear(root);

}

void insert(T value) {

insert(root, value);

}

void inorder() const {

inorder(root);

std::cout << std::endl;

}

};

int main()

{

setlocale(0, "rus");

BinaryTree<int> intTree;

BinaryTree<std::string> stringTree;

int choice;

do {

std::cout << "1. Добавить целое число\n";

std::cout << "2. Добавить строку\n";

std::cout << "3. Показать отсортированные целые числа\n";

std::cout << "4. Показать отсортированные строки\n";

std::cout << "5. Выход\n";

std::cout << "Ваш выбор: ";

std::cin >> choice;

if (choice == 1) {

int value;

std::cout << "Введите целое число: ";

std::cin >> value;

intTree.insert(value);

}

else if (choice == 2) {

std::string value;

std::cout << "Введите строку: ";

std::cin >> value;

stringTree.insert(value);

}

else if (choice == 3) {

std::cout << "Отсортированные целые числа: ";

intTree.inorder();

}

else if (choice == 4) {

std::cout << "Отсортированные строки: ";

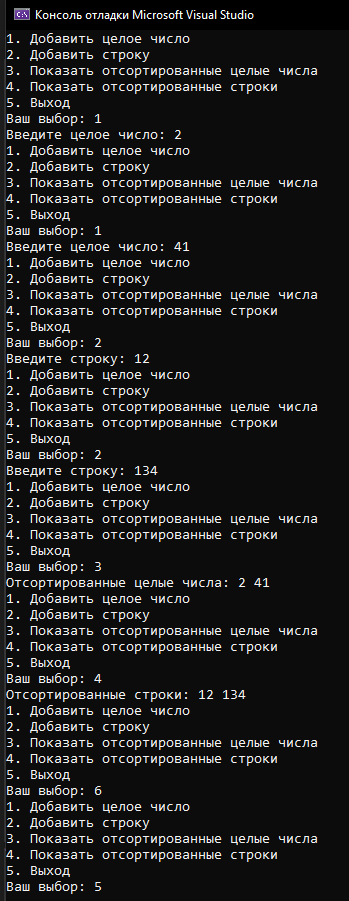
stringTree.inorder();

}

} while (choice != 5);

return 0;

}



**https://github.com/strit595/Progrm**