Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Курский государственный университет»

Кафедра программного обеспечения и администрирования информационных систем

Направление подготовки математическое обеспечение и администрирование информационных систем

Форма обучения очная

**Отчет**

**по лабораторной работе №4**

«Шаблоны классов»

Выполнил:

студент группы 213.1 Козявин М. С.

Проверил:

старший преподаватель кафедры ПОиАИС Ураева Е. Е.

Курск, 2022

***Цель работы:*** Изучить особенности написания программ на языке С++

с использованием шаблонов классов.

***Задание***

*Задача 1.* Создать шаблонный класс Queue для работы с очередью элементов любого типа. В качестве членов-данных рекомендуется брать два элемента (определяющие начало и конец очереди) самоссылочного класса Node (должен быть другом основному классу) следующего вида:

class Node { Type data; Node \*next; };

Класс должен содержать конструктор по умолчанию, основной конструктор и конструктор копирования. Определить в этом классе функции-члены класса, обеспечивающие: добавление элементов в очередь, удаление элемента из очереди, распечатку элементов очереди. Дополнительно перегрузить операторную функцию для операции ! (логическое отрицание), которая возвращает минимальный элемент очереди.

***Разработка алгоритма***

*Задача 1*

Выходные данные: *элементы списка и минимальный из них*.

UML диаграмма классов задачи представлен на рисунке 1.

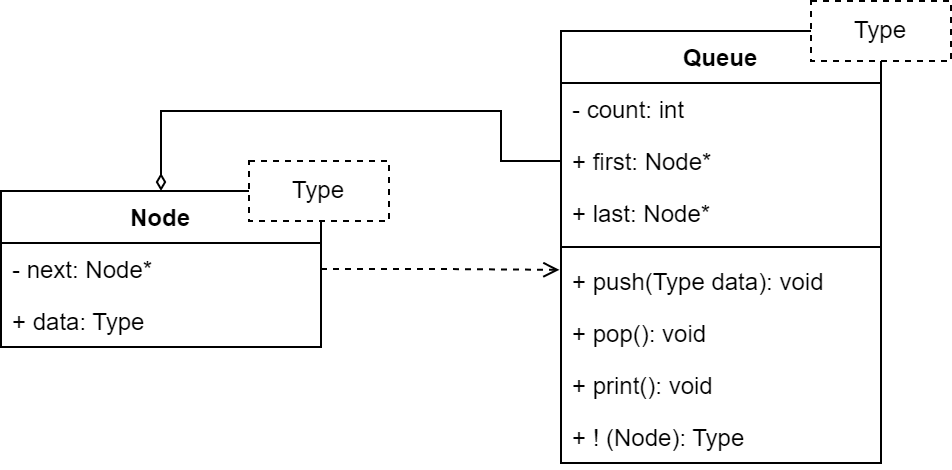


Рисунок 1 - UML диаграмма классов задачи 1

Алгоритм решения задачи представлен на рисунке 2

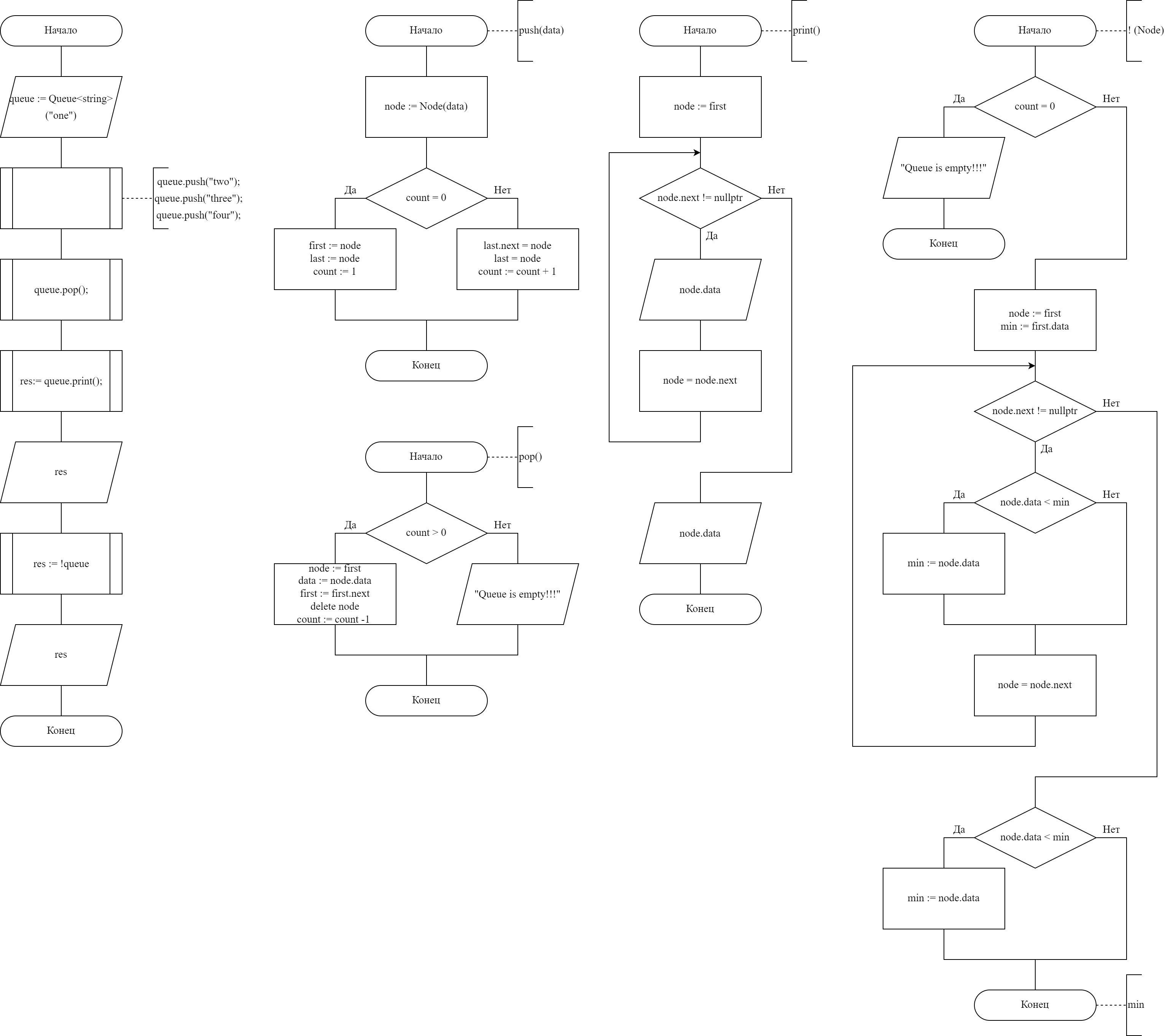


Рисунок 2 - Алгоритм решения задачи 1

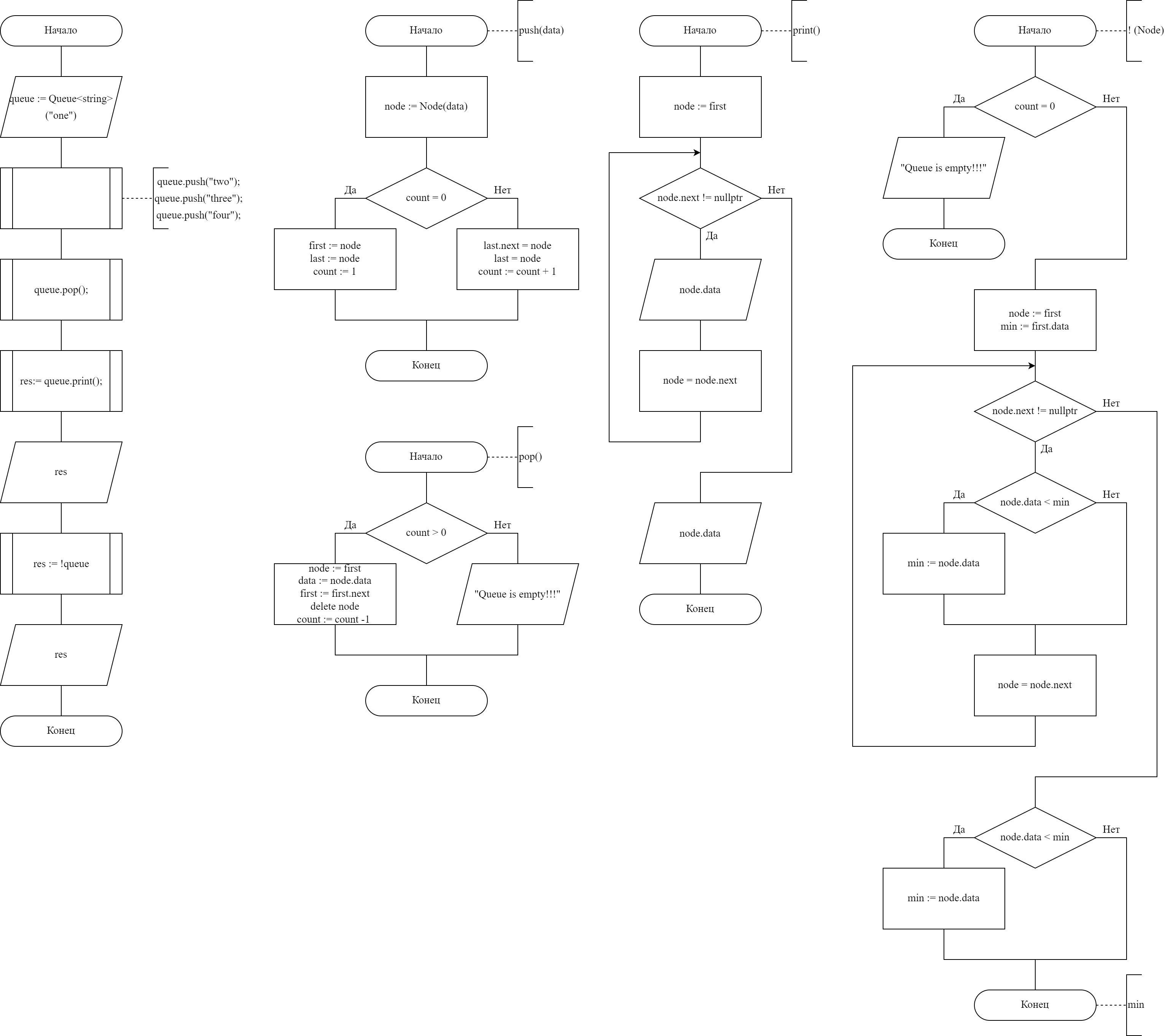


Рисунок 2 - Продолжение

***Текст программы***

*Текст программы для решения задачи 1*

*#include <iostream>*

*using namespace std;*

*template <typename Type> class Node;*

*template <typename Type> class Queue;*

*template <typename Type>*

*class Node {*

*friend class Queue<Type>;*

*Node<Type>\* next;*

*public:*

*Type data;*

*Node() {*

*next = nullptr;*

*};*

*Node(Type data): Node() {*

*this->data = data;*

*}*

*};*

*template <typename Type>*

*class Queue {*

*int count;*

*public:*

*Node<Type>\* first;*

*Node<Type>\* last;*

*Queue() {*

*count = 0;*

*};*

*Queue(Type data) {*

*Node<Type>\* node = new Node<Type>(data);*

*first = node;*

*last = node;*

*count = 1;*

*}*

*void push(Type data) {*

*Node<Type>\* node = new Node<Type>(data);*

*if (count == 0) {*

*first = node;*

*last = node;*

*count = 1;*

*} else {*

*last->next = node;*

*last = node;*

*count++;*

*}*

*}*

*void pop() {*

*if (count > 0) {*

*Node<Type>\* node = first;*

*Type data = node->data;*

*first = first->next;*

*delete node;*

*count--;*

*} else {*

*cout << "Queue is empty!!!";*

*}*

*}*

*void print() {*

*Node<Type>\* node = first;*

*while (node->next != nullptr) {*

*cout << node->data << " ";*

*node = node->next;*

*}*

*cout << node->data << "\n";*

*}*

*Type operator ! () {*

*if (count == 0) {*

*cout << "Queue is empty!!!";*

*return Type();*

*}*

*Node<Type>\* node = first;*

*Type min = first->data;*

*while (node->next != nullptr) {*

*if (node->data < min) {*

*min = node->data;*

*};*

*node = node->next;*

*};*

*if (node->data < min) {*

*min = node->data;*

*};*

*return min;*

*}*

*};*

*int main() {*

*Queue<string> queue = Queue<string>("one");*

*queue.push("two");*

*queue.push("three");*

*queue.push("four");*

*queue.pop();*

*queue.print();*

*cout << !queue << "\n";*

*}*

***Тестирование программы***

Тестирование задачи 1 представлено на рисунке 3.

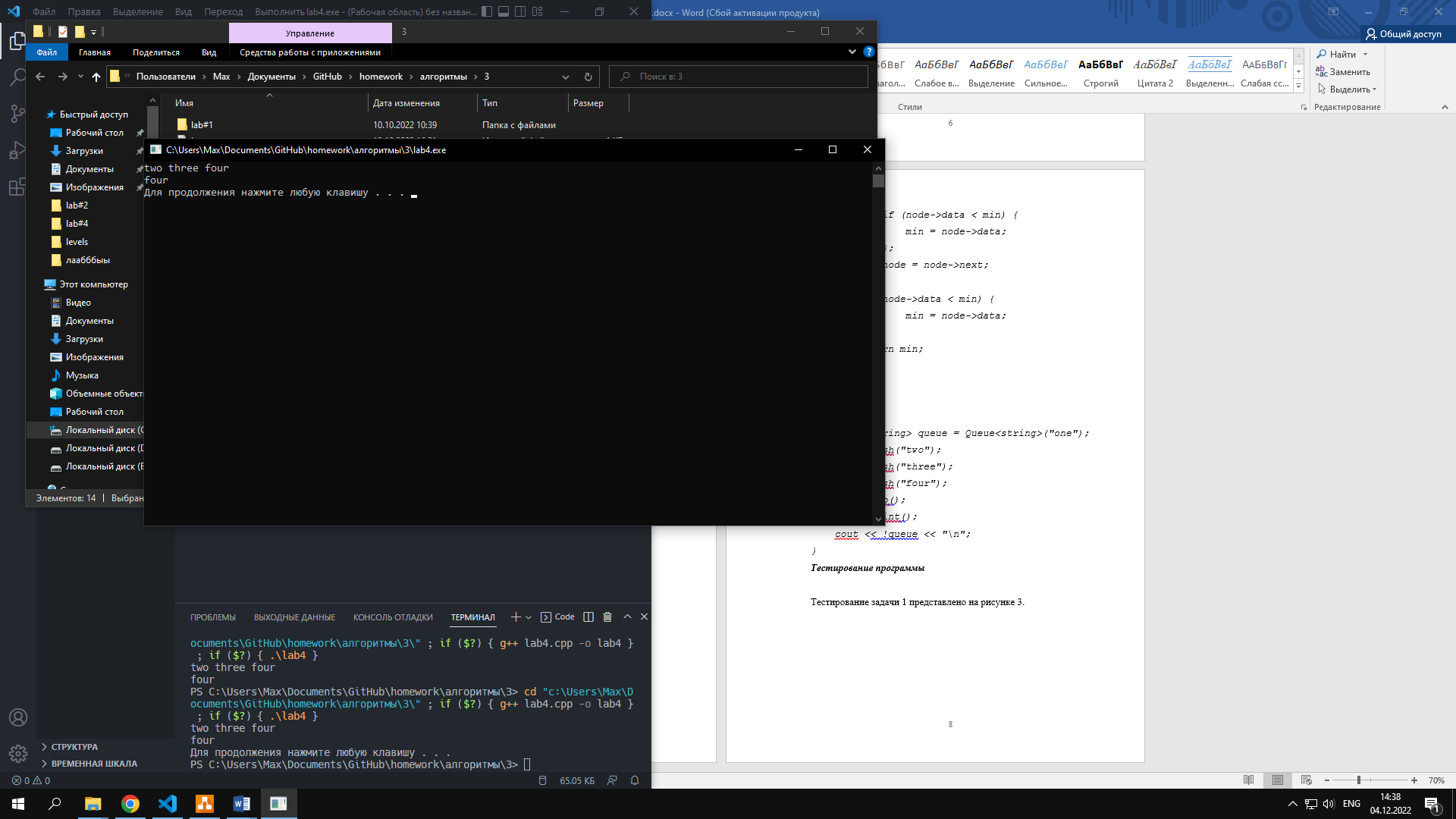


Рисунок 3 - Тест 1 задачи 1