Министерство высшего образования и науки Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

**лабораторная работа №11 - очередь**

Дисциплина: основы алгоритмизации и программирования

Выполнил:

Студент группы РИС-22-1б

Баженов Тимофей

Проверил:

Доцент кафедры ИТАС

Полякова Ольга Андреевна

**Пермь, 2023**

**1. Постановка задачи**

Создать функции для работы с очередью

**2. Анализ задачи**

1. Структура узла (элемента очереди) содержит данные типа char и указатель на следующий узел. Структура очереди содержит размер очереди, указатель на головной и хвостовой узел – head\_node и tail\_node.

2. Функция push добавляет элемент в конец очереди. Новому узлу присваиваются данные и указатель на следующий = nullptr, так как последний элемент никуда не указывает. Если очередь еще не содержит элементов, то новый узел становиться одновременно и головным и хвостовым узлом. Если очередь уже содержала элементы, то после tail\_node добавляется новый узел, который потом становиться tail\_node.

3. Функция full запрашивает у пользователя значения узлов очереди и последовательно циклом for (количество итераций – размер очереди) отправляет значение в функцию push, тем самым введённые значения последовательно добавляются в очередью.

4. Функция print начиная циклом while начиная с головного элемента head\_node выводит значение узлов очереди и перемещает указатель на следующий, пока не достигнет nullptr.

5. Функция pop запоминает первый элемент очереди, переносит указатель head\_node на следующий и удаляет тот элемент, который запоминался. pop удаляет первый элемент очереди.

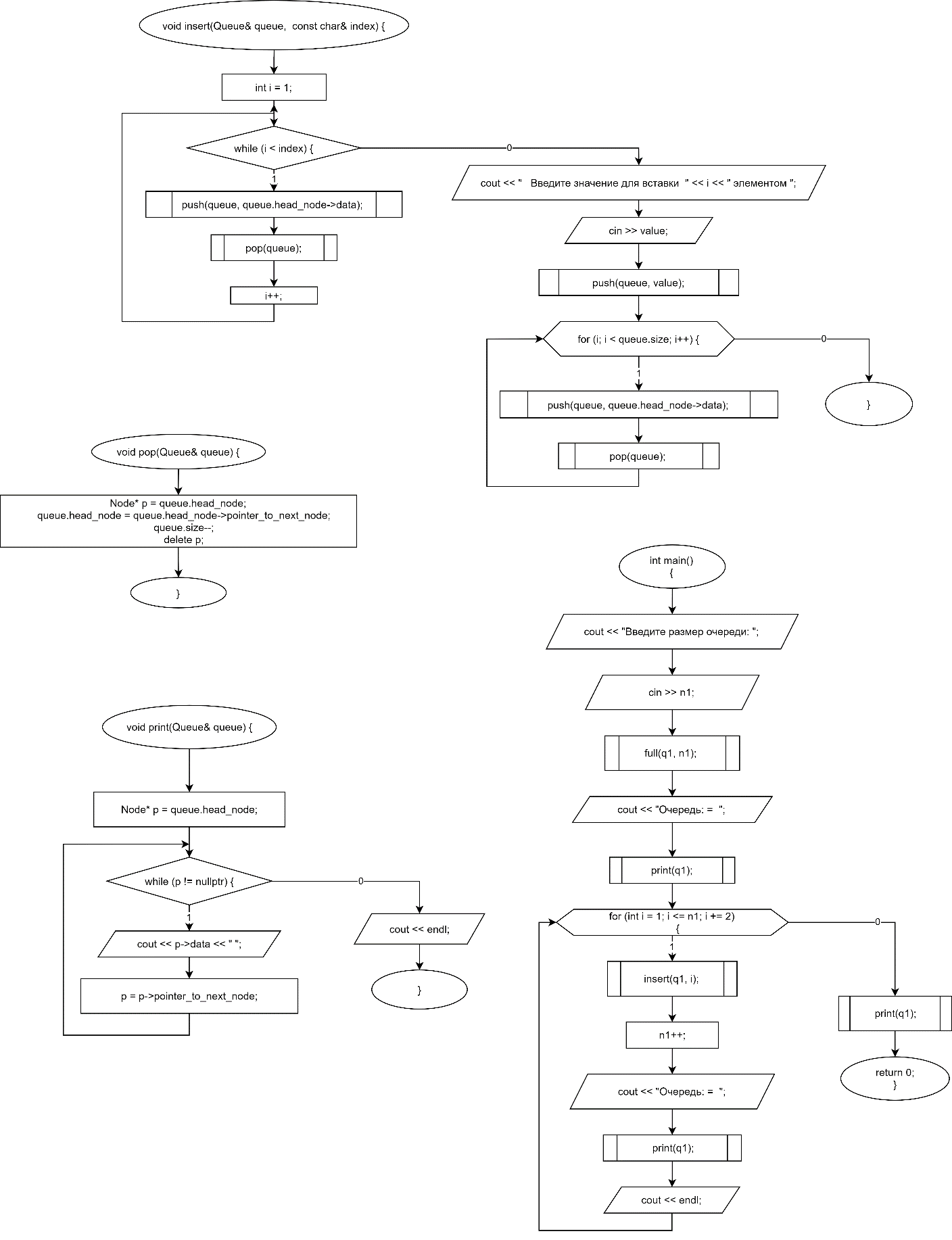
6. Функция insert вставляет новый элемент после заданного индекса. циклом while элементы перемещаются с начала очереди в конец (элемент head\_node сначала добавляется в конец функцией push, а потом удаляется сначала функцией pop).

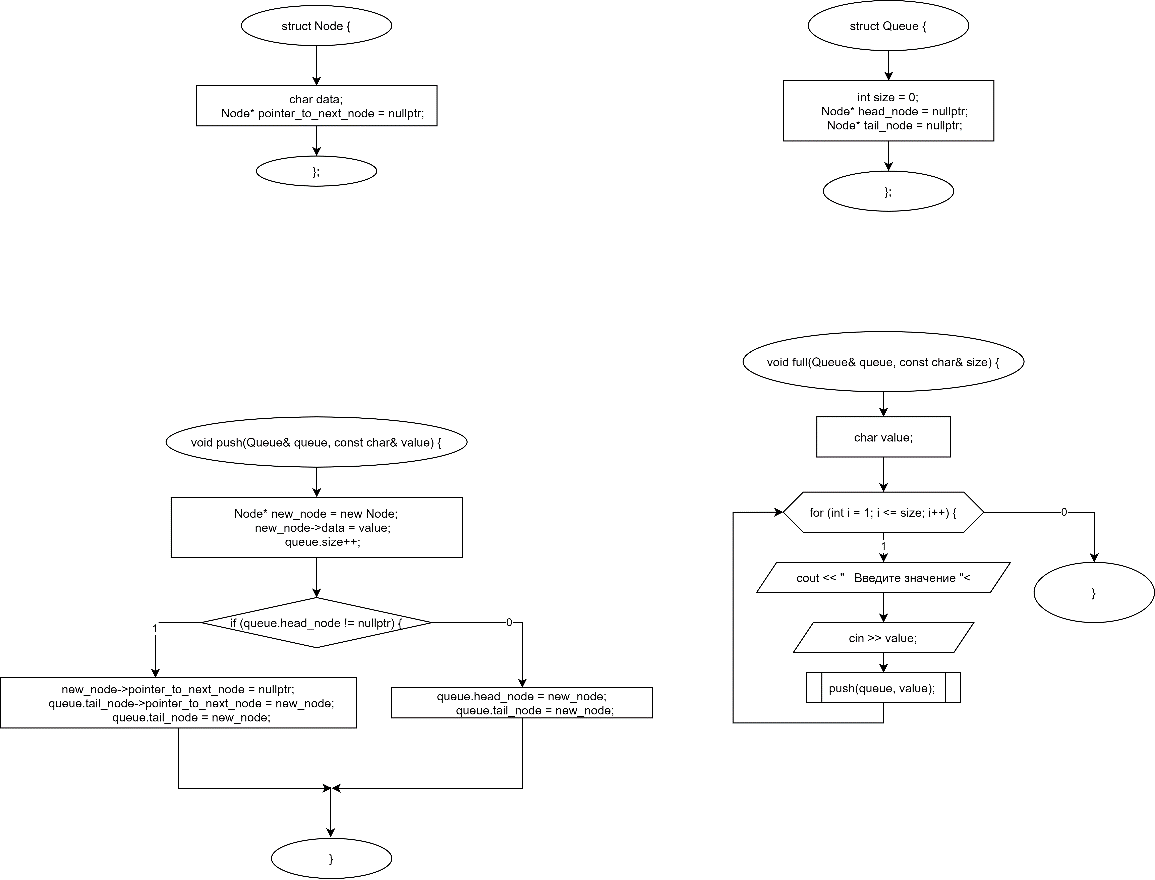
Элементы перемещаются пока цикл не дойдет до заданного индекса – элемент заданного индекса оказывается в конце очереди. После этого вводится значение для вставки и функцией push новый элемент добавляется в коней. Теперь после элемента с заданным индексом стоит введенный элемент.

Далее очередь прокручивается до конца (последовательно функциями push и pop), пока очередь не примет изначальное расположение элементов.

7. Функция int main(). Создается объект структуры очередь Queue q1. У пользователя запрашивается размер очереди, после чего она заполняется функцией full.

Далее циклом for с шагом в 2 для каждого нечетного элемента очереди вызывается функция insert. После каждого вызова insert выводится очередь функцией print.

**3. Блок-схема** 



**4. Исходный код**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

struct Point {

    string value;

    Point\* next = nullptr;

};

struct Queue {

    int size = 0;

    Point\* head = nullptr;

    Point\* tail = nullptr;

};

void push(Queue& queue, string& value) {

    Point\* new\_Point = new Point;

    new\_Point->value = value;

    queue.size++;

    if (queue.head != nullptr) {

        new\_Point->next = nullptr;

        queue.tail->next = new\_Point;

        queue.tail = new\_Point;

    }

    else {

        queue.head = new\_Point;

        queue.tail = new\_Point;

    }

}

void full(Queue& queue, int size) {

    string value;

    for (int i = 1; i <= size; i++) {

        cout << "Введите значение " << i << " элемента: ";

        cin >> value;

        push(queue, value);

    }

}

void print(Queue& queue) {

    cout << "Очередь: =  ";

    Point\* p = queue.head;

    while (p != nullptr) {

        cout << p->value << " ";

        p = p->next;

    }

    cout << '\n';

}

void pop(Queue& queue) {

    Point\* p = queue.head;

    queue.head = queue.head->next;

    queue.size--;

    delete p;

}

void insert(Queue& queue, int index) {

    int i = 1;

    while (i < index) {

        push(queue, queue.head->value);

        pop(queue);

        i++;

    }

    string value;

    cout << "Введите значение для вставки " << i << " элементом ";

    cin >> value;

    push(queue, value);

    for (i; i < queue.size; i++) {

        push(queue, queue.head->value);

        pop(queue);

    }

}

void addq(Queue & queue){

    ofstream out("file4.txt");

    Point\* current = queue.head;

    while (current != nullptr) {

        out << current->value << " ";

        current = current->next;

    }

    out.close();

}

void fromq(Queue& queue, int size){

    string buf;

    ifstream in("file4.txt");

    getline(in, buf);

    push(queue, buf);

    in.close();

}

int main(){

    Queue queue;

    int size, i;

    cout << "Введите размер очереди: ";

    cin >> size;

    full(queue, size);

    print(queue);

    cout << "номер\n";

    cin >> i;

    size++;

    insert(queue, i);

    print(queue);

    pop(queue);

    print(queue);

    addq(queue);

    return 0;

}

**5 Вывод в консоль**

