Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

направление подготовки: 09.03.01 - «Информатика и вычислительная техника»

**Отчет по Лабораторной работе Очередь**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

Выполнил студент гр. ИВТ-21-1б

Ахунов Руслан Булатович

Проверил:

Ст. Преподаватель кафедры ИТАС

Яруллин Денис Владимирович

(оценка) (подпись)

г. Пермь - 2022

**Постановка задачи:**

Вариант 2

Тип информационного поля char\*. Добавить в очередь элементы с номерами 1, 3, 5 и т. д.

**Анализ задачи:**

template <typename T>

struct Node

{

    T data;

    Node<T>\* next;

};

template <typename T>

struct Queue

{

    int size = 0;

    Node<T>\* head = nullptr;

    Node<T>\* tail = nullptr;

};

Структура Нода, поля для значения и указатель на следующую Ноду.

Структура Очередь, есть поля для размера очереди, а так же указатели типа Нода на начало и конец.

template <typename T>

void push(Queue<T>& q, const T& data)

{

    if (q.head != nullptr)

    {

        Node<T>\* newnode = new Node<T>;

        q.size++;

        newnode->data = data;

        newnode->next = nullptr;

        q.tail->next = newnode;

        q.tail = newnode;

    }

    else

    {

        Node<T>\* newnode = new Node<T>;

        newnode->data = data;

        q.head = newnode;

        q.tail = newnode;

        q.size = 1;

    }

}

Функция добавления элемента, сперва проверяем пуста ли очередь, если нет, то в конец очереди добавляем нашу Ноду, а так же увеличиваем размер очереди.

template <typename T>

void pop(Queue<T>& q)

{

    Node<T>\* tmp = q.head;

    q.head = q.head->next;

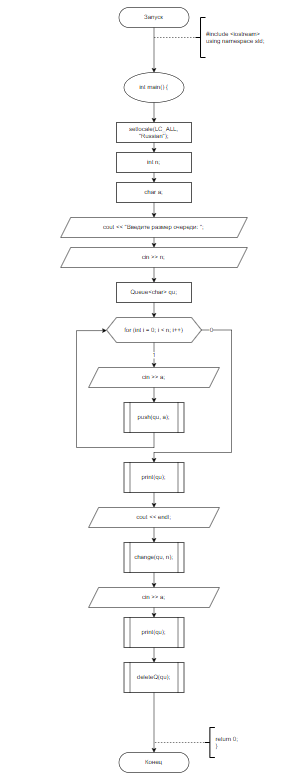
    q.size--;

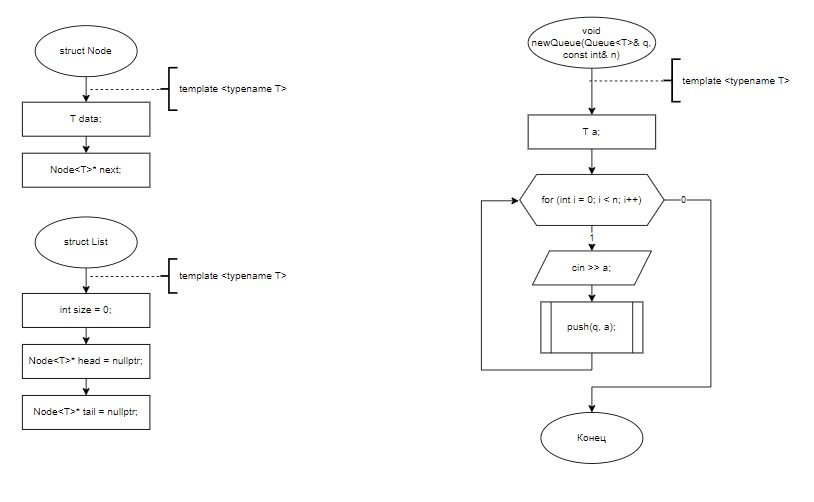
    delete tmp;

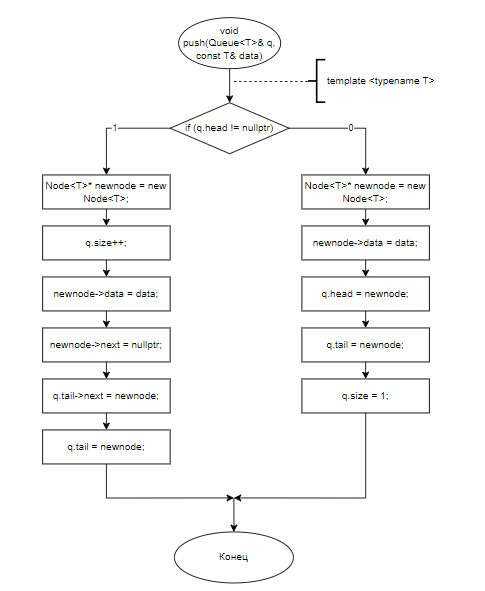
}

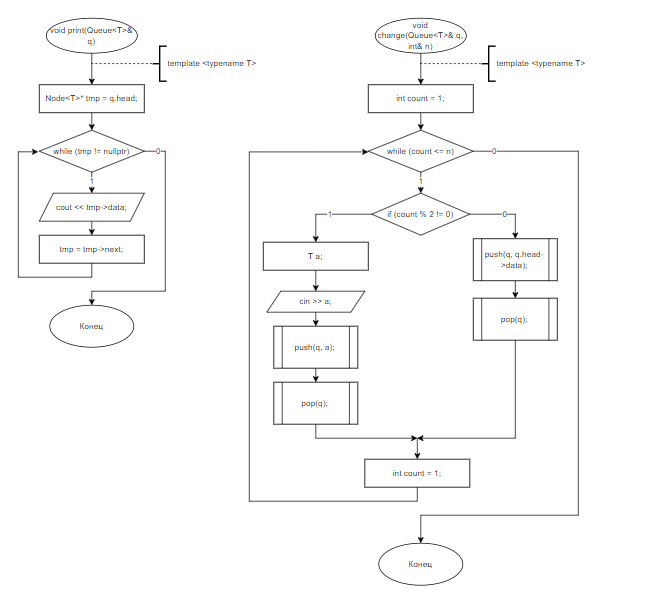
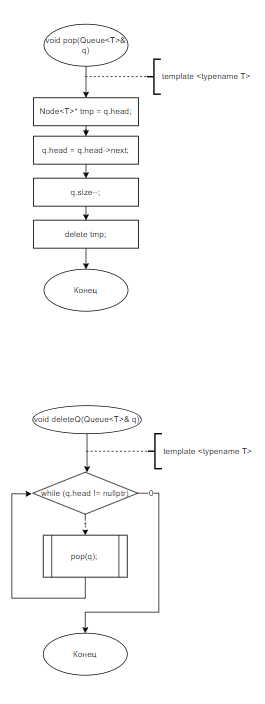
Функция удаления элемента. Начало списка переприсваиваем на следующий элемент. А текущий удаляем. Первый пришел, первым ушел.

**Блок-схема**

****

****

****

****

**Код программы**

#include <iostream>

using namespace std;

template <typename T>

struct Node

{

    T data;

    Node<T>\* next;

};

template <typename T>

struct Queue

{

    int size = 0;

    Node<T>\* head = nullptr;

    Node<T>\* tail = nullptr;

};

template <typename T>

void newQueue(Queue<T>& q, const int& n)

{

    T a;

    for (int i = 0; i < n; i++)

    {

        cin >> a;

        push(q, a);

    }

}

template <typename T>

void push(Queue<T>& q, const T& data)

{

    if (q.head != nullptr)

    {

        Node<T>\* newnode = new Node<T>;

        q.size++;

        newnode->data = data;

        newnode->next = nullptr;

        q.tail->next = newnode;

        q.tail = newnode;

    }

    else

    {

        Node<T>\* newnode = new Node<T>;

        newnode->data = data;

        q.head = newnode;

        q.tail = newnode;

        q.size = 1;

    }

}

template <typename T>

void pop(Queue<T>& q)

{

    Node<T>\* tmp = q.head;

    q.head = q.head->next;

    q.size--;

    delete tmp;

}

template <typename T>

void print(Queue<T>& q)

{

    Node<T>\* tmp = q.head;

    while (tmp != nullptr)

    {

        cout << tmp->data;

        tmp = tmp->next;

    }

}

template <typename T>

void change(Queue<T>& q, int& n)

{

    int count = 1;

    while (count <= n)

    {

        if (count % 2 != 0)

        {

            T a;

            cin >> a;

            push(q, a);

            pop(q);

        }

        else

        {

            push(q, q.head->data);

            pop(q);

        }

        count++;

    }

}

template <typename T>

void deleteQ(Queue<T>& q)

{

    while (q.head != nullptr)

    {

        pop(q);

    }

}

int main()

{

    setlocale(LC\_ALL, "Russian");

    int n;

    char a;

    cout << "Введите размер очереди: ";

    cin >> n;

    Queue<char> qu;

    for (int i = 0; i < n; i++)

    {

        cin >> a;

        push(qu, a);

    }

    print(qu);

    cout << endl;

    change(qu, n);

    cout << endl;

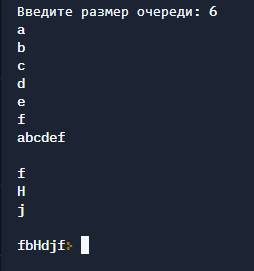
    print(qu);

    deleteQ(qu);

    return 0;

}

**Вывод программы**

****