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»

**О Т Ч Е Т  
по лабораторной работе Очереди №2**

**по дисциплине «Основы алгоритмизации и программирования»**

Выполнил студент гр. РИС-21-1б

Ибрагимов Марк Магамедович

(Фамилия, Имя, Отчество)

(подпись)

Проверил:

доцент

кафедры ИТАС

Полякова О.А.

(оценка) (подпись)

(дата)

Пермь 2022

**Постановка задачи**

Тип информационного поля double.

Удалить из очереди все элементы с четными номерами (2, 4, 6 и. т. д.).

**Анализ задачи**

1. С помощью структур Node, Queue, функции new\_queue и переменных a, n создаем очередь q
2. Используя функцию print\_queue, выводим очередь
3. Благодаря циклу for и функциям pop и push, которые удаляют и добавляют элемент в очередь, используем функцию delete\_even для удаления всех элементов с чётным номером
4. Для вывода окончательной очереди используем снова функцию print\_queue

**Описание переменных**

Node – структура, содержащая переменную data со значением элемента и указателем на следующий элемент prev

Queue – структура, содержащая переменную size со значением величины очереди и указателями head и tail, указывающие на начало и конец очереди

new\_queue – функция создающая очередь

n – целочисленная переменная со значением величины очереди

tmp – указатель, используемый для связки элементов очереди

i – счетчик

print\_queue – функция типа void, которая выводит очередь

pop – функция типа void, удаляющая элемент очереди

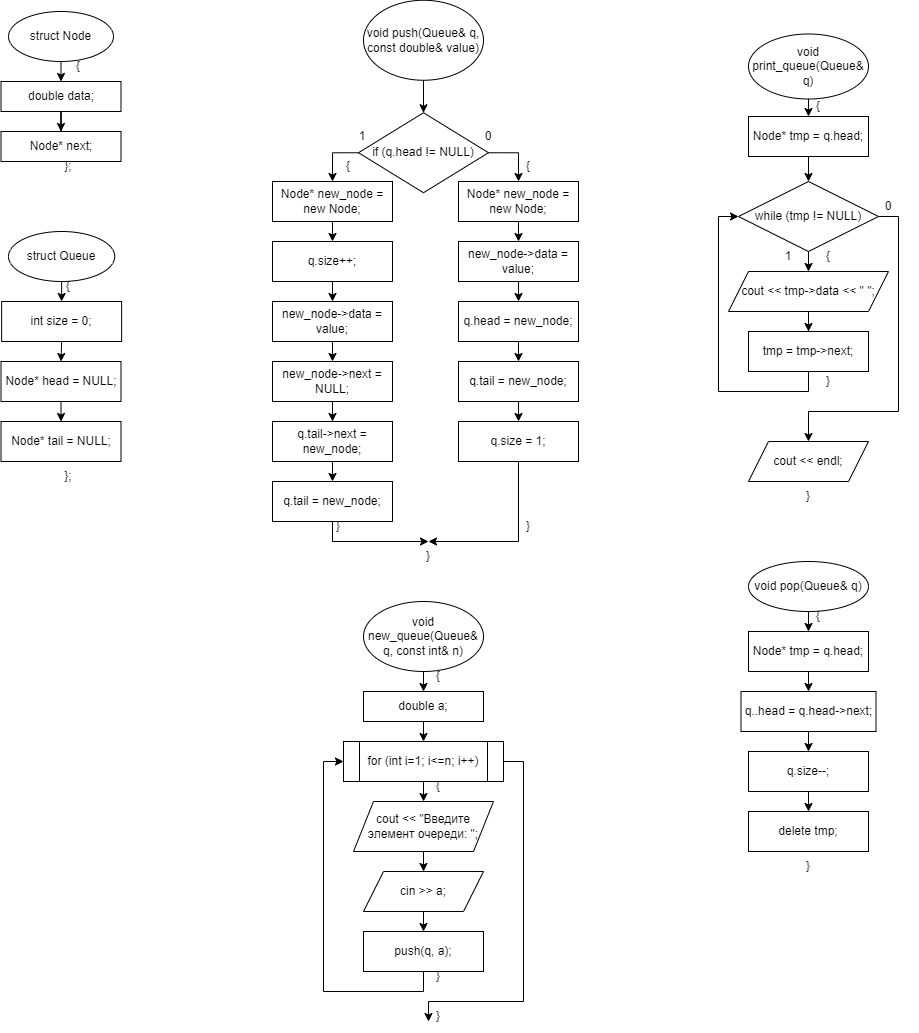
push – функция типа void, добавляющая элемент в очередь

a – вещественная переменная, принимающая значение элемента, который пользователь хочет добавить в очередь

q – очередь

delete\_even – функция типа void, удаляющая все элементы с четным номером в очереди

**Блок схема**

****

**Код программы**

#include <iostream>

using namespace std;

struct Node

{

double data;

Node\* next;

};

struct Queue

{

int size = 0;

Node\* head = NULL;

Node\* tail = NULL;

};

void push(Queue& q, const double& value)

{

if (q.head != NULL)

{

Node\* new\_node = new Node;

q.size++;

new\_node->data = value;

new\_node->next = NULL;

q.tail->next = new\_node;

q.tail = new\_node;

}

else

{

Node\* new\_node = new Node;

new\_node->data = value;

q.head = new\_node;

q.tail = new\_node;

q.size = 1;

}

}

void new\_queue(Queue& q, const int& n)

{

double a;

for (int i=1; i<=n; i++)

{

cout << "Введите элемент очереди: ";

cin >> a;

push(q, a);

}

}

void print\_queue(Queue& q)

{

Node\* tmp = q.head;

while (tmp != NULL)

{

cout << tmp->data << " ";

tmp = tmp->next;

}

cout << endl;

}

void pop(Queue& q)

{

Node\* tmp = q.head;

q.head = q.head->next;

q.size--;

delete tmp;

}

void delete\_even(Queue& q)

{

int i = 1;

while (i <= q.size)

{

if (i % 2 == 0)

pop(q);

else

{

push(q, q.head->data);

pop(q);

i++;

}

}

}

int main() {

setlocale(LC\_ALL,"Rus");

Queue q;

int n;

do

{

cout << "Введите количество элементов очереди: ";

cin >> n;

}

while (n <= 0);

new\_queue(q, n);

cout << endl;

print\_queue(q);

delete\_even(q);

cout << endl;

print\_queue(q);

return 0;

}

**Примеры выполнения программы**

**Анализ результатов**

Программа сработала корректно и вывела необходимые результаты.