Московский авиационный институт

(Национальный исследовательский университет)

Факультет №3

Кафедра 304

Отчёт по лабораторной работе

По учебной дисциплине

Структуры и алгоритмы обработки данных

На тему:  
“Работа с очередью”

Выполнил:

Плотников Кирилл

Крещук Николай

Группа: М30-224БК-17

Бригада №7

Принял:

Силаев А.В.

Москва, 2018

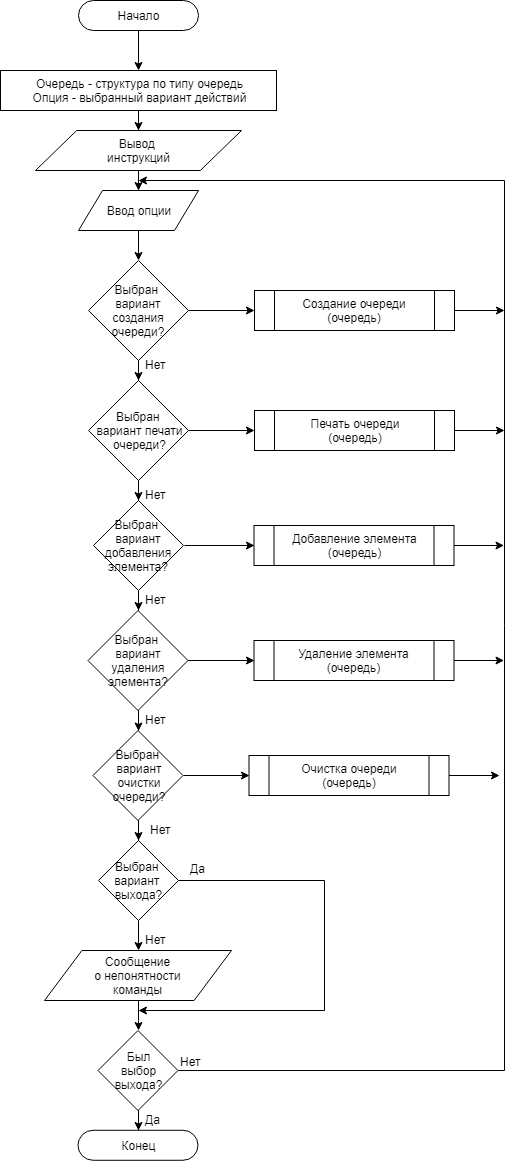
**Задание.**

Программно реализовать список по типу очереди, где вид списка – циклический массив, тип элементов – вещественный, а количество элементов – 19, включающие следующие операции:

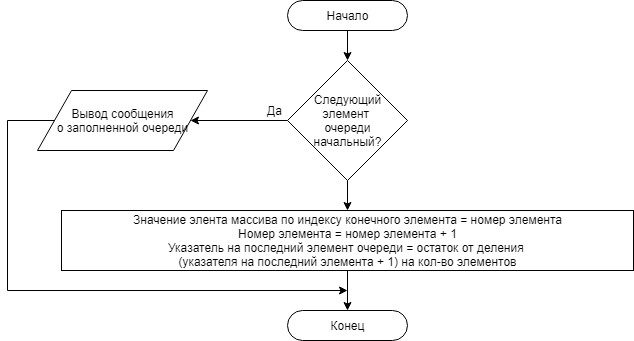
1. Создание списка.
2. Вывод на экран или в файл значений элементов списка с их индексами (номерами).
3. Включение нового элемента в список.
4. Выборка элемента из списка.
5. Очистка списка.

Структурная схема программы.

Функция main



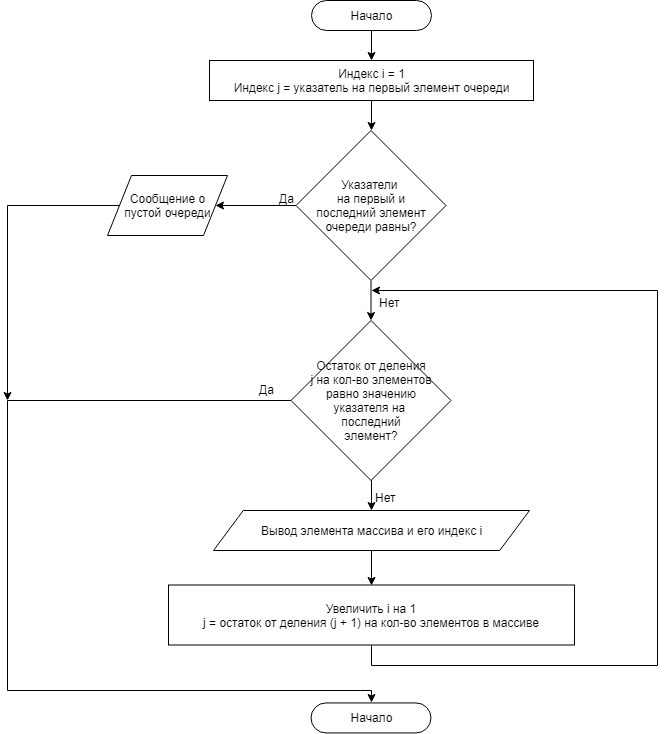
Функция добавления элемента.



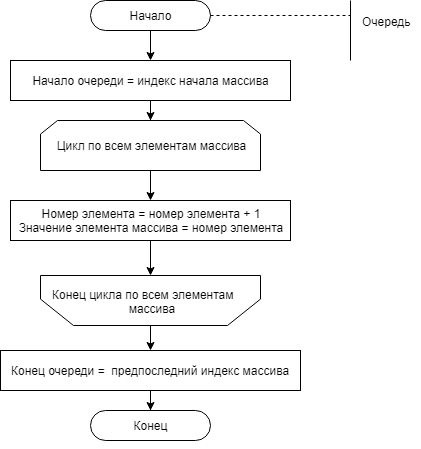
Функция удаления списка



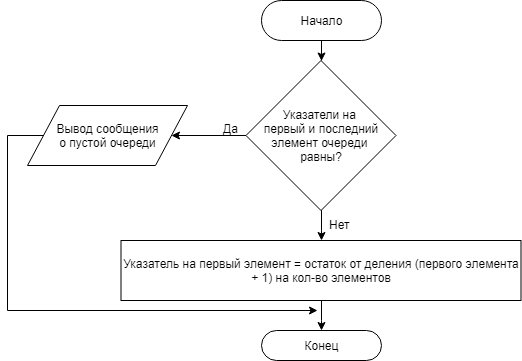
Функция печати очереди



Функция создания очереди



Функция удаления элемента



**Код программы.**

﻿/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Project Name :Lab\_3 \*

\* Project Type :Win32 Console Application \*

\* File Name :Lab\_3.cpp \*

\* Language :C++, MAVS 2010 and above \*

\* Programmer(2) :Крещук Николай, Плотников Кирилл, вариант №7 \*

\* Created :28/10/18 \*

\* Last revision :28/10/18 \*

\* Comment :Работа с очередью \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "pch.h"

#include <iostream>

using namespace std;

// Размер очереди.

const int N = 20;

int number = 0;

struct QueueCycleArray

{

double data[N];

int first = 0;

int last = 0;

};

void Creation(QueueCycleArray \*Q)

{

Q->first = 0;

for (int i = 0; i < N; i++)

{

number++;

Q->data[i] = number;

}

Q->last = N - 1;

}

void PrintQueue(QueueCycleArray \*Q)

{

int i = 1;

int j = Q->first;

if (Q->first == Q->last)

{

cout << "Список пуст." << endl;

return;

}

while (j % N != Q->last)

{

cout << "Элемент: " << Q->data[j] << " Индекс: " << i++ << endl;

j = (j + 1) % N;

}

}

void AddElement(QueueCycleArray \*Q)

{

if ((Q->last + 1) % N == Q->first)

{

cout << "Список заполен." << endl;

return;

}

Q->data[Q->last] = number++;

Q->last = (Q->last + 1) % N;

}

void DeleteElement(QueueCycleArray \*Q)

{

if (Q->first == Q->last)

{

cout << "Список пуст." << endl;

return;

}

Q->first = (Q->first + 1) % N;

}

void ClearQueue(QueueCycleArray \*Q)

{

Q->first = 0;

Q->last = 0;

}

void main()

{

setlocale(LC\_ALL, "Rus");

QueueCycleArray Q;

char number;

cout << "1. Создать список." << endl;

cout << "2. Вывод списка." << endl;

cout << "3. Добавление элемента в список." << endl;

cout << "4. Удалить элемент." << endl;

cout << "5. Удалить список." << endl;

cout << "0. Выйти." << endl;

cout << "Номер команды: ";

do

{

cin >> number;

switch (number)

{

case '1':

Creation(&Q);

break;

case '2':

PrintQueue(&Q);

break;

case '3':

AddElement(&Q);

break;

case '4':

DeleteElement(&Q);

break;

case '5':

ClearQueue(&Q);

break;

case '0':

break;

default:

cout << "Команда не определена." << endl;

break;

}

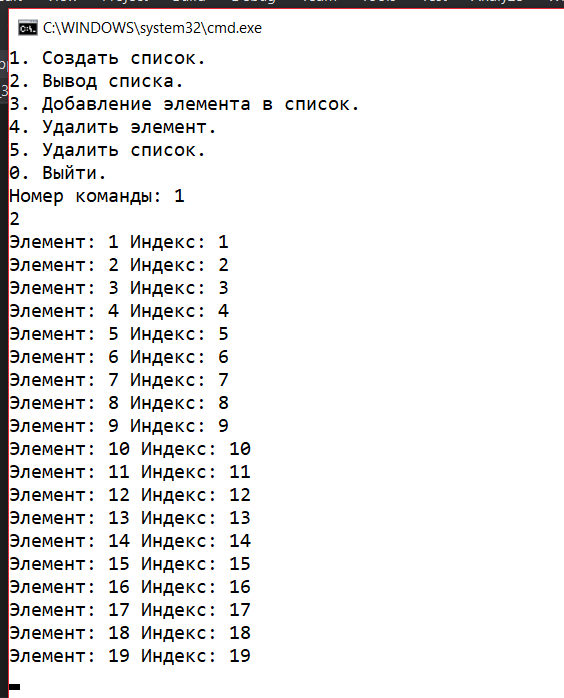
} while (number != '0');

system("pause");

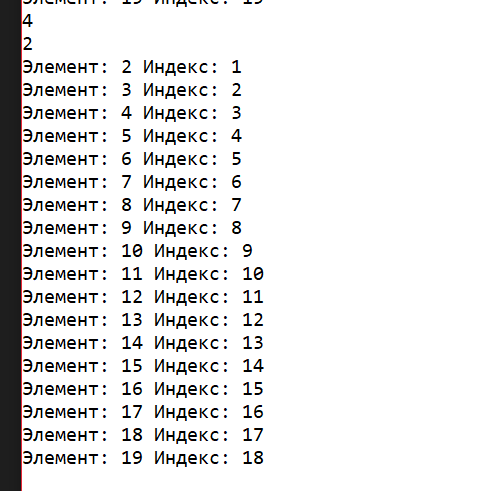
}

**Результаты работы программы.**

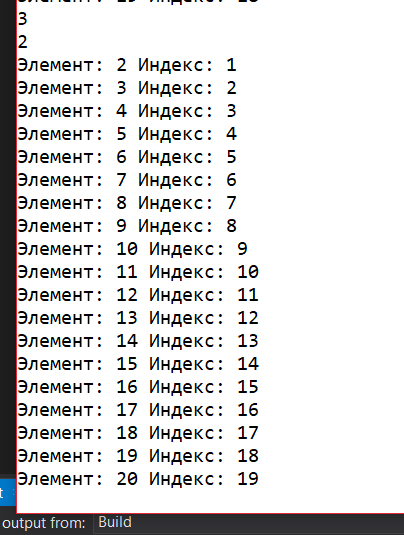
1. Создание списка.



2. Удаление элемента из списка.



3. Добавление элемента в список.



**Выводы.**

При реализации списка по типу очереди с помощью циклического массива имеются достоинства, такие как:

1. Удаление элемента из очереди за константное время, не зависящее от размера самой очереди.
2. Доступ к самим элементам осуществляется также за константное время, благодаря тому, что используется массив.

Существует также ряд недостатков, например то, что массив имеет фиксированную длину, из-за этого размер очереди статичен, и не может быть больше размера самого массива. В циклическом массиве также можно хранить кол-во элементов равной кол-во элементов в массиве минус 1, из-за указателя на последний элемент очереди.