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»

**О Т Ч Е Т**

**по лабораторной работе №11.4 (Очереди)**

**по дисциплине**

**«Основы алгоритмизации и программирования» семестр 2**

Выполнил студент гр. ИВТ-21-1б

Ипатов Дмитрий Сергеевич

Проверил:

Ст. Преподаватель кафедры ИТАС

Яруллин Д.В.

(оценка) (подпись)

г. Пермь-2022

**Постановка задачи:**

1. Сформировать очередь. Тип информационного поля указан в варианте.

2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

4. Распечатать полученный результат.

5. Удалить соответствующую структуру из памяти.

Выполнить двумя способами (без библиотеки queue и с ней).

Анализ задачи:

1 способ

1. Создаем структуру Node с информационным полем типа double и указателем на следующий элемент;
2. В функции main просим пользователя ввести количество элементов, которое мы передадим в функцию add;
3. В функции add типа Node\* (структуры), создаем переменную k типа double, указатель на первый элемент в очереди, выделяем под него память и создаем два дополнительных указателя;
4. Пользователь вводит элемент вне цикла, и он становится первым (присваивается переменной first);
5. Заходим в цикл и выделяем память под новый элемент, пользователь вводит значение с клавиатуры, меняем указатели;
6. В функции main вызываем функцию для вывода элементов на экран - print типа void, в которую передаем начало очереди. В ней пока элементы не пусты мы проверяем информационные поля на условие больше ли они нуля, если больше, то выводим это инф. Поле, если нет, то смещаем указатель на следующий элемент (пропускаем отрицательный элемент);

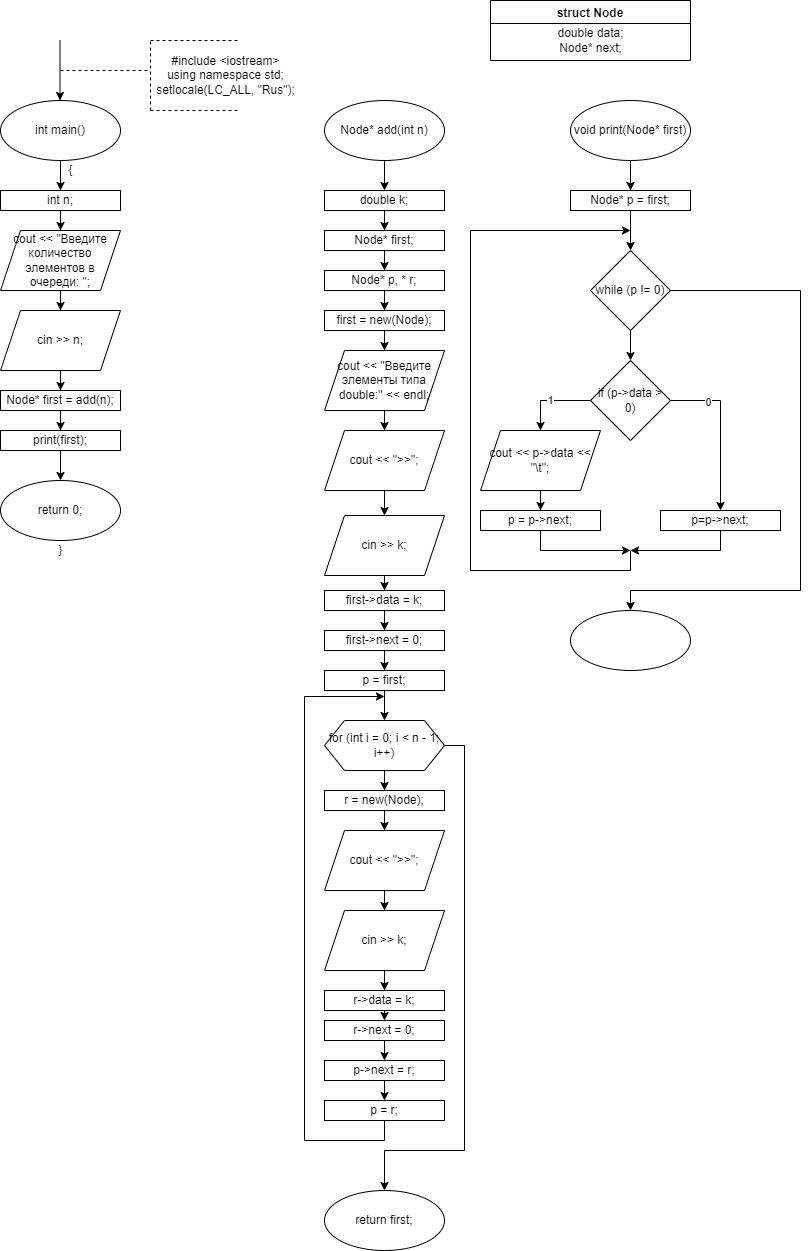
2 способ

1. Подключаем библиотеку <queue>, создаем переменную int n(количество элементов), double k(переменная для ввода данных);
2. Создаем очередь с типом данных double с названием q;
3. Вводим кол-во элементов;
4. Заходим в цикл, который работает до тех пор по i не дойдет до значения равного n;
5. В цикле вводим значение, которое тут же проверятся на условие больше ли оно нуля, если больше, то оно пушится в очередь, если нет, то просто пропускается (не пушится);
6. Далее выводим результат, но сначала проверяем пуста очередь или нет, если пуста, то выводим это на экран, если нет, то входим в цикл while,

в котором пока очередь не пуста выводим первый стоящий элемент в очереди и потом удаляем его.

**1 способ (без библиотеки queue)**

**Блок схема:**

****

**Код программы:**

#include <iostream>

using namespace std;

struct Node

{

double data;

Node\* next;

};

Node\* add(int n)

{

double k;

Node\* first;

Node\* p, \* r;

first = new(Node);

cout << "Введите элементы типа double:" << endl;

cout << ">>";

cin >> k;

first->data = k;

first->next = 0;

p = first;

for (int i = 0; i < n - 1; i++)

{

r = new(Node);

cout << ">>";

cin >> k;

r->data = k;

r->next = 0;

p->next = r;

p = r;

}

return first;

}

void print(Node\* first)

{

Node\* p = first;

while (p != 0)

{

if (p->data > 0)

{

cout << p->data << "\t";

p = p->next;

}

else

{

p=p->next;

}

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

int n;

cout << "Введите количество элементов в очереди: ";

cin >> n;

Node\* first = add(n);

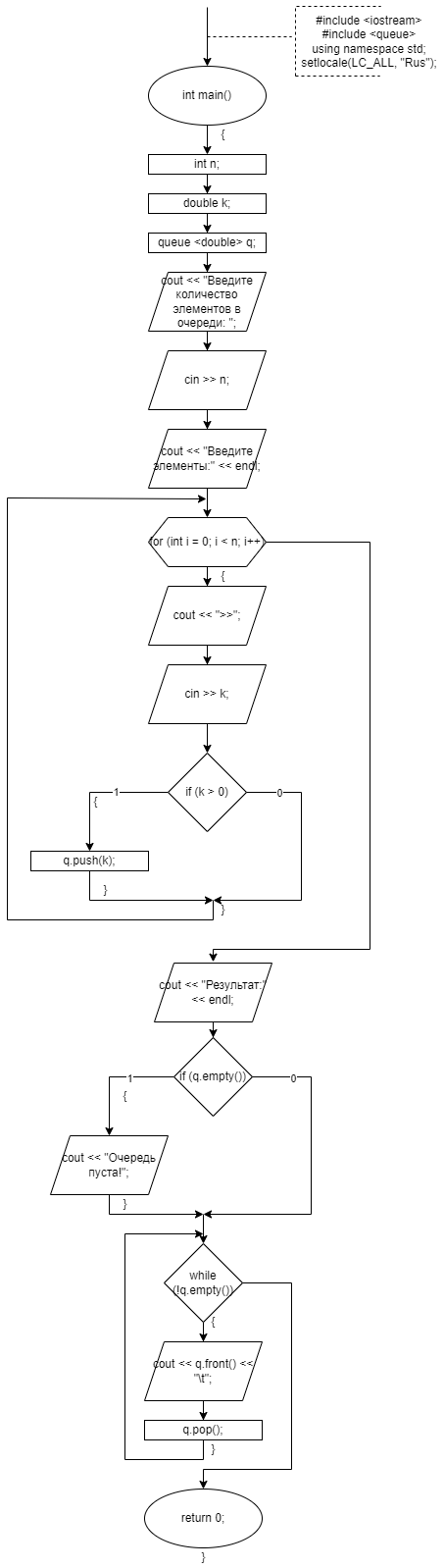
print(first);

return 0;

}

**2 способ (с библиотекой queue)**

**Блок схема:**

****

**Код программы:**

#include <iostream>

#include <queue>

using namespace std;

int main()

{

int n;

double k;

setlocale(LC\_ALL, "Rus");

queue <double> q;

cout << "Введите количество элементов в очереди: ";

cin >> n;

cout << "Введите элементы:" << endl;

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

{

cout << ">>";

cin >> k;

if (k > 0)

{

q.push(k);

}

}

cout << "Результат:" << endl;

if (q.empty()) cout << "Очередь пуста!";

while (!q.empty())

{

cout << q.front() << "\t";

q.pop();

}

return 0;

}