Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа**

Выполнил:   
студент группы РИС-23-1б   
Молодых Никита Андреевич

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

Пермь, 2024 г.

**«Очередь»**

**Условие:**

Написать программу, в которой создаются динамические

структуры и выполнить их обработку в соответствии со

своим вариантом.

Записи в линейном списке содержат ключевое поле типа

\*char(строка символов). Сформировать двунаправленный

список. Удалить К элементов из конца списка. Добавить

элемент после элемента с заданным ключом.

**Анализ задачи:**

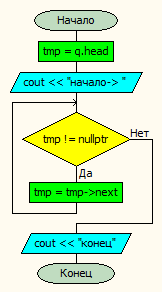
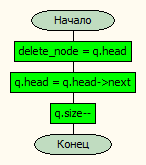
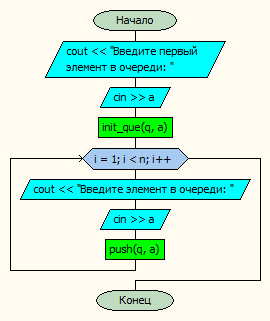
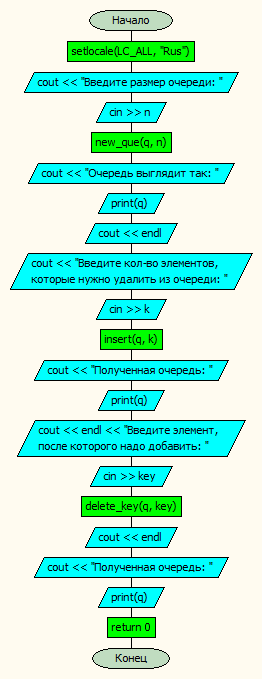
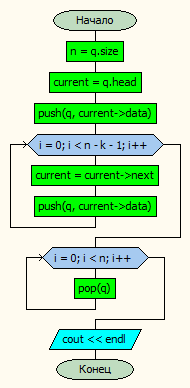
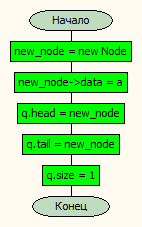
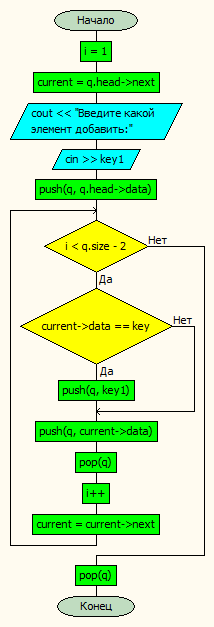
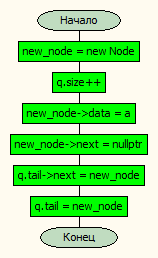
1)Для выполнения данной задачи необходимо создать структуру данных для хранения записей в виде линейного списка. В каждой записи будет содержаться ключевое поле типа \*char (строка символов). Затем необходимо сформировать двунаправленный список, где каждый элемент будет содержать указатель как на предыдущий, так и на следующий элемент.

2)После этого нужно реализовать функцию удаления K последних элементов из списка. Для этого можно использовать указатель на конец списка и последовательно освобождать память для K последних элементов.

3)Далее необходимо реализовать функцию добавления элемента после элемента с заданным ключом. Для этого нужно осуществить поиск элемента с заданным ключом в списке, затем создать новый элемент и вставить его после найденного элемента, корректно перенаправив указатели.

4)Таким образом, программа будет создавать и обрабатывать динамические структуры данных в виде двунаправленного списка, удалять элементы из конца списка и добавлять новые элементы после элемента с заданным ключом.

**Блок-Схема:**



**Код на языке C++:**

#include<iostream>

using namespace std;

struct Node {

char data;

Node\* next;

};

struct Queue {

int size = 0;

Node\* head;

Node\* tail;

};

void init\_que(Queue& q, const char a) {

Node\* new\_node = new Node;

new\_node->data = a;

q.head = new\_node;

q.tail = new\_node;

q.size = 1;

}

void push(Queue& q, char a) {

Node\* new\_node = new Node;

q.size++;

new\_node->data = a;

new\_node->next = nullptr;

q.tail->next = new\_node;

q.tail = new\_node;

}

void print(Queue& q) {

Node\* tmp = q.head;

cout << "начало-> ";

while (tmp != nullptr) {

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

tmp = tmp->next;

}

cout << "конец";

}

void pop(Queue& q) {

Node\* delete\_node = q.head;

q.head = q.head->next;

q.size--;

delete delete\_node;

}

void delete\_key(Queue& q, char key) {

int i = 1;

char key1;

Node\* current = q.head->next;

cout << "Введите какой элемент добавить:";

cin >> key1;

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

while (i < q.size -2) {

if (current->data == key) {

push(q, key1);

}

push(q, current->data);

pop(q);

i++;

current = current->next;

}

pop(q);

}

void insert(Queue& q, int k) {

int n = q.size;

Node\* current = q.head;

push(q, current->data);

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

current = current->next;

push(q, current->data);

}

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

pop(q);

}

cout << endl;

}

void new\_que(Queue& q, int n) {

char a;

cout << "Введите первый элемент в очереди: ";

cin >> a;

init\_que(q, a);

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

cout << "Введите элемент в очереди: ";

cin >> a;

push(q, a);

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

Queue q;

int n, k;

cout << "Введите размер очереди: ";

cin >> n;

new\_que(q, n);

cout << "Очередь выглядит так: ";

print(q);

char key;

cout << endl;

cout << "Введите кол-во элементов, которые нужно удалить из очереди: ";

cin >> k;

insert(q, k);

cout << "Полученная очередь: ";

print(q);

cout << endl << "Введите элемент, после которого надо добавить: ";

cin >> key;

delete\_key(q, key);

cout << endl;

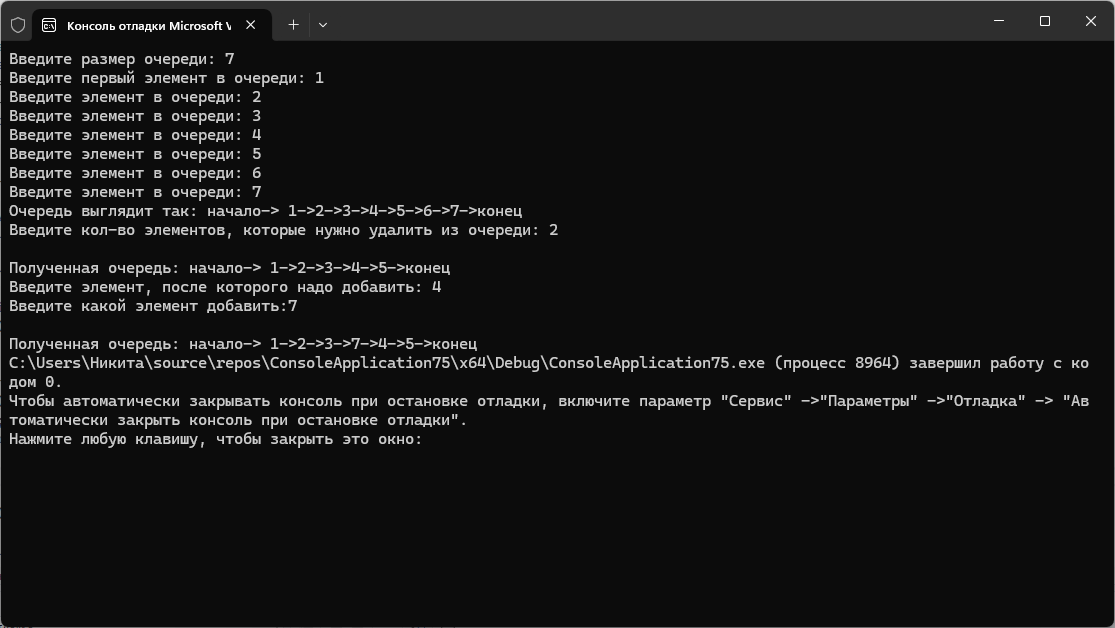
cout << "Полученная очередь: ";

print(q);

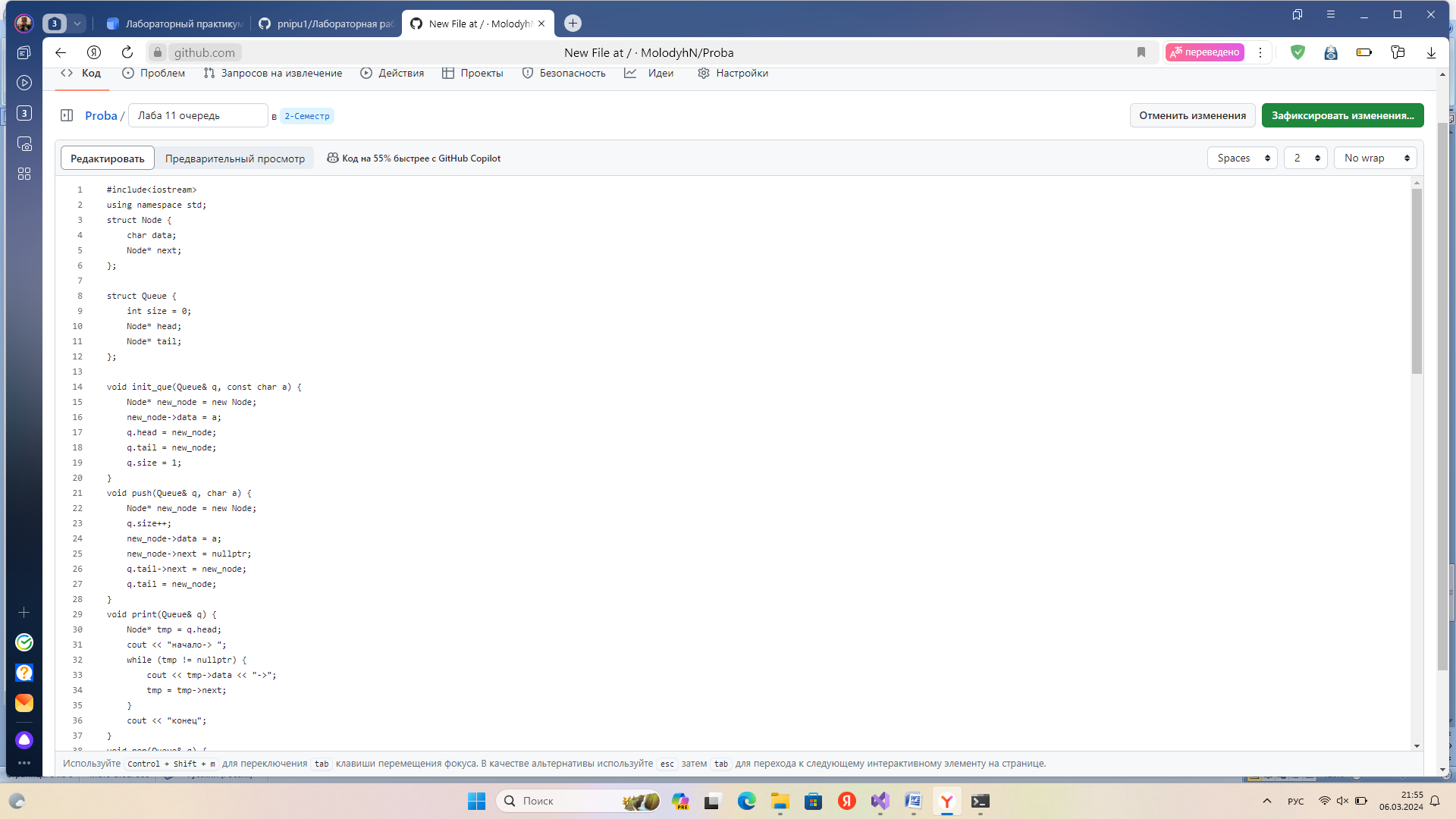
return 0;

}

**Работа программы:**



**Cкрины из гита:**

****

https://github.com/MolodyhN/Proba

**Вывод:** Задача была выполнена. Всё получилось.