Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №11.2**

Дисциплина: «Основы алгоритмизации и программирования»

Тема.Информационные и динамические структуры. Очереди

Вариант 1

Выполнила:

Студент группы РИС-20-1б

Азмагулов Артём Вадимович

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

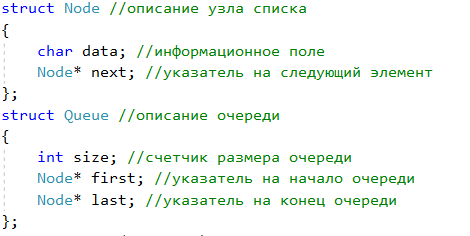
**Пермь, 2021**

Очереди

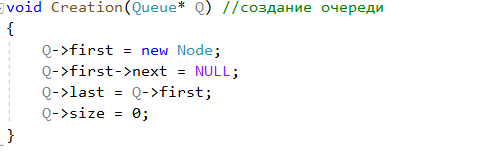
**Анализ задач**

**1.** Для решения задачи необходимо:

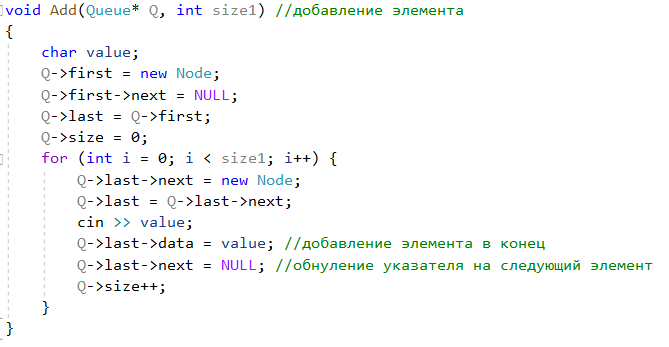
1.1.Создать структуру переменных



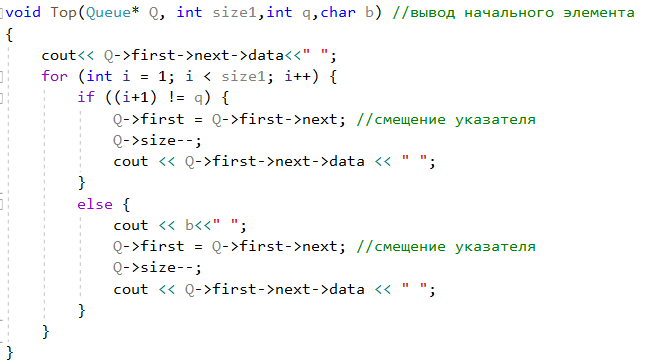
**1.2.** Разработать метод по созданию очереди:



**1.3.**Разработать функцию для вставки элемента в очередь:



**1.4.** Разработать функцию по выводу элементов очереди на консоль:



**2.**В ходе работы были использованы следующие типы данных:

**2.1.** Для функции Creation используется указатель на очередь



**2.2.** Для функции Top используется указатель на очередь, номер нового элемента и размер очереди типа int , и его значение типа char:



2.3 Для функции Add используется указатель на очередь и размер очереди типа int.



**3.**Для решения задачи данные были представлены в следующем виде:

**3.1.** Для удобной обработки элементов было решено представить элементы в виде очереди

**4.**Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.**Ввод данных происходит благодаря функции cin:

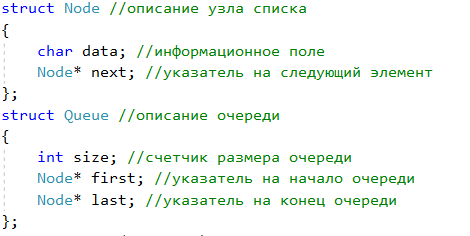


**4.2.** Вывод данных на консоль реализован с помощью оператора cout.

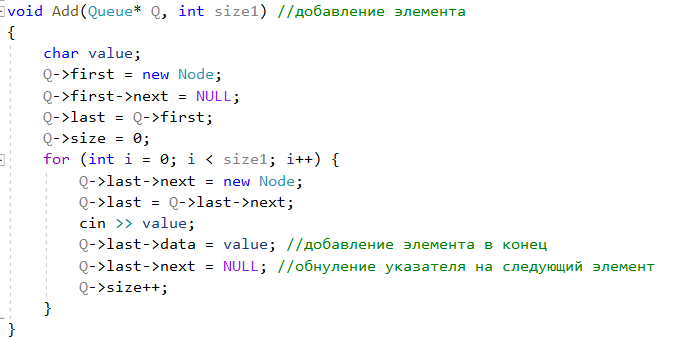


**5.**Поставленные задачи будут решены следующими действиями:

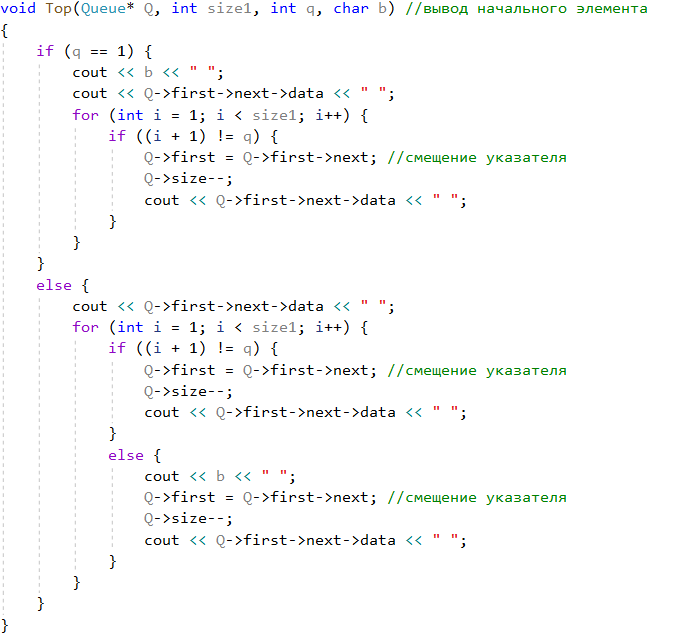
**5.1.** Создание очереди начинается с ее объявления:



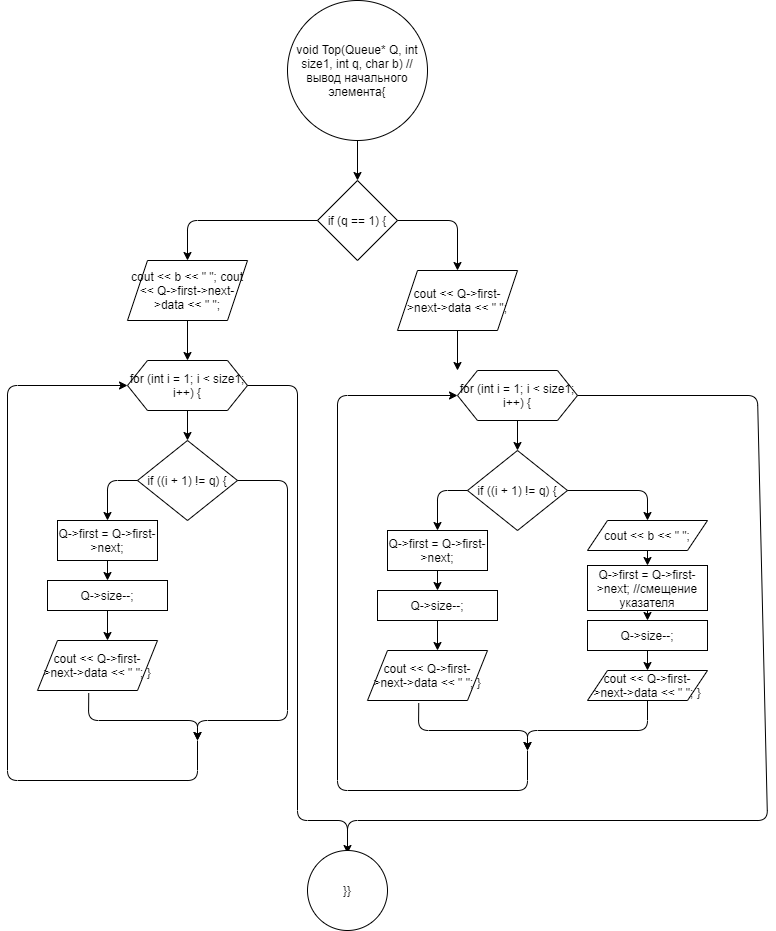
**5.2.**Добавление элементов в очередь происходит благодаря циклу for, присваивающий концу очереди новый входящий элемент :

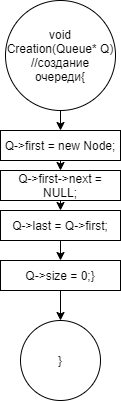


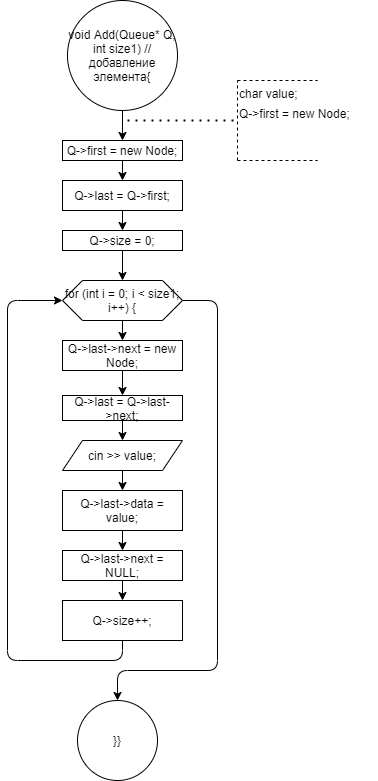
**5.3** Вывод элементов происходит благодаря функции Top, выводящий элементы до вставляемого элемента, затем следует вставка нового элемента и вывод оставшихся элементов:

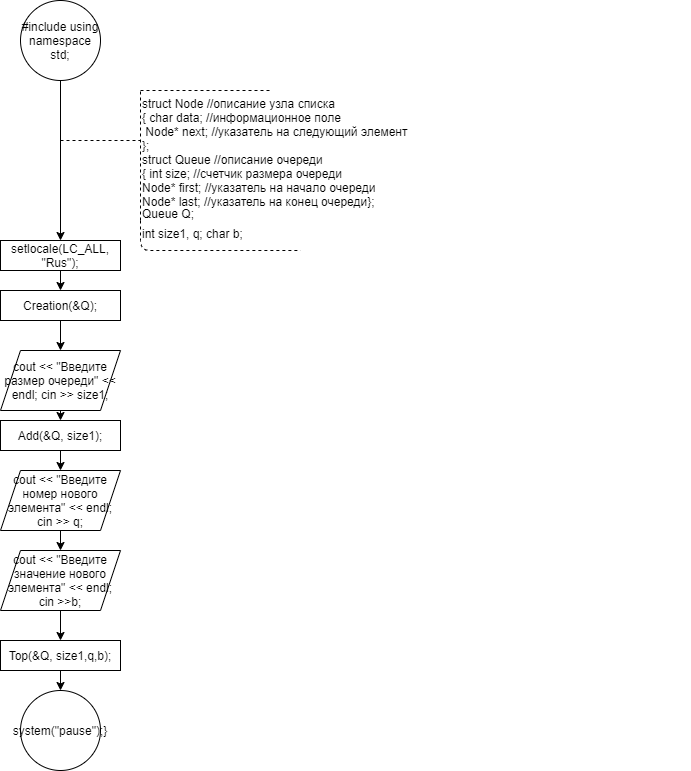


**Блок-схема программы**









**Решение**

#include <iostream>

using namespace std;

struct Node //описание узла списка

{

char data; //информационное поле

Node\* next; //указатель на следующий элемент

};

struct Queue //описание очереди

{

int size; //счетчик размера очереди

Node\* first; //указатель на начало очереди

Node\* last; //указатель на конец очереди

};

void Creation(Queue\* Q) //создание очереди

{

Q->first = new Node;

Q->first->next = NULL;

Q->last = Q->first;

Q->size = 0;

}

void Top(Queue\* Q, int size1, int q, char b) //вывод начального элемента

{

if (q == 1) {

cout << b << " ";

cout << Q->first->next->data << " ";

for (int i = 1; i < size1; i++) {

if ((i + 1) != q) {

Q->first = Q->first->next; //смещение указателя

Q->size--;

cout << Q->first->next->data << " ";

}

}

}

else {

cout << Q->first->next->data << " ";

for (int i = 1; i < size1; i++) {

if ((i + 1) != q) {

Q->first = Q->first->next; //смещение указателя

Q->size--;

cout << Q->first->next->data << " ";

}

else {

cout << b << " ";

Q->first = Q->first->next; //смещение указателя

Q->size--;

cout << Q->first->next->data << " ";

}

}

}

}

void Add(Queue\* Q, int size1) //добавление элемента

{

char value;

Q->first = new Node;

Q->first->next = NULL;

Q->last = Q->first;

Q->size = 0;

for (int i = 0; i < size1; i++) {

Q->last->next = new Node;

Q->last = Q->last->next;

cin >> value;

Q->last->data = value; //добавление элемента в конец

Q->last->next = NULL; //обнуление указателя на следующий элемент

Q->size++;

}

}

void main() //главная функция

{

setlocale(LC\_ALL, "Rus");

Queue Q;

Creation(&Q);

int size1, q;

char b;

cout << "Введите размер очереди" << endl;

cin >> size1;

Add(&Q, size1);

cout << "Введите номер нового элемента" << endl;

cin >> q;

cout << "Введите значение нового элемента" << endl;

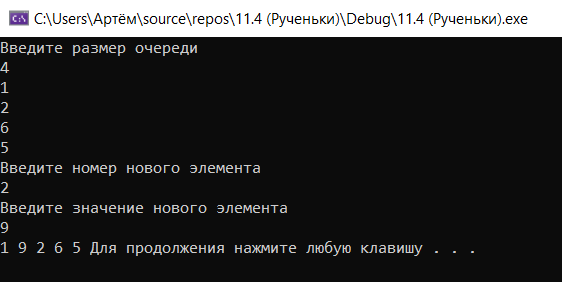
cin >>b;

Top(&Q, size1,q,b);

system("pause");

}

**Скриншоты**

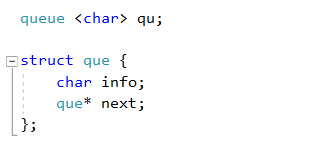


2) Очередь(stl)

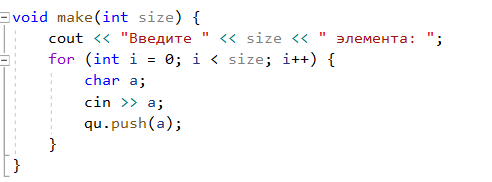
**Анализ задач**

**1.** Для решения задачи необходимо:

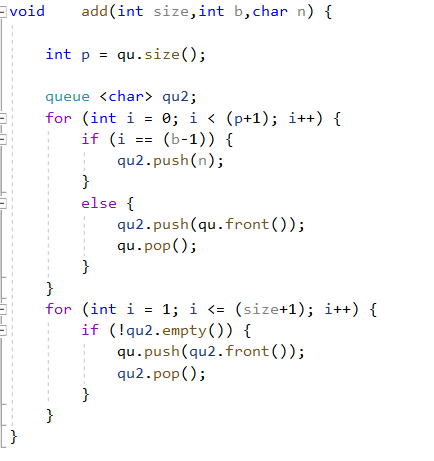
1.1.Создать структуру переменных



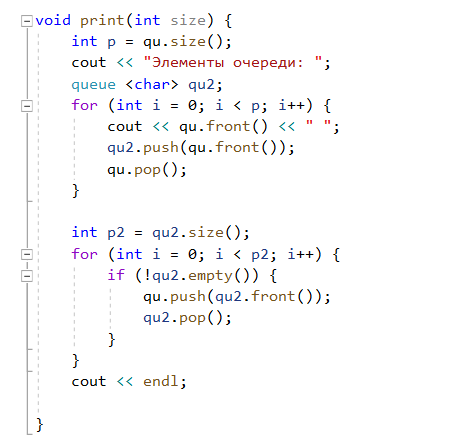
**1.2.** Разработать метод по созданию очереди:



**1.3.**Разработать функцию для вставки элемента в очередь:



**1.4.** Разработать функцию по выводу элементов очереди на консоль:



**2.**В ходе работы были использованы следующие типы данных:

**2.1.** Для функции make используется целочисленный размер очереди



**2.2.** Для функции add используется размер очереди, номер нового элемента очереди и его значение:



2.3 Для функции размер очереди типа int.



**3.**Для решения задачи данные были представлены в следующем виде:

**3.1.** Для удобной обработки элементов было решено представить элементы в виде очереди

**4.**Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.**Ввод данных происходит благодаря функции cin:

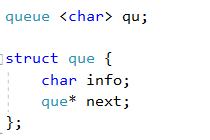


**4.2.** Вывод данных на консоль реализован с помощью оператора cout:

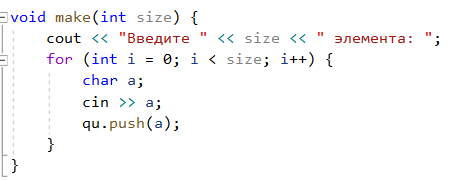


**5.**Поставленные задачи будут решены следующими действиями:

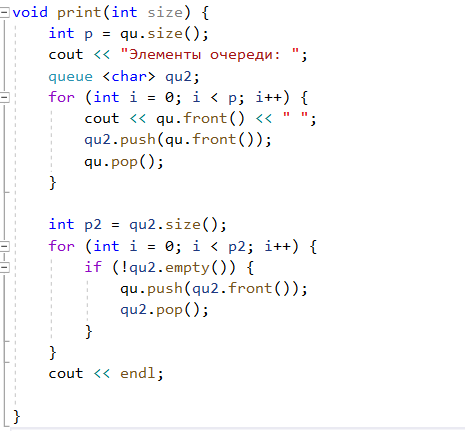
**5.1.** Создание очереди начинается с ее объявления:



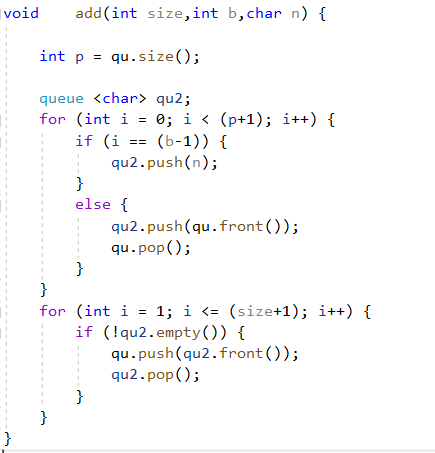
**5.2.**Добавление элементов в очередь происходит благодаря циклу for, присваивающий концу очереди новый входящий элемент :



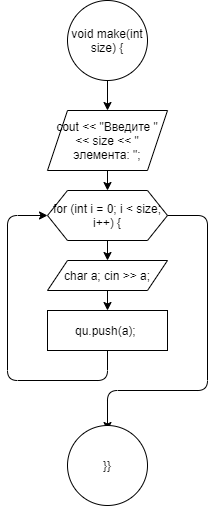
**5.3** Вывод элементов происходит благодаря функции print, выводящий все элементы до вставляемого элемента с последующим присвоением элементов новой структуре

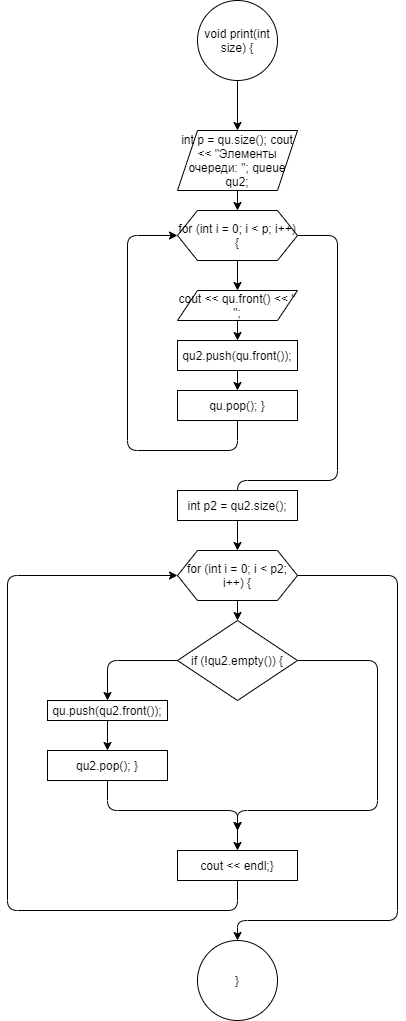


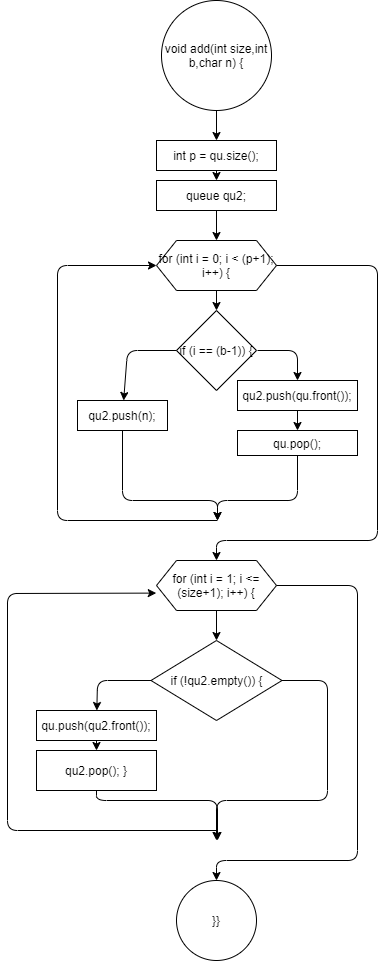
**5.4.** Добавление нового элемента происходит посредством присвоения элементов одной очереди другой, с добавлением нового элемента:

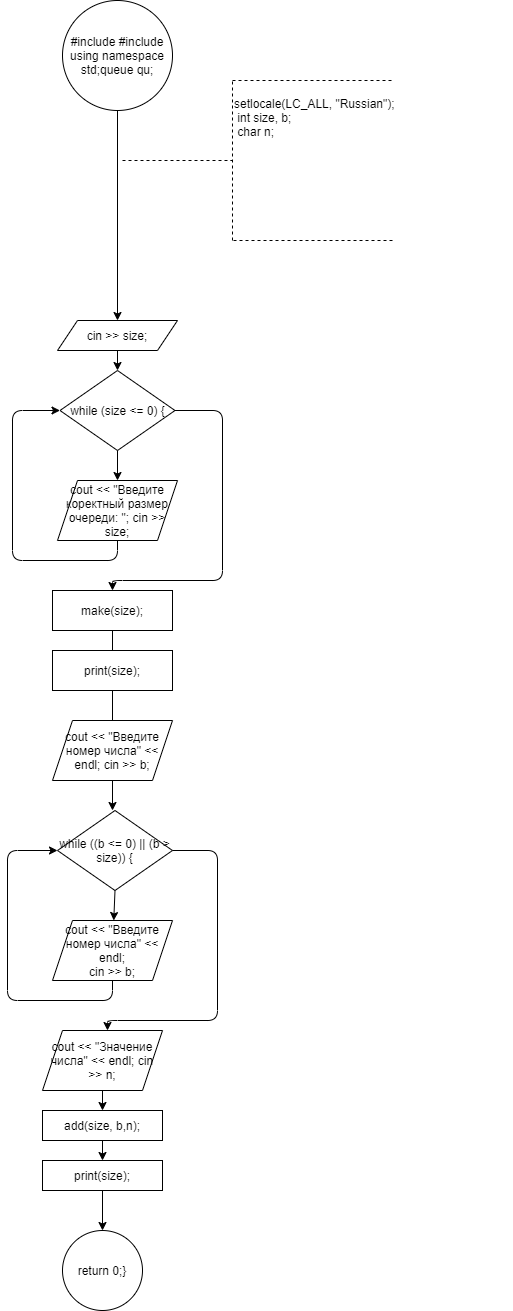


**Блок-схема программы**









**Решение**

#include <iostream>

#include <queue>

using namespace std;

queue <char> qu;

struct que {

char info;

que\* next;

};

void make(int size) {

cout << "Введите " << size << " элемента: ";

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

char a;

cin >> a;

qu.push(a);

}

}

void print(int size) {

int p = qu.size();

cout << "Элементы очереди: ";

queue <char> qu2;

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

cout << qu.front() << " ";

qu2.push(qu.front());

qu.pop();

}

int p2 = qu2.size();

for (int i = 0; i < p2; i++) {

if (!qu2.empty()) {

qu.push(qu2.front());

qu2.pop();

}

}

cout << endl;

}

void add(int size,int b,char n) {

int p = qu.size();

queue <char> qu2;

for (int i = 0; i < (p+1); i++) {

if (i == (b-1)) {

qu2.push(n);

}

else {

qu2.push(qu.front());

qu.pop();

}

}

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

if (!qu2.empty()) {

qu.push(qu2.front());

qu2.pop();

}

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

int size, b;

char n;

cin >> size;

while (size <= 0) {

cout << "Введите коректный размер очереди: ";

cin >> size;

}

make(size);

print(size);

cout << "Введите номер числа" << endl;

cin >> b;

while ((b <= 0) || (b > size)) {

cout << "Введите номер числа" << endl;

cin >> b;

}

cout << "Значение числа" << endl;

cin >> n;

add(size, b,n);

print(size);

return 0;

}

**Скриншоты**

