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 № 11.3**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Информационные и динамические структуры. Стеки

Вариант 21

Выполнил:

Студент группы РИС-20-1б

Ознобихин Елисей Андреевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

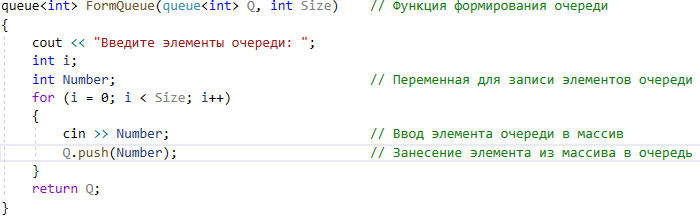
1. Получить практические навыки работы с очередями.

**Постановка задачи**

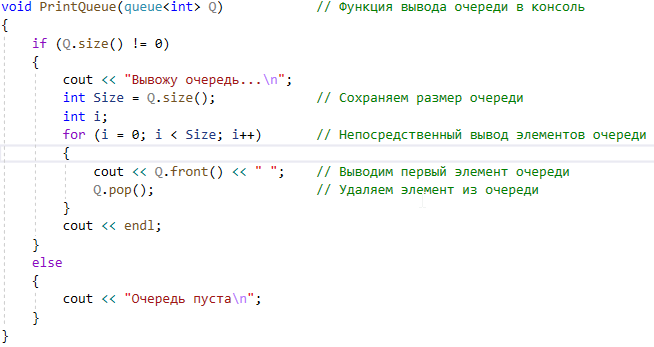
1. Тип информационного поля int. Удалить из очереди все элементы с четными информационными полями.

**STL-библиотека**

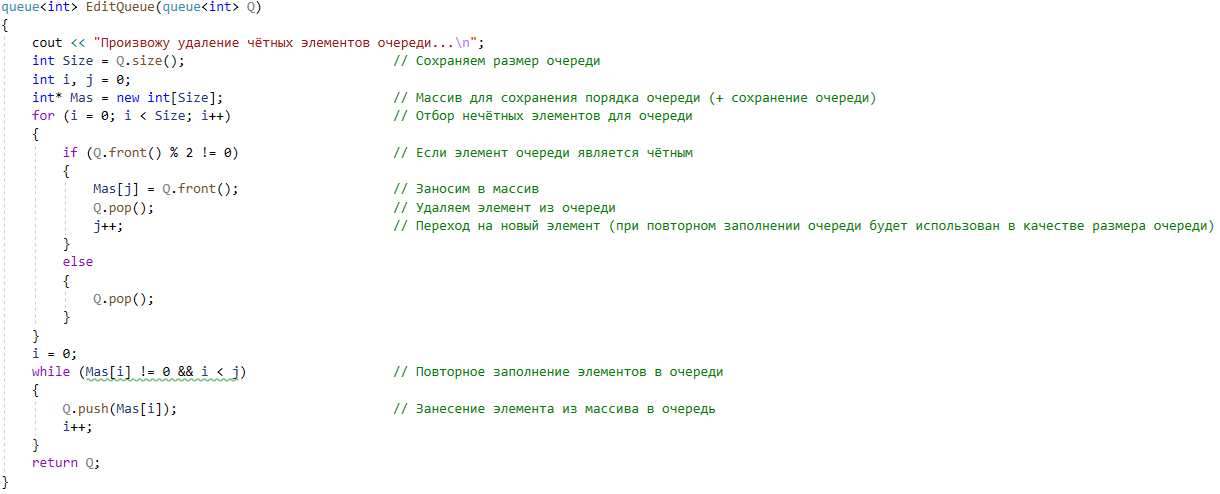
1. Для решения задачи необходимо:
   1. Разработать функцию для формирования очереди через stl-библиотеки.



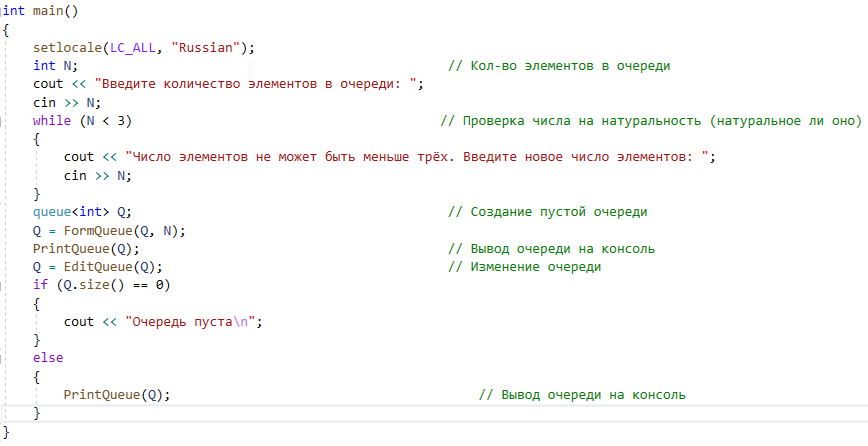
* 1. Разработать функцию для вывода информационного поля очереди на экран через консоль.



* 1. Разработать функцию для удаления элементов с чётными информационными полями из очереди.



* 1. Реализовать применение этих функций в программе.



1. В ходе работы были использованы типы данных:
   1. Для функции FormQueue использовались следующие аргументы:
   2. Тип queue<int>: очередь.
   3. Тип int: размер очереди.

Сама функция имеет тип queue<int> и возвращает список/ стек/ очередь.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_bo5dRURJbM.png

* 1. Для функции PrintQueue использовались следующие аргументы:
  2. Тип queue<int>: очередь.

Сама функция имеет тип void, поскольку при работе функции ничего не нужно возвращать.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_VCaMJPYCB7.png

* 1. Для функции EditQueue использовались следующие аргументы:
     + 1. Тип queue<int>: список/ очередь.

Сама функция имеет тип queue<int> и возвращает очередь.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_1HdD6vJJoE.png

1. Для решения задачи данные были представлены в следующем виде:
   1. Для осуществления работы данные были записаны ручным вводом с консоли в виде структур: однонаправленные и двунаправленные списки, стек и очередь.
2. Для операций ввода и вывода данных использовались следующие операторы и функции:
   1. Для вывода информации на консоль использовался оператор cout.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_MmkH9JWjLe.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_zyxq5dwRVe.png

* 1. Для ввода данных через консоль использовался оператор cin.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_MaXQm3TaYp.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_d5CIhDC2W5.png

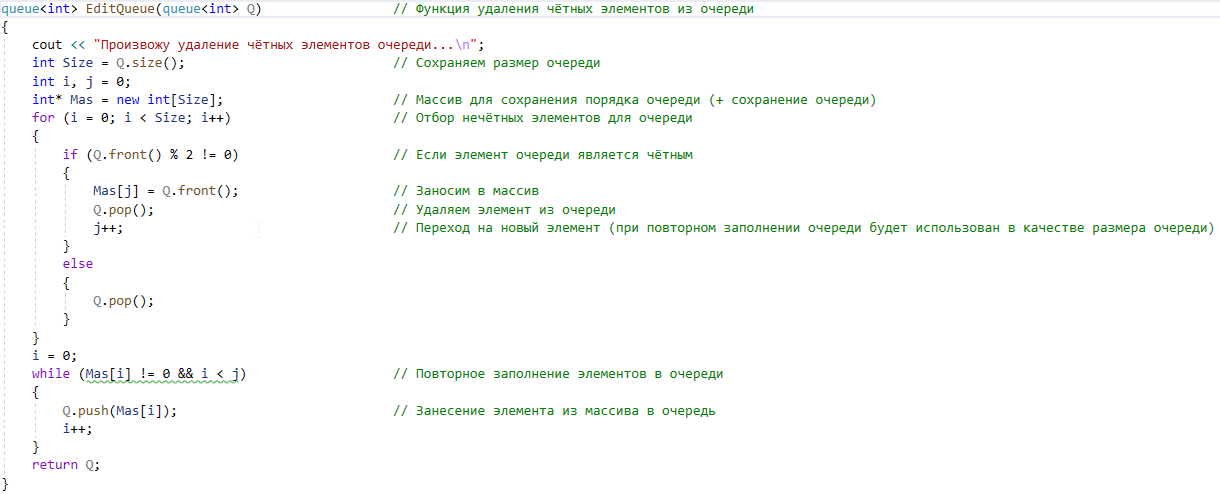
1. Поставленные задачи решены следующими действиями:
   1. Все действия со списком/ со стеком/ с очередью реализованы через написанные вручную функции.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_bo5dRURJbM.png

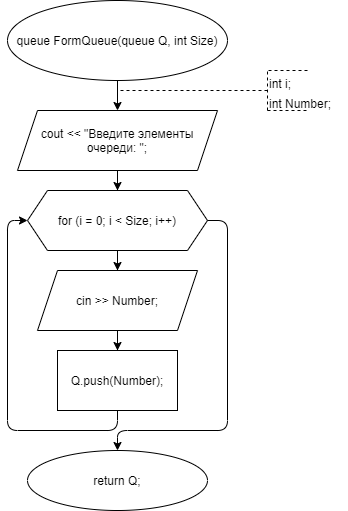
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_1HdD6vJJoE.png

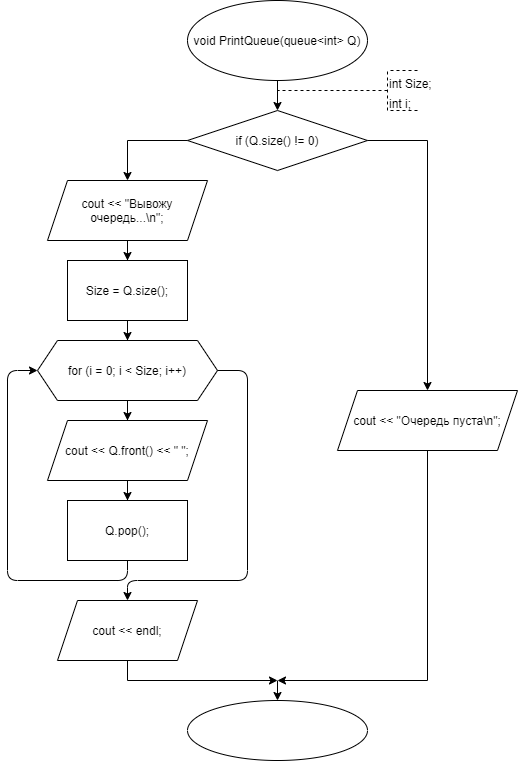
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_VCaMJPYCB7.png

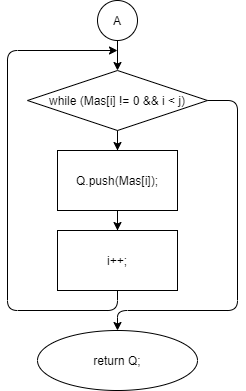
* 1. Удаление элементов из списка/ очереди реализовано с помощью цикла while и условного оператора if.

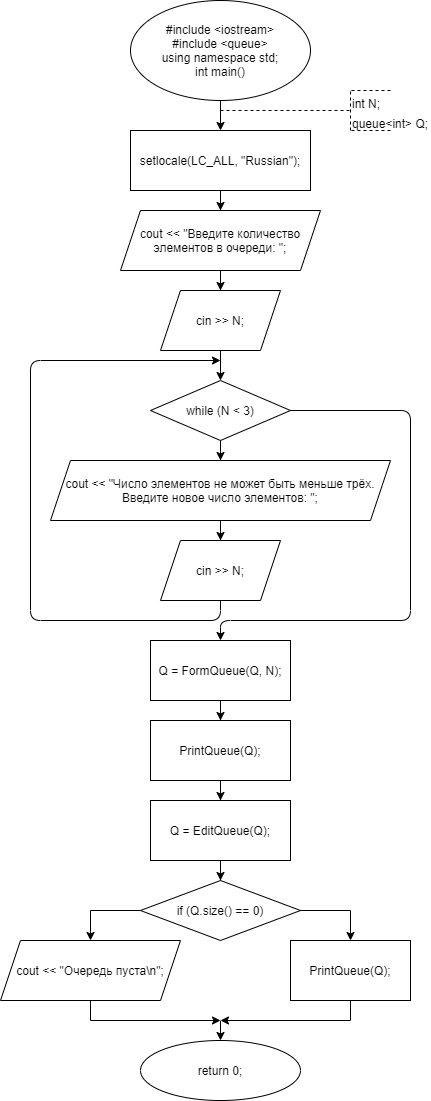


**Блок-схема программы**









**Реализация задачи на языке С++**

#include <iostream>

#include <queue>

using namespace std;

queue<int> FormQueue(queue<int> Q, int Size)

{

cout << "Введите элементы очереди: ";

int i;

int Number;

for (i = 0; i < Size; i++)

{

cin >> Number;

Q.push(Number);

}

return Q;

}

void PrintQueue(queue<int> Q)

{

if (Q.size() != 0)

{

cout << "Вывожу очередь...\n";

int Size = Q.size();

int i;

for (i = 0; i < Size; i++)

{

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

Q.pop();

}

cout << endl;

}

else

{

cout << "Очередь пуста\n";

}

}

queue<int> EditQueue(queue<int> Q)

{

cout << "Произвожу удаление чётных элементов очереди...\n";

int Size = Q.size();

int i, j = 0;

int\* Mas = new int[Size];

for (i = 0; i < Size; i++)

{

if (Q.front() % 2 != 0)

{

Mas[j] = Q.front();

Q.pop();

j++;

}

else

{

Q.pop();

}

}

i = 0;

while (Mas[i] != 0 && i < j)

{

Q.push(Mas[i]);

i++;

}

return Q;

}

int main()

{

setlocale(LC\_ALL, "Russian");

int N;

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

cin >> N;

while (N < 3)

{

cout << "Число элементов не может быть меньше трёх. Введите новое число элементов: ";

cin >> N;

}

queue<int> Q;

Q = FormQueue(Q, N);

PrintQueue(Q); Q = EditQueue(Q);

if (Q.size() == 0)

{

cout << "Очередь пуста\n";

}

else

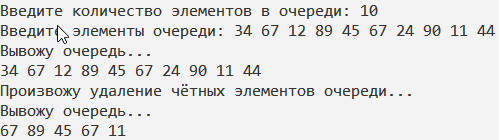
{

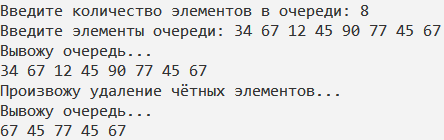
PrintQueue(Q);

}

}

**Скриншоты результатов работы программы**

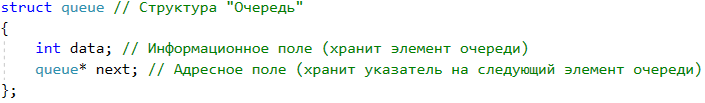
****

****

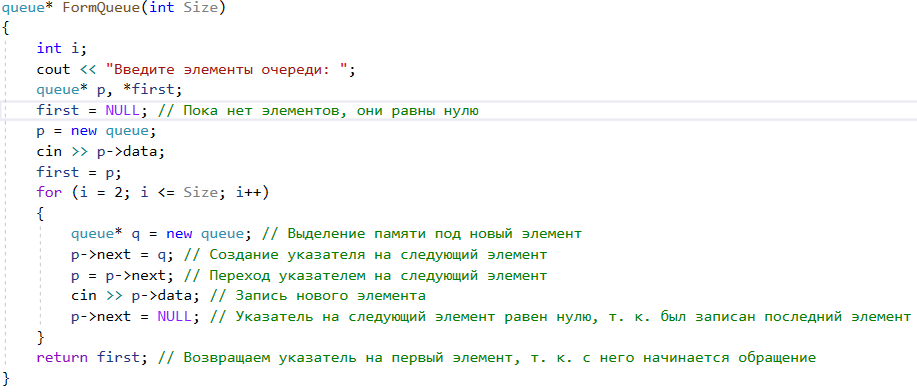
**Cтруктура**

**Анализ задачи**

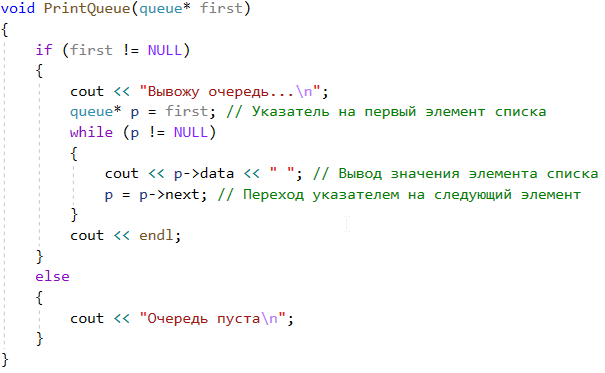
1. Для решения задачи необходимо:
   1. Зарезервировать память под структуру, содержащую информационное поле и адресное поле-указатель на следующий/предыдущий элемент.



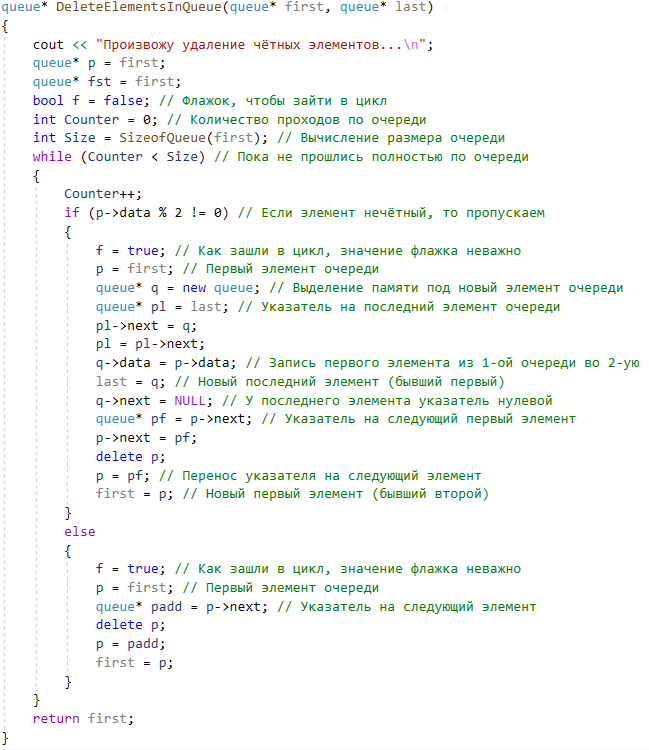
* 1. Разработать функцию для формирования списка/ списка/ стека/ очереди через структуру.



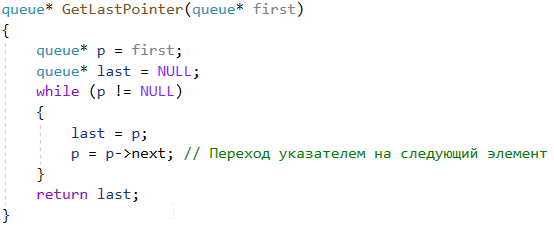
* 1. Разработать функцию для вывода информационного поля списка/ списка/ стека/ очереди на экран через консоль.



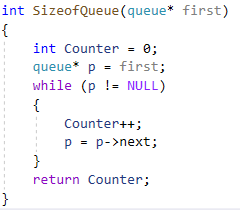
* 1. Разработать функцию для удаления элементов с чётными информационными полями из списка/ очереди.



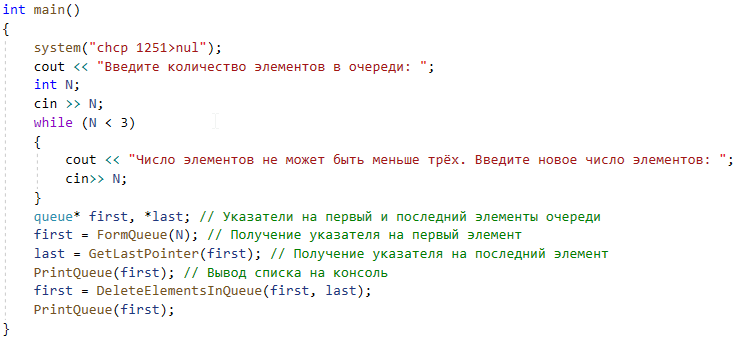
* 1. Разработать функцию для получения указателя на конец очереди.



* 1. Разработать функцию для вычисления размера очереди.



* 1. Реализовать применение этих функций в программе.



1. В ходе работы были использованы типы данных:
   1. Для функции FormQueue использовались следующие аргументы:
   2. Тип int: Размер очереди

Сама функция имеет тип указателя на queue и возвращает указатель на начало очереди.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_qUUbea2vQp.png

* 1. Для функции PrintQueue использовались следующие аргументы:
  2. Указатель queue: Указатель на первый элемент очереди.

Сама функция имеет тип void, поскольку при работе функции ничего не нужно возвращать.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_Ww8wr8hN2X.png

* 1. Для функции SizeofQueue использовались следующие аргументы:
  2. Указатель на queue: Указатель на начало очереди.

Сама функция имеет тип указателя на queue и возвращает указатель на начало очереди.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_M4QPKJlVSB.png

* 1. Для функции DeleteElementsInQueue использовались следующие аргументы:
  2. Указатель на queue: Указатель на начало списка/ очереди.
  3. Указатель на queue: Указатель на конец очереди.

Сама функция имеет тип указатель на queue и возвращает указатель на первый элемент очереди.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_vmSHdtQARa.png

* 1. Для функции GetLastPointer использовались следующие аргументы:
  2. Указатель на queue: Указатель на начало очереди.

Сама функция имеет тип указателя на queue и возвращает указатель на конец очереди.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_u5gkJinGFF.png

1. Для решения задачи данные были представлены в следующем виде:
   1. Для осуществления работы данные были записаны ручным вводом с консоли в виде структур: однонаправленные и двунаправленные списки, стек и очередь.
2. Для операций ввода и вывода данных использовались следующие операторы и функции:
   1. Для вывода информации на консоль использовался оператор cout.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_MmkH9JWjLe.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_NZuMkD0GWK.png

* 1. Для ввода данных через консоль использовался оператор cin.



1. Поставленные задачи решены следующими действиями:
   1. Все действия с очередью реализованы через написанные вручную функции.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_vmSHdtQARa.png

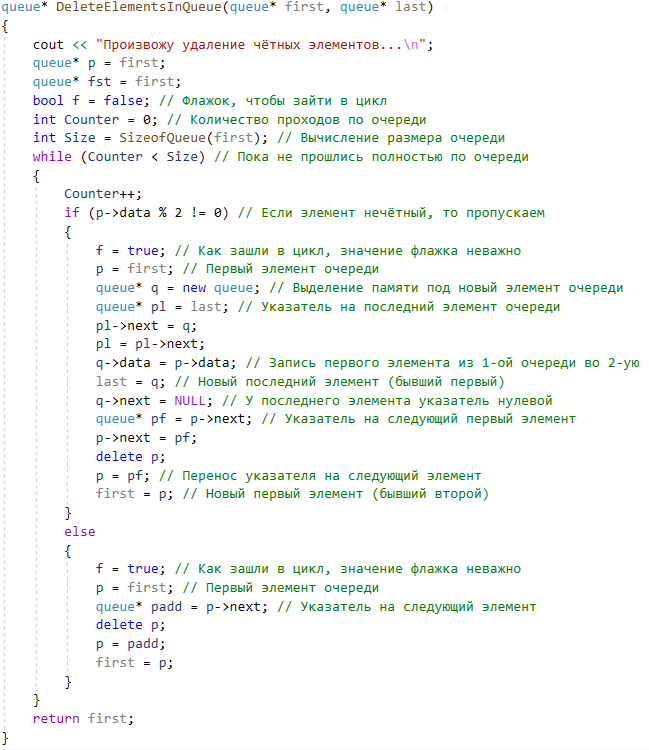
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_u5gkJinGFF.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_M4QPKJlVSB.png

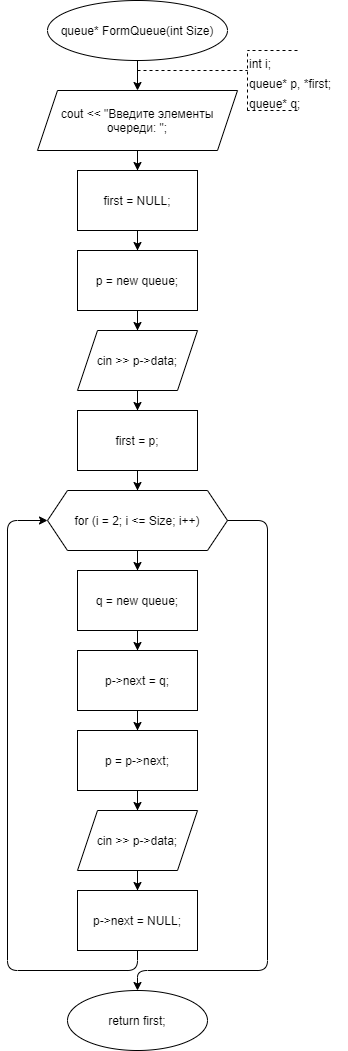
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_Ww8wr8hN2X.png

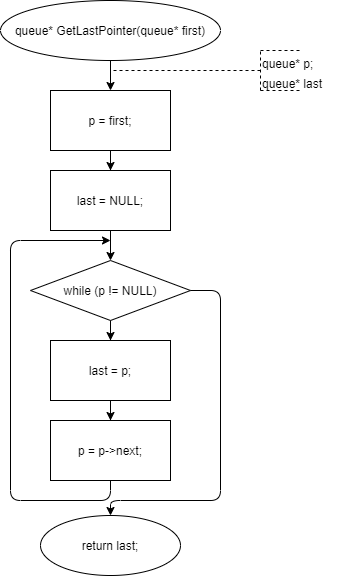
C:\Users\Асус\Documents\ShareX\Screenshots\2021-02\devenv_qUUbea2vQp.png

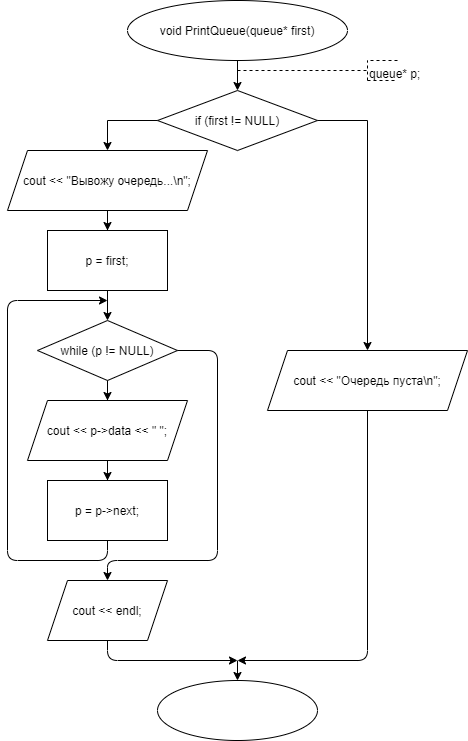
* 1. Удаление элементов из очереди реализовано с помощью цикла while. Первый элемент в очереди проверяется условным оператором на делимость на 2, и в случае истины удаляет элемент из очереди. В противном случае элемент переносится в конец очереди и проверяется следующий за ним элемент. Цикл while продолжается до тех пор, пока не прошлись по всей длине очереди.

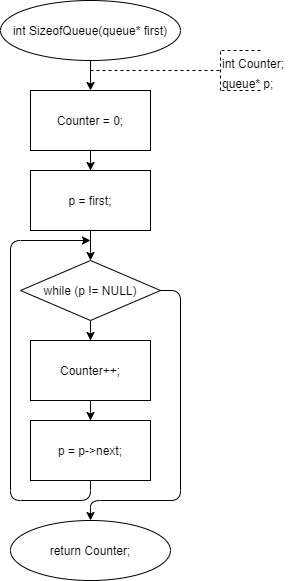


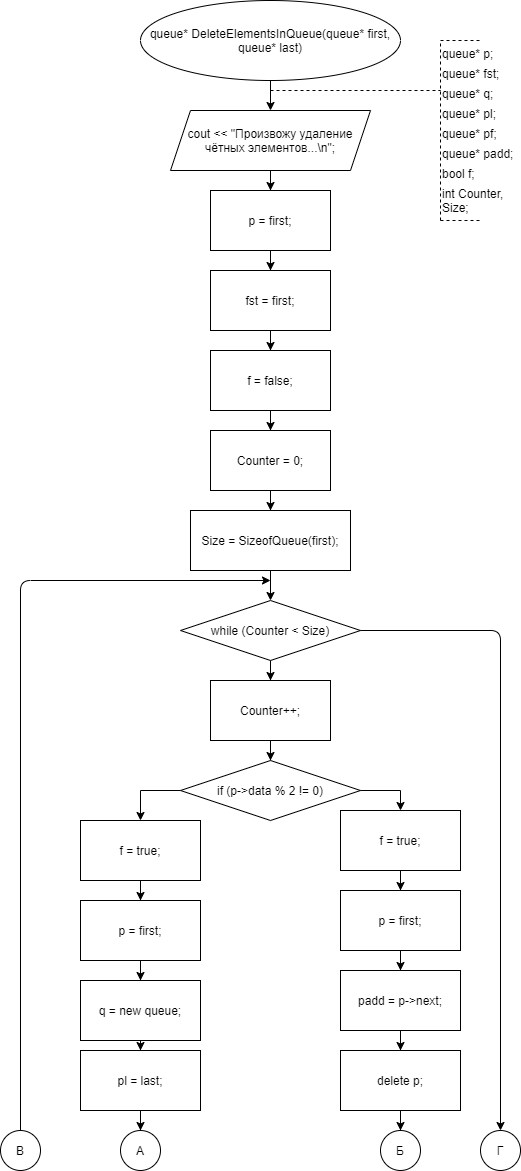
**Блок-схема программы**

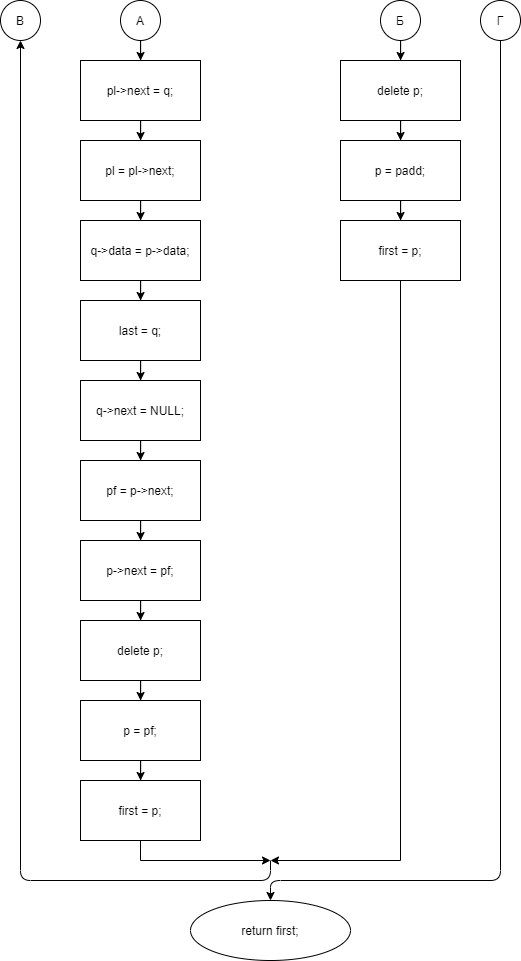


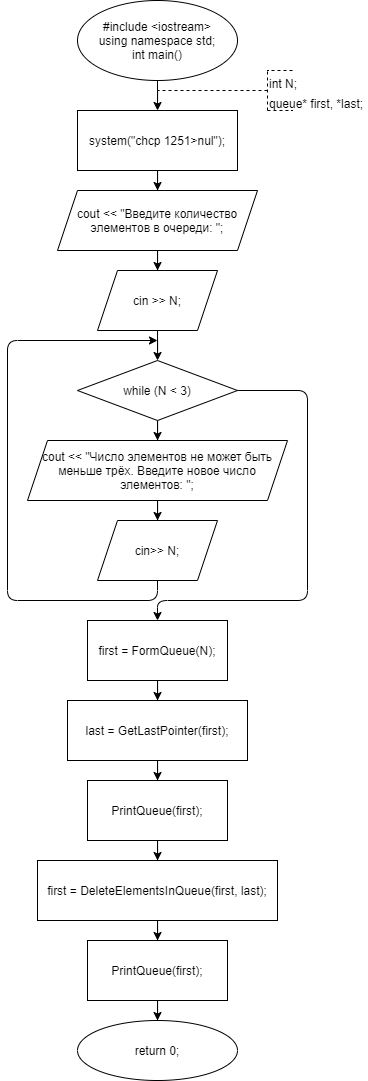












**Реализация задачи на языке С++**

#include <iostream>

using namespace std;

struct queue

{

int data;

queue\* next;

};

queue\* FormQueue(int Size)

{

int i;

cout << "Введите элементы очереди: ";

queue\* p, \*first;

first = NULL;

p = new queue;

cin >> p->data;

first = p;

for (i = 2; i <= Size; i++)

{

queue\* q = new queue;

p->next = q;

p = p->next;

cin >> p->data;

p->next = NULL;

}

return first;

}

queue\* GetLastPointer(queue\* first)

{

queue\* p = first;

queue\* last = NULL;

while (p != NULL)

{

last = p;

p = p->next;

}

return last;

}

void PrintQueue(queue\* first)

{

if (first != NULL)

{

cout << "Вывожу очередь...\n";

queue\* p = first;

while (p != NULL)

{

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

p = p->next;

}

cout << endl;

}

else

{

cout << "Очередь пуста\n";

}

}

int SizeofQueue(queue\* first)

{

int Counter = 0;

queue\* p = first;

while (p != NULL)

{

Counter++;

p = p->next;

}

return Counter;

}

queue\* DeleteElementsInQueue(queue\* first, queue\* last)

{

cout << "Произвожу удаление чётных элементов...\n";

queue\* p = first;

queue\* fst = first;

bool f = false;

int Counter = 0;

int Size = SizeofQueue(first);

while (Counter < Size)

{

Counter++;

if (p->data % 2 != 0)

{

f = true;

p = first;

queue\* q = new queue;

queue\* pl = last;

pl = pl->next;

q->data = p->data;

last = q;

q->next = NULL;

queue\* pf = p->next;

p->next = pf;

delete p;

p = pf;

first = p;

}

else

{

f = true;

p = first;

queue\* padd = p->next;

delete p;

p = padd;

first = p;

}

}

return first;

}

int main()

{

system("chcp 1251>nul");

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

int N;

cin >> N;

while (N < 3)

{

cout << "Число элементов не может быть меньше трёх. Введите новое число элементов: ";

cin>> N;

}

queue\* first, \*last;

first = FormQueue(N);

last = GetLastPointer(first);

PrintQueue(first);

first = DeleteElementsInQueue(first, last);

PrintQueue(first);

}

**Скриншоты результатов работы программы**

