Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №11.2**

Дисциплина: основы алгоритмизации и программирования

Тема: “Списки, очереди, стеки"

**Вар.21**

Выполнил работу

студент группы ИВТ-20-2Б

Галинов О.Ю.

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

**Пермь, 2021**

**Постановка задачи**

1. Сформировать однонаправленный и двунаправленный списки или стек и очередь. Тип информационного поля указан в варианте. (Тип информационного поля int.) Удалить из очереди все элементы с четными информационными полями.

2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

4. Распечатать полученный результат.

**Анализ задачи**

1. *Используемые типы данных.*

В программе используются: структура ochered, int, queue <int>.

**1.1.** Структура ochered используется для хранения одного элемента типа int.

ochered\* p = new ochered;

**1.2.** Тип int используется для хранения данных, как параметр в цикле, для временного хранения размера динамической структуры.

Int size;

**1.3.** Тип int используется для хранения данных внутри структуры.

int a;

cout << "Введите 1 элемент очереди: ";

cin >> a;

p->data = a;

* 1. Тип queue<int>используется для хранения очереди через STL.

1. *Действия над используемыми данными.*

**2.1.** С данными типа int производятся следующие действия: математические операции (инкремент в циклах с параметром).

for (int i = 2; i <= n; i++)

**2.2.** С данными типа queue производятся следующие действия: создание очереди, добавление элементов в очередь, вывод на экран, удаление первого элемента очереди.

**2.3.** Данные типа int используются только для хранения, и над ними не производятся какие-либо действия.

**2.4.** С данными типа queue<int>производятся те же действия (создание очереди, добавление элементов в очередь, вывод на экран, удаление первого элемента очереди), что и с типом ochered, но они реализованы с использованием STL.

1. *Вид данных.*

**3.1.** Очередь реализована в программе с помощью структуры и STL.

1. *Структура. //in 1*

**4.1.** В программе используется структура ochered, которая реализует очередь. Поле data имеет тип int и предназначено для хранения данных. Поле next имеет тип ochered \* и предназначено для хранения указателя на следующий элемент списка.

struct ochered

{

int data;

ochered\* next;

};

1. *Ввод и вывод.*

**5.1.** Ввод и вывод всех данных осуществляется в консоль через операторы cin и cout соответственно.

cout << "Введите 1 элемент очереди: ";

cin >> a;

p->data = a;

**5.2.** Для ввода и вывода структур реализованы функции queue<int>CreateQueueSTL(queue<int> qu), void PrintlistSTL(), ochered\* make(int n, ochered\*& first, ochered\*& last), void print(ochered\*& last). Функции с символами STLв названии предназначены для работы с очередью, реализованной через STL. Остальные предназначены для работы с очередью, реализованной через структуры.

ochered\* make(int n, ochered\*& first, ochered\*& last)

{

if (n == 0) return NULL;

ochered\* p = new ochered;

int a;

cout << "Введите 1 элемент очереди: ";

cin >> a;

p->data = a;

p->next = NULL;

first = p;

last = p;

for (int i = 2; i <= n; i++)

{

ochered\* h = new ochered;

cout << "Введите " << i << " элемент очереди: ";

cin >> a;

h->data = a;

h->next = p;

last = h;

p = last;

}

return first;

}

void print(ochered\* last)

{

ochered\* p = last;

while (p != NULL)

{

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

p = p->next;

}

cout << endl;

}

queue<int> CreateQueueSTL(int size)

{

cout << "Заполнение очереди на " << size << " элементов" << endl;

queue<int> res;

int tmp;

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

{

cout << "Введите элемент " << i + 1 << ": ";

cin >> tmp;

res.push(tmp);

}

return res;

}

void PrintQueueSTL(queue<int> qu)

{

while (!qu.empty())

{

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

qu.pop();

}

cout << endl;

}

1. *Действия для решения задачи.*

**6.1.** Удаление элементов очереди через STL.

Передвижение по очереди происходит через итератор, который увеличивается на каждой итерации цикла.

void DeleteSTL(queue<int> qu)

{

queue<int>tmp;

while (!qu.empty())

{

if (qu.front() % 2 == 0)

qu.pop();

else

{

tmp.push(qu.front());

qu.pop();

}

}

while (!tmp.empty())

{

qu.push (tmp.front());

tmp.pop();

}

cout << "Ваша очередь без четных элементов:" << endl;

while (!qu.empty())

{

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

qu.pop();

}

cout << endl;

}

**6.2.** Удаление элементов из очереди, реализованной через структуру.

Происходит в главной функции main при помощи двух других функций, созданных для добавления элемента в начало очереди (push) и удаления из начала (pop).

int pop(ochered\*&first, ochered\*& last)

{

ochered\* p = last;

int count = 0;

while (p != NULL)

{

count++;

p = p->next;

}

p = last;

if (count == 1)

{

int t = last->data;

delete p;

last = NULL;

first = NULL;

return t;

}

else {

while (p->next->next != NULL)

p = p->next;

ochered\* t = p;

int e = p->next->data;

p = p->next;

t->next = NULL;

first = t;

delete p;

return e;

}

}

ochered\* push(int n, ochered\*& last)

{

ochered\* p = new ochered;

p->data = n;

p->next = last;

last = p;

return last;

}

int\* mas = new int[size];

int i = 0;

while (i < size)

{

int t = pop(first, last);

if (t % 2 != 0)

{

mas[i] = t;

i++;

}

else {

size--;

}

}

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

{

push(mas[i], last);

}

**Код программы**

#include <iostream>

#include<queue>

#include<vector>

using namespace std;

//реализация через структуру:

struct ochered

{

int data;

ochered\* next;

};

ochered\* make(int n, ochered\*& first, ochered\*& last)

{

if (n == 0) return NULL;

ochered\* p = new ochered;

int a;

cout << "Введите 1 элемент очереди: ";

cin >> a;

p->data = a;

p->next = NULL;

first = p;

last = p;

for (int i = 2; i <= n; i++)

{

ochered\* h = new ochered;

cout << "Введите " << i << " элемент очереди: ";

cin >> a;

h->data = a;

h->next = p;

last = h;

p = last;

}

return first;

}

void print(ochered\* last)

{

ochered\* p = last;

while (p != NULL)

{

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

p = p->next;

}

cout << endl;

}

int pop(ochered\*&first, ochered\*& last)

{

ochered\* p = last;

int count = 0;

while (p != NULL)

{

count++;

p = p->next;

}

p = last;

if (count == 1)

{

int t = last->data;

delete p;

last = NULL;

first = NULL;

return t;

}

else {

while (p->next->next != NULL)

p = p->next;

ochered\* t = p;

int e = p->next->data;

p = p->next;

t->next = NULL;

first = t;

delete p;

return e;

}

}

ochered\* push(int n, ochered\*& last)

{

ochered\* p = new ochered;

p->data = n;

p->next = last;

last = p;

return last;

}

//other

int getsize()

{

int size;

cin >> size;

while (size < 1)

{

cout << "Введите размер корректно: ";

cin >> size;

}

return size;

}

//STL-libraries

queue<int> CreateQueueSTL(int size)

{

cout << "Заполнение очереди на " << size << " элементов" << endl;

queue<int> res;

int tmp;

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

{

cout << "Введите элемент " << i + 1 << ": ";

cin >> tmp;

res.push(tmp);

}

return res;

}

void PrintQueueSTL(queue<int> qu)

{

while (!qu.empty())

{

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

qu.pop();

}

cout << endl;

}

void DeleteSTL(queue<int> qu)

{

queue<int>tmp;

while (!qu.empty())

{

if (qu.front() % 2 == 0)

qu.pop();

else

{

tmp.push(qu.front());

qu.pop();

}

}

while (!tmp.empty())

{

qu.push (tmp.front());

tmp.pop();

}

cout << "Ваша очередь без четных элементов:" << endl;

while (!qu.empty())

{

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

qu.pop();

}

cout << endl;

}

int main()

{

//Structttttttt

setlocale(LC\_ALL, "rus");

int size;

cout << "Введите размер очереди: ";

size = getsize();

ochered\* first, \* last;

make(size, first, last);

cout << "Ваша очередь:" << endl;

print(last);

int\* mas = new int[size];

int i = 0;

while (i < size)

{

int t = pop(first, last);

if (t % 2 != 0)

{

mas[i] = t;

i++;

}

else {

size--;

}

}

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

{

push(mas[i], last);

}

cout << "Ваша очередь без элементов с четным информационным полем:" << endl;

print(last);

//STL-libraries

int sizeofSTL;

cout << "Введите размер очереди, реализованной через STL - библиотеку: ";

sizeofSTL = getsize();

queue<int> myocher = CreateQueueSTL(sizeofSTL);

cout << "Ваша очередь:" << endl;

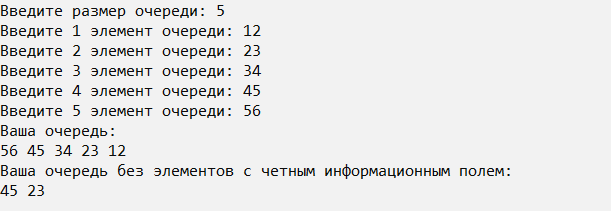
PrintQueueSTL(myocher);

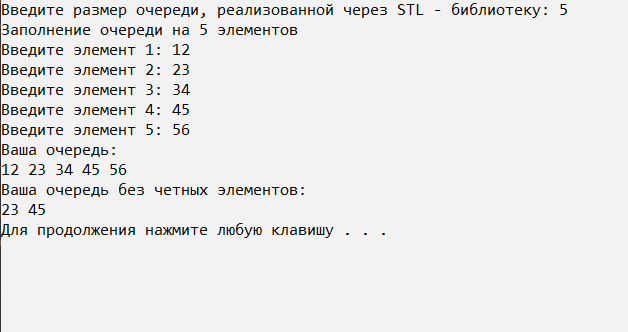
DeleteSTL(myocher);

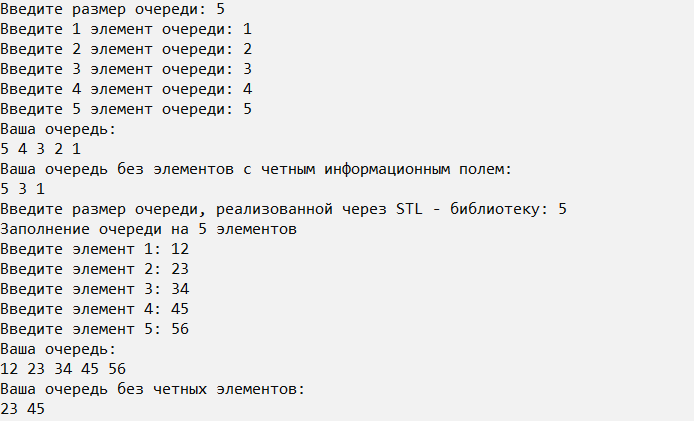
system("pause");

}

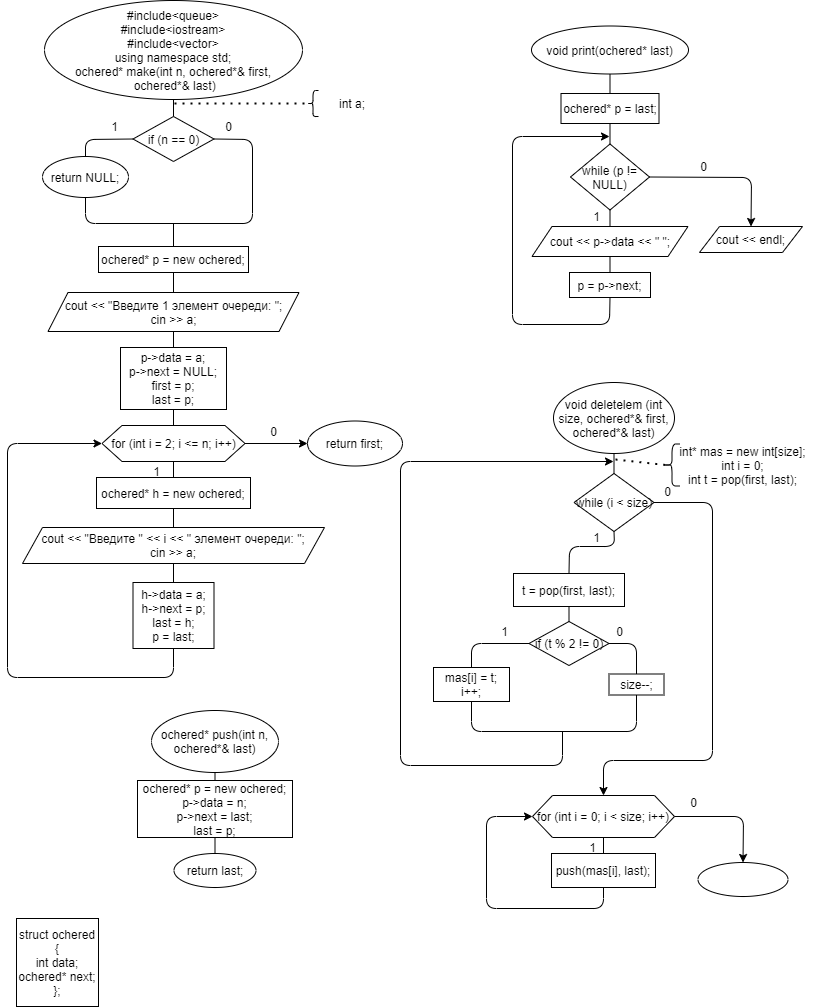
**Скриншоты**

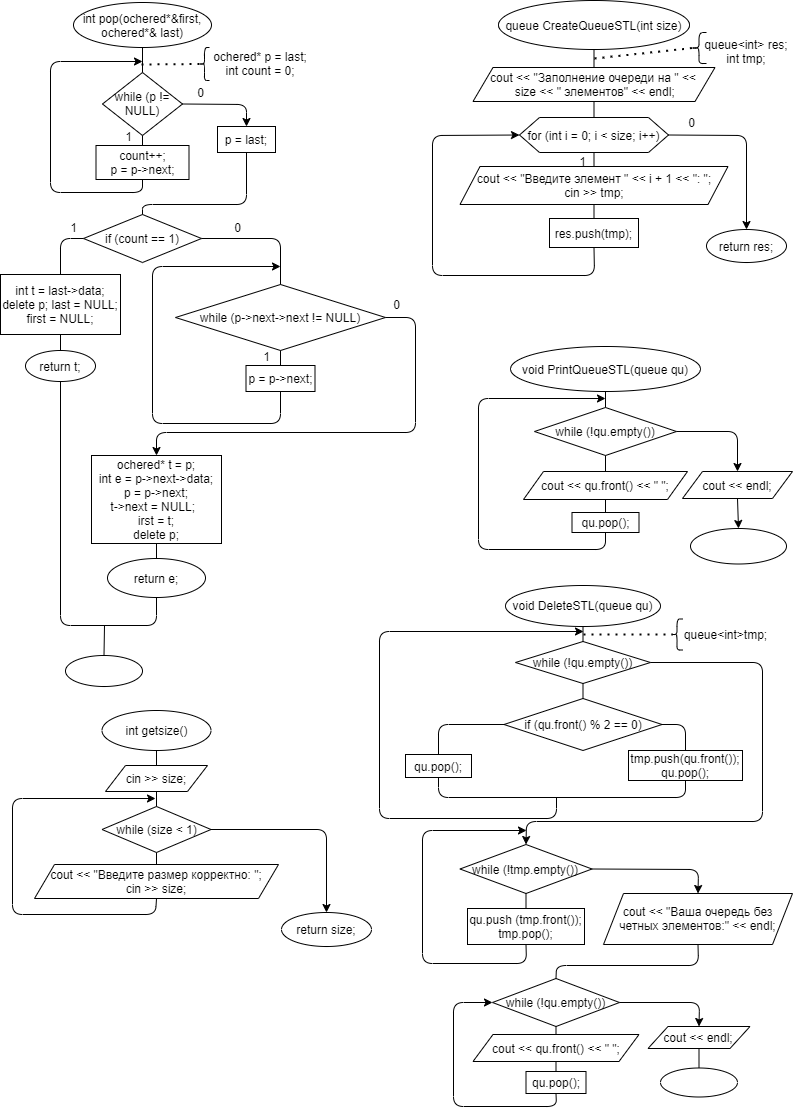
****

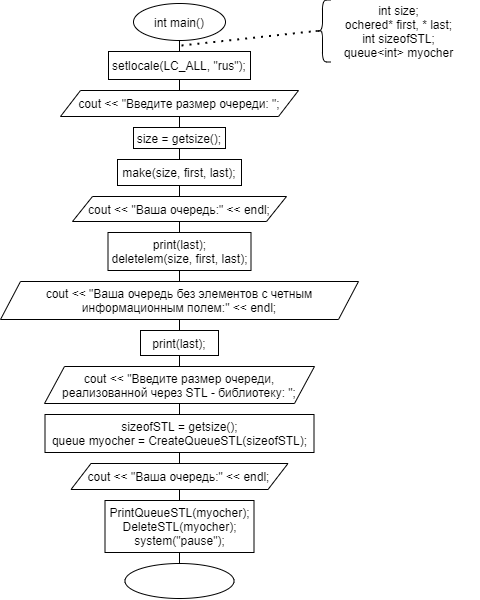
****

****

**Блок-схема**

****

****

****