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 №11.4**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Динамические структуры данных. Стеки

Вариант 21

Выполнил:

Студент группы ИВТ-20-1б

Углицких Максим Сергеевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь 2021**

**Цель работы:** получить практические навыки работы с очередями.

**Постановка задачи:**

* Сформировать стеки с помощью структуры.
* Распечатать полученную структуру.
* Выполнить обработку структуры в соответствии с заданием.
* Распечатать полученный результат.
* Удалить соответствующую структуру из памяти.

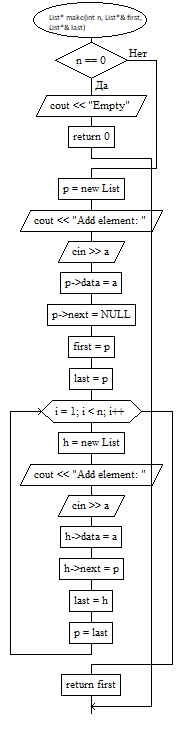
**Вариант:** 21

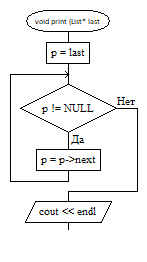
Тип информационного поля int. Удалить из очереди все элементы с четными информационным и полями.

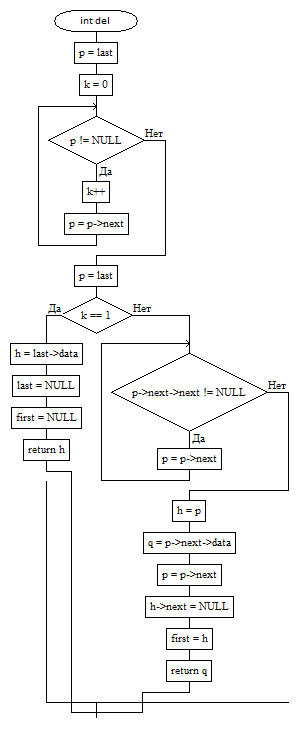
**Анализ задачи:**

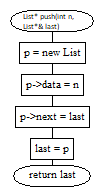
1. Создаем структуру с информационным полем int. После чего создаем указатель на следующий элемент List\* next
2. Создаем функцию List\* make (int n, List\*& first, List\*& last) для стека. После чего заполняем его с клавиатуры. Вводим сначала первый элемент, динамически выделяя поде него память. Затем ставим указатель верхний элемент стека.
3. После заполняем оставшиеся элементы стека, попутно возвращая указатель на верхний элемент.
4. Создаем функцию void print для вывода элементов, с проверкой на возможность вывода ( while (p != NULL)).
5. Удаление нечетных элементов будет происходить в теле int main() с помощью функции int del. Если указатель на элемент указывает на четный элемент, то функция int del сдвигает все элементы стека так: сохраняем значение последнего элемента, а второй элемент делаем первым, после чего удаляем последний и возвращаем первый элемент.

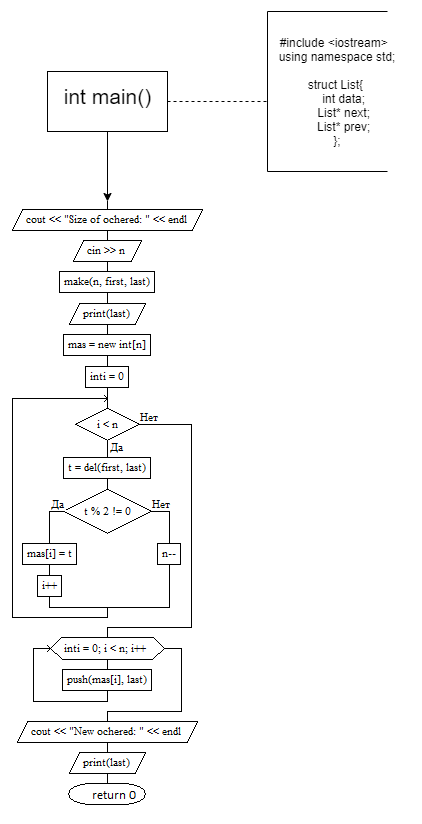
**Блок-схема:**

****

****

****

****



**Код программы:**

#include <iostream>

using namespace std;

int key;

struct List{

    int data;

    List\* next;

};

List\* make(int n, List\*& first, List\*& last){

    if (n==0)

    {

        cout << "Empty";

        return 0;

    }

    List\* p = new List;

    cout << "Add element: ";

    int a;

    cin >> a;

    p->data = a;

    p->next = NULL;

    first = p;

    last = p;

    for (int i = 1; i < n; i++)

    {

        List\* h = new List;

        cout << "Add element: ";

        cin >> a;

        h->data = a;

        h->next = p;

        last = h;

        p = last;

    }

    return first;

}

void print(List\* last){

    List\* p = last;

    while (p != NULL)

    {

        cout << p->data << " ";

        p=p->next;

    }

    cout << endl;

}

int del(List\*& first, List\*& last){

    List\* p = last;

    int k = 0;

    while (p!=NULL)

    {

        k++;

        p=p->next;

    }

    p=last;

    if (k==1)

    {

        int h = last->data;

        delete p;

        last = NULL;

        first = NULL;

        return h;

    }

    else

    {

        while (p->next->next != NULL)

        {

            p=p->next;

        }

        List\* h = p;

        int q = p->next->data;

        p=p->next;

        h->next =NULL;

        first = h;

        delete p;

        return q;

    }

}

List\* push(int n, List\*& last)

{

    List\*p = new List;

    p->data = n;

    p->next = last;

    last = p;

    return last;

}

int main(){

    cout << "Size of ochered: " << endl;

    int n;

    cin >> n;

    List\* first, \*last;

    make(n, first, last);

    print(last);

    int \*mas = new int[n];

    int i=0;

    while (i<n)

    {

        int t = del(first, last);

        if (t %2 != 0)

        {

            mas[i] = t;

            i++;

        }

        else{

            n--;

        }

    }

    for (int i=0; i<n; i++)

    {

        push(mas[i], last);

    }

    cout << "New ochered: " << endl;

    print(last);

}

**Результат работы программы:**

