Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

направление подготовки: 09.03.01 - «Информатика и вычислительная техника»

**Отчет по Лабораторной работе Стеки**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

Выполнил студент гр. ИВТ-21-1б

Ахунов Руслан Булатович

Проверил:

Ст. Преподаватель кафедры ИТАС

Яруллин Денис Владимирович

(оценка) (подпись)

г. Пермь - 2022

**Постановка задачи:**

Вариант 2

 Тип информационного поля double. Удалить из стека все элементы с четными номерами (2, 4, 6 и. т. д.).

**Анализ задачи:**

template <typename T>

struct Node

{

    T data;

    Node<T>\* next;

};

template <typename T>

struct Stack

{

    Node<T>\* head = nullptr;

    int size = 0;

};

Структура Нода, поля для значения и указатель на следующую Ноду.

Структура Стек, есть поля размера, а также указатель типа Нода на «голову».

template <typename T>

void Push(Stack<T>& tmp, const T& data)

{

    Node<T>\* newnode = new Node<T>;

    newnode->next = tmp.head;

    newnode->data = data;

    tmp.head = newnode;

    tmp.size++;

}

Функция добавления элемента, создаем указатель на новую Ноду, указатель на следующий элемент равен «голове» текущего стека, но эту «голову» мы переприсваиваем новой ноде, также увеличиваем размер стека.

template <typename T>

void pop(Stack<T>& tmp)

{

    if (tmp.head != nullptr)

    {

        Node<T>\* deleted = tmp.head;

        tmp.head = tmp.head->next;

        delete deleted;

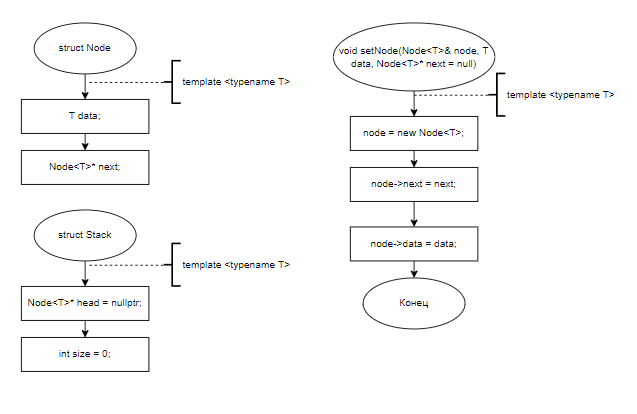
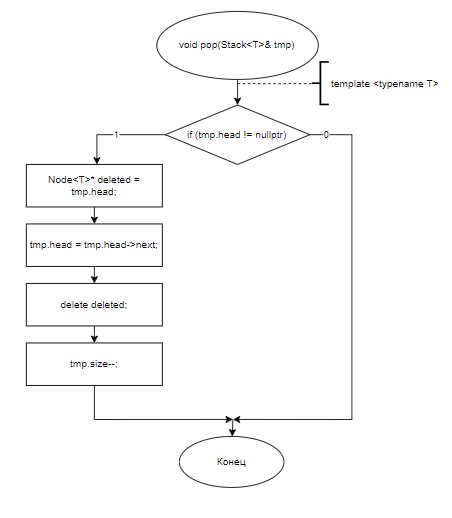
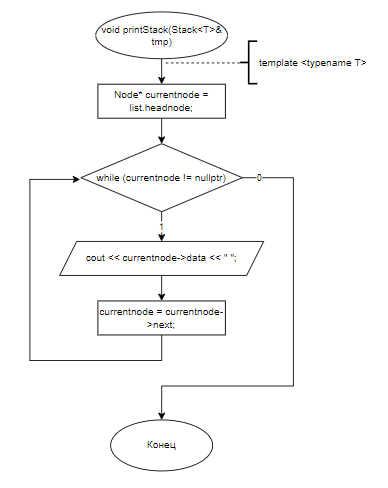
        tmp.size--;

    }

}

Удаление элемента, удаляем самый «верхний» элемент стека, то есть последним пришел, первым ушел.

**Блок-схема**

**** ****  

**Код программы**

#include <iostream>

using namespace std;

template <typename T>

struct Node

{

    T data;

    Node<T>\* next;

};

template <typename T>

struct Stack

{

    Node<T>\* head = nullptr;

    int size = 0;

};

template <typename T>

void setNode(Node<T>& node, T data, Node<T>\* next = null)

{

    node = new Node<T>;

    node->next = next;

    node->data = data;

}

template <typename T>

void Push(Stack<T>& tmp, const T& data)

{

    Node<T>\* newnode = new Node<T>;

    newnode->next = tmp.head;

    newnode->data = data;

    tmp.head = newnode;

    tmp.size++;

}

template <typename T>

void pop(Stack<T>& tmp)

{

    if (tmp.head != nullptr)

    {

        Node<T>\* deleted = tmp.head;

        tmp.head = tmp.head->next;

        delete deleted;

        tmp.size--;

    }

}

template <typename T>

void printStack(Stack<T>& tmp)

{

    Node<T>\* currentnode = tmp.head;

    while (currentnode != nullptr)

    {

        cout << currentnode->data << " ";

        currentnode = currentnode->next;

    }

}

int main()

{

    setlocale(LC\_ALL, "Russian");

    int n;

    int count = 1;

    cout << "Введите размер стака: ";

    cin >> n;

    double k;

    Stack <double> st;

    Stack <double> st2;

    for (int i = 0; i < n; i++)

    {

        cin >> k;

        Push(st, k);

    }

    cout << endl;

    printStack(st);

    cout << endl;

    for (int i = 0; i < n; i++)

    {

        if (count % 2 == 0)

        {

            Push(st2, st.head->data);

        }

        pop(st);

        count++;

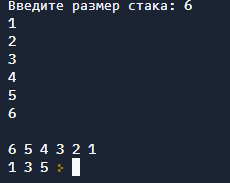
    }

    printStack(st2);

    return 0;

}

**Вывод программы**

****