Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное учреждение высшего образования  
«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет  
Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т  
по лабораторной работе**

Дисциплина: «Основы алгоритмизации и программирования»

Выполнил  
студент группы КЗИ-21-1б  
Полковников Т.Ю.

Проверил  
доцент кафедры ИТАС  
ст. преп. Яруллин Д.В.

Пермь, 2022

## Постановка задачи:

1. Сформировать однонаправленный и двунаправленный списки или стек и очередь. Тип информационного поля указан в варианте.

2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

4. Распечатать полученный результат.

5. Удалить соответствующую структуру из памяти.

## Анализ решения задачи:

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

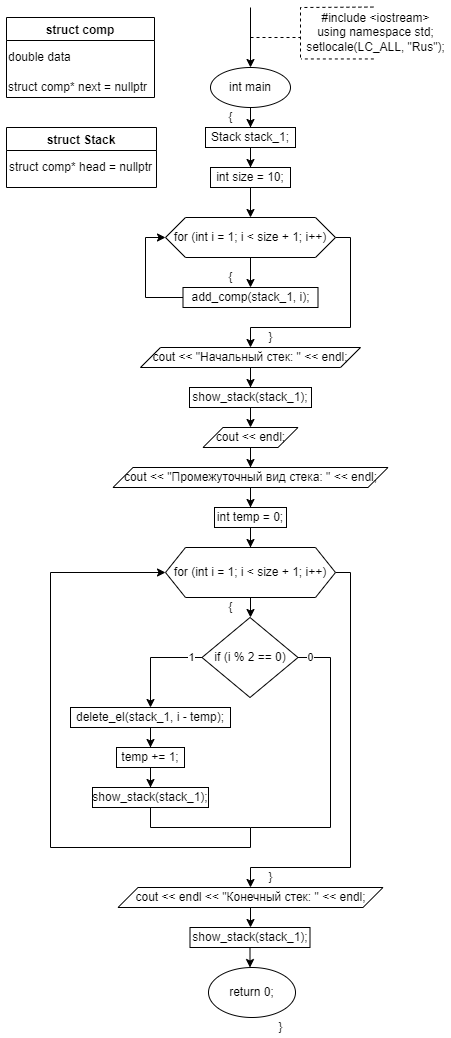
## Описание переменных:

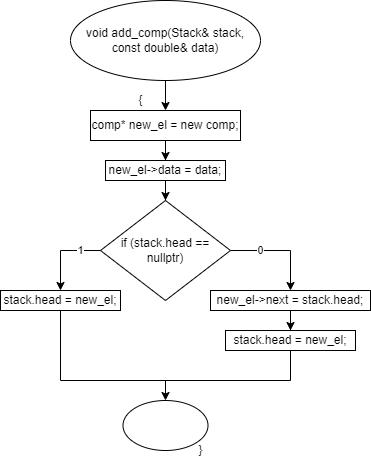
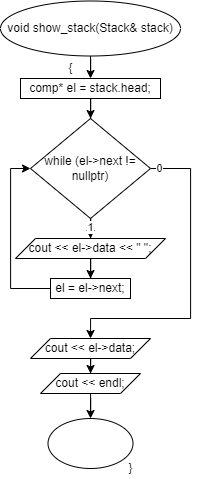
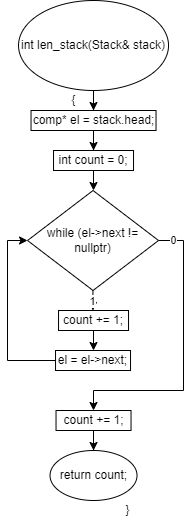
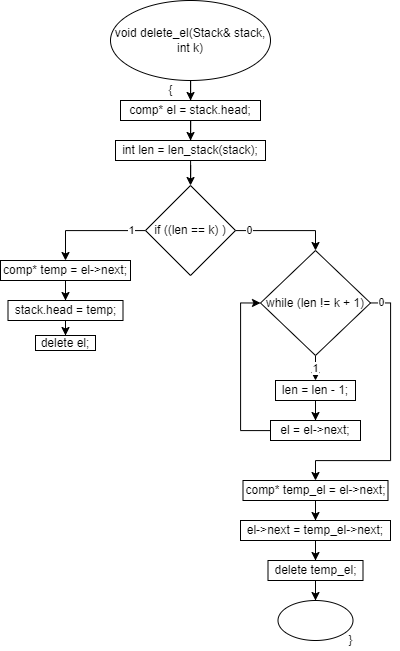
stack\_1 – стек

size – длинна стека

temp – временная переменная

**Блок схема**



## Код программы

## include <iostream>

using namespace std;

// компонент стека

struct comp {

double data;

struct comp\* next = nullptr;

};

//стек

struct Stack {

struct comp\* head = nullptr;

};

// добавление в стек элемента

void add\_comp(Stack& stack, const double& data) {

comp\* new\_el = new comp;

new\_el->data = data;

if (stack.head == nullptr) {

stack.head = new\_el;

}

else {

new\_el->next = stack.head;

stack.head = new\_el;

}

}

// вывод стека

void show\_stack(Stack& stack) {

comp\* el = stack.head;

while (el->next != nullptr) {

cout << el->data << " ";

el = el->next;

}

cout << el->data;

cout << endl;

}

// длинна стека

int len\_stack(Stack& stack) {

comp\* el = stack.head;

int count = 0;

while (el->next != nullptr) {

count += 1;

el = el->next;

}

count += 1;

return count;

}

// удаление элемента из стека

void delete\_el(Stack& stack, int k) {

comp\* el = stack.head;

int len = len\_stack(stack);

if ((len == k) ) {

comp\* temp = el->next;

stack.head = temp;

delete el;

}

else {

while (len != k + 1) {

len = len - 1;

el = el->next;

}

comp\* temp\_el = el->next;

el->next = temp\_el->next;

delete temp\_el;

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

// создаём стек

Stack stack\_1;

// размер стека

int size = 10;

// заполняем поле data компонентов стека числами от 1 до size

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

add\_comp(stack\_1, i);

}

cout << "Начальный стек: " << endl;

show\_stack(stack\_1);

cout << endl;

cout << "Промежуточный вид стека: " << endl;

int temp = 0;

// удаление элементов с чётными номерами

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

if (i % 2 == 0) {

delete\_el(stack\_1, i - temp);

temp += 1;

show\_stack(stack\_1);

}

}

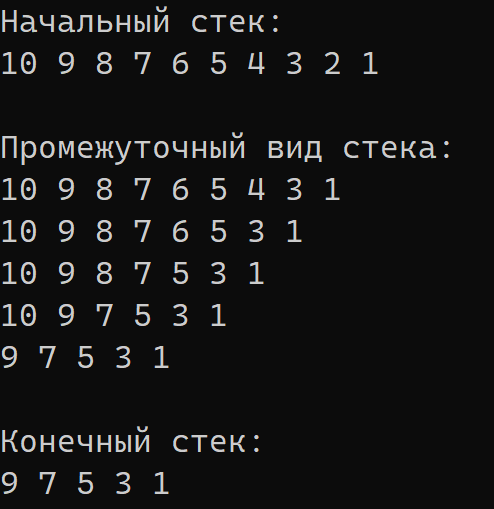
cout << endl << "Конечный стек: " << endl;

show\_stack(stack\_1);

return 0;

}

Скриншоты результатов:



Анализ результатов:

Программа работает правильно. Выводиться начальный стек, промежуточные результаты стека и конченый отредактированный стек.