Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**о работе по информатике**

Семестр: 2

На тему: «Стек»

Выполнил студент ИВТ-22-2б:

Афонин Артем Александрович

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

Пермь 2023

**Постановка задачи**

Записи в линейном списке содержат ключевое поле типа string (строка символов). Сформировать стек. Удалить элемент с заданным ключом. Добавить К элементов перед элементом с заданным номером.

**Алгоритм решения**

1. Определить узел стека через структуру struct Node. Узел содержит ключевое слово типа string и указатель на следующий узел.

2. Определить структуру стека struct Stack. Структура указатель на вершину стека и конструктор для указателя на вершину.

3. Реализовать функцию void push() для добавления элемента в стек. В функции создаётся и заполняется узел стека, а также обновляется вершина стека.

4. Реализовать функцию для удаления элемента из стека void pop(). В функции проводится проверка на пустоту стека, удаляется узел стека и обновляется его вершина.

5. Реализовать функцию void printStack(), которая выводит элементы стека.

6. Реализовать функцию void copy\_stack1(), с помощью которой происходит копирование элемента из стека в промежуточный стек для выполнения первой части задания.

7. Реализовать функцию void copy\_stack2(), с помощью которой происходит копирование элемента из стека в промежуточный стек для выполнения второй части задания.

8. Реализовать функцию void delete\_key(), с помощью которой происходи удаление заданного ключа.

9. Реализовать функцию void delete\_elem() для удаления элемента перед элементом с заданным номером.

10. В главной функции int main() реализовать вызов всех внешних функций согласно постановке задачи.

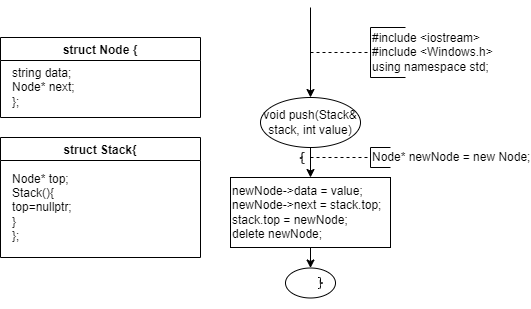
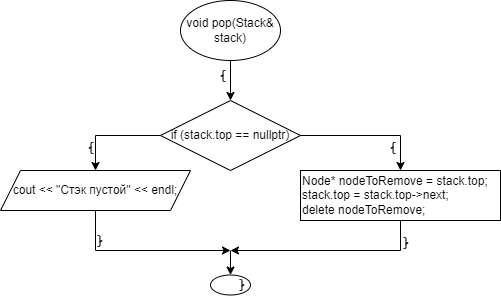
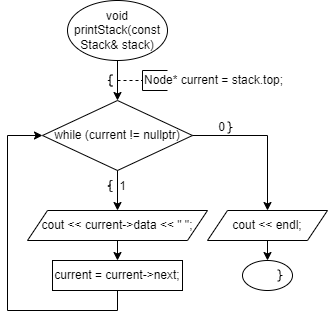
**Блок-схемы**

Рисунок 1 – Функция void push() и структуры struct Node и struct Stack

Рисунок 2 – Функция void pop()

Рисунок 3 – Функция void printStack()

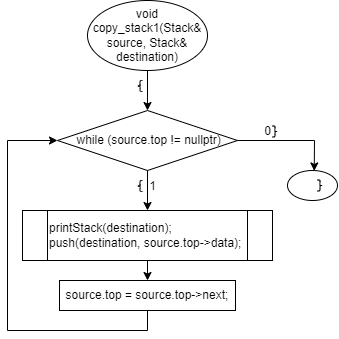
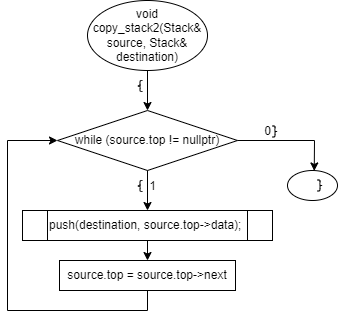
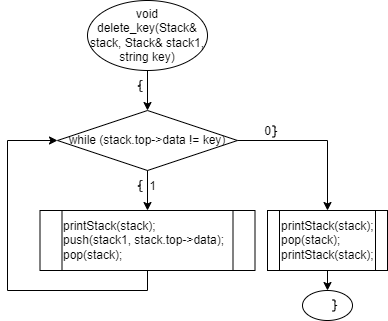
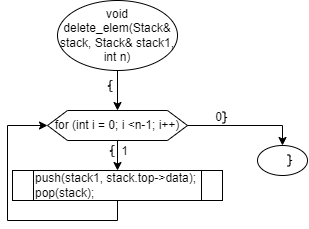


Рисунок 4 – Функция void copy\_stack1

Рисунок 5 – Функция void copy\_stack2

Рисунок 6 – Функция void delete\_key()

Рисунок 7 – Функция void delete\_elem()

**Код программы**

#include <iostream>

#include <string>

#include <Windows.h>

using namespace std;

// Определение узла стека

struct Node {

string data; // Данные одного узла

Node\* next; // Указатель на следующий узел

};

// Определение структуры стека

struct Stack {

Node\* top; // Указатель на вершину стека

Stack() {

top = nullptr;

}

};

// Функция добавления элемента в стек

void push(Stack& stack, string value) {

Node\* newNode = new Node; // Создание нового узла стека

newNode->data = value;

newNode->next = stack.top;

stack.top = newNode; // Обновление вершины стека

delete newNode;

}

// Функция удаления элемента из стека

void pop(Stack& stack) {

if (stack.top == nullptr) {

cout << "Стэк пустой" << endl; // Стек пуст, невозможно удалить элемент

}

else {

Node\* nodeToRemove = stack.top; // Сохранение указателя на удаляемый узел

stack.top = stack.top->next; // Обновление вершины стека

delete nodeToRemove; // Удаление узла стека

}

}

// Функция вывода содержимого стека

void printStack(const Stack& stack) {

Node\* current = stack.top;

while (current != nullptr) {

cout << current->data << " ";

current = current->next;

}

cout << endl;

}

//Функция для копирования элементов в промежуточный стек для 1-го задания

void copy\_stack1(Stack& source, Stack& destination) {

while (source.top != nullptr) {

printStack(destination);

push(destination, source.top->data); // Копирование значени элемента в новый стек

source.top = source.top->next; // Переход к следующему элементу в старом стеке

}

}

//Функция для копирования элементов в промежуточный стек для 2-го задания

void copy\_stack2(Stack& source, Stack& destination) {

while (source.top != nullptr) {

push(destination, source.top->data); // Копирование значени элемента в новый стек

source.top = source.top->next; // Переход к следующему элементу в старом стеке

}

}

//Функция для удаления ключа

void delete\_key(Stack& stack, Stack& stack1, string key) {

while (stack.top->data != key) {

printStack(stack);

push(stack1, stack.top->data);

pop(stack);

}

printStack(stack);

pop(stack);

printStack(stack);

}

//Функция для удаления элемента перед элементом с заданным номером

void delete\_elem(Stack& stack, Stack& stack1, int n) {

for (int i = 0; i <n-1; i++) {

push(stack1, stack.top->data);

pop(stack);

}

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Stack stack, stack1, stack3;

string a, s, el;

int n, m, k;

cout << "Введите количество элементов в стеке: "; cin >> n;

for (int i = 0; i < n; i++) {

cout << "Введите элемент стека: "; cin >> a;

push(stack, a);

}

cout << endl << "Ваш стек: ";

printStack(stack);

cout << endl << "Введите ключ удаления: "; cin >> s;

delete\_key(stack, stack1, s);

cout << endl << "Промежуточный стек: ";

printStack(stack1);

cout << endl;

copy\_stack1(stack1, stack);

cout << "Стэк после удаления элемента: "; printStack(stack);

cout << endl << "Введите номер элемента, перед которым нужно вставить новые элементы стека: "; cin >> m;

delete\_elem(stack, stack3, m);

cout << "Введите количество добавляемых элементов: "; cin >> k;

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

cout << "Введите добавляемый элемент: "; cin >> el;

push(stack, el);

}

copy\_stack2(stack3, stack);

cout << "Стек после добавления элементов: ";

printStack(stack);

cout << endl;

cout << "Удаление элементов стека:" << endl;

for (int i = 0; i < n; i++) {

printStack(stack);

pop(stack);

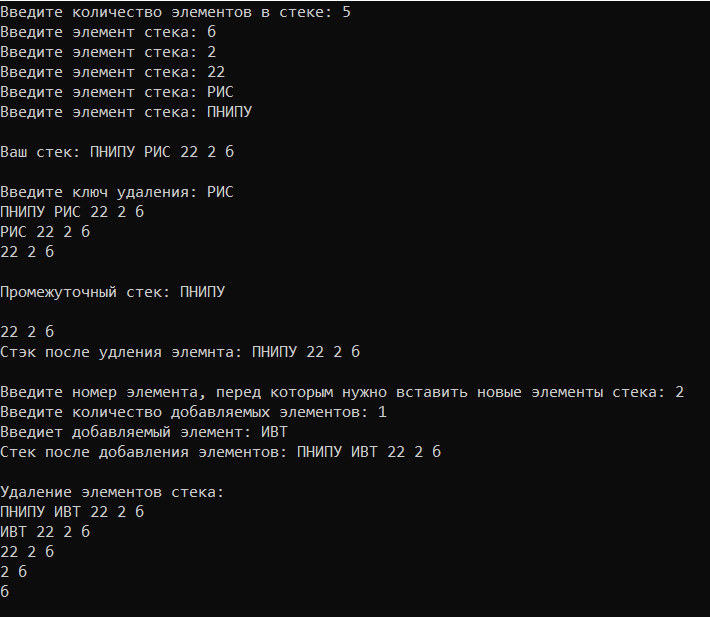
}

printStack(stack);

cout << endl;

return 0;

}

**Вывод программы**