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: «Информационные динамические структуры. Стеки»

Выполнил студент ИВТ-22-2б:

Захаров Дмитрий Сергеевич

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

Пермь 2023

**Постановка задачи**

Записи в линейном списке содержат ключевое поле типа int. Сформировать стек. Удалить из него К элементов, начиная с заданного номера, добавить элемент перед элементом с заданным ключом

**Алгоритм решения**

1. Создать структуру узла стека
2. Создать функцию создания стека
3. Создать функцию вставки элемента в стек
4. Создать функцию удаления элемента из стека
5. Создать функцию печати стека
6. Создать функцию записи стека в файл
7. Создать функцию уничтожения стека
8. Создать главную функцию main

**Код программы**

#include <iostream>

#include <fstream>

using namespace std;

struct Node // структура узла стека

{

int key; // ключевое поле

Node\* next; // указатель на следующий элемент

};

void createStack(Node\*& top) // создание пустого стека

{

top = NULL;

}

void push(Node\*& top, int key) // добавление элемента в стек

{

Node\* p = new Node; // создаем новый элемент

p->key = key; // присваиваем ему значение ключа

p->next = top; // новый элемент становится первым

top = p; // вершина стека указывает на новый элемент

}

void pop(Node\*& top) // удаление верхнего элемента из стека

{

if (top == NULL) // если стек пустой

{

cout << "Стек пустой." << endl;

return;

}

Node\* p = top; // сохраняем вершину стека

top = top->next; // вершина стека указывает на следующий элемент

delete p; // удаляем сохраненный элемент

}

void printStack(Node\* top) // печать стека

{

if (top == NULL) // если стек пустой

{

cout << "Стек пустой." << endl;

return;

}

while (top != NULL) // пока не достигнут конец стека

{

cout << top->key << " "; // печатаем значение ключа

top = top->next; // переходим к следующему элементу

}

cout << endl;

}

void writeStackToFile(Node\* top, string fileName) // запись стека в файл

{

ofstream file(fileName); // создаем файл для записи

if (file.is\_open()) // если файл открыт

{

while (top != NULL) // пока не достигнут конец стека

{

file << top->key << " "; // записываем значение ключа

top = top->next; // переходим к следующему элементу

}

file.close(); // закрываем файл

}

else // если файл не открыт

{

cout << "Ошибка открытия файла." << endl;

}

}

void destroyStack(Node\*& top) // уничтожение стека

{

while (top != NULL) // пока не достигнут конец стека

{

Node\* p = top; // сохраняем вершину стека

top = top->next; // вершина стека указывает на следующий элемент

delete p; // удаляем сохраненный элемент

}

}

void restoreStackFromFile(Node\*& top, string fileName) // восстановление стека из файла

{

destroyStack(top); // уничтожаем существующий стек

ifstream file(fileName); // открываем файл для чтения

if (file.is\_open()) // если файл открыт

{

int key;

while (file >> key) // пока не достигнут конец файла

{

push(top, key); // добавляем элемент в стек

}

file.close(); // закрываем файл

}

else // если файл не открыт

{

cout << "Ошибка открытия файла." << endl;

}

}

int main()

{

setlocale(LC\_ALL, "Russian"); // подключаем русский язык

Node\* top; // указатель на вершину стека

int n, k;

createStack(top); // создаем пустой стек

cout << "Введите количество элементов в стеке: ";

cin >> n;

for (int i = 0; i < n; i++) // заполняем стек

{

int key;

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

cin >> key;

push(top, key); // добавляем элемент в стек

}

cout << "Стек: ";

printStack(top); // печатаем стек

cout << "Введите номер элемента, который нужно добавить: ";

cin >> k;

push(top, k); // добавляем элемент с заданным номером

cout << "Стек после добавления элемента: ";

printStack(top); // печатаем стек

cout << "Введите количество элементов, которые нужно удалить: ";

cin >> k;

for (int i = 0; i < k; i++) // удаляем элементы из конца стека

{

pop(top);

}

cout << "Стек после удаления элементов: ";

printStack(top); // печатаем стек

writeStackToFile(top, "stack.txt"); // записываем стек в файл

destroyStack(top); // уничтожаем стек

restoreStackFromFile(top, "stack.txt"); // восстанавливаем стек из файла

cout << "Стек после восстановления из файла: ";

printStack(top); // печатаем стек

destroyStack(top); // уничтожаем стек

return 0;

}