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 №11.3**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Информационные и динамические структуры. Стеки

Вариант 24

Выполнила:

Студент группы ИВТ-20-2б

Ананина Арина Юрьевна

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

1. Постановка задачи

Сформировать стек. Распечатать полученную структуру. Выполнить обработку стека:

* Добавить в список элементы с номерами 1, 3, 5 и т. д.

Распечатать полученный результат. Удалить соответствующую структуру из памяти.

2. Анализ задачи

1. В программе используется структура Stack, int, stack<char>.

Тип int используется для хранения данных, как параметр в цикле, для временного хранения размера динамической структуры.

Int size;

Тип char используется для хранения данных внутри структуры.

cin>> p->data;

Тип stack<char>используется для хранения стека через STL.

2. С данными типа int производятся следующие действия:

математические операции (инкремент в циклах с параметром).

for (inti = 1; i<size; i++)

С данными типа Stack производятся следующие действия: создание стека, добавление элементов, вывод на экран.

Данные типа char используются только для хранения, и над ними не производятся какие-либо действия.

С данными типа list<char> производятся те же действия, что и с типом Stack, но они реализованы с использованием STL.

3. Стек реализован в программе с помощью структуры и STL.

4. Поля, используемые в структуре:

struct Stack

{

char data; //данные элемента стека

Stack\* prev; // ссылка на следующий объект

};

5. Ввод стека организован с клавиатуры, с помощью функции. В качестве параметров в функцию передаётся значение переменной n, которая отвечает за кол-во элементов стека. Функция возвращает ссылку на вершину стека.

Stack\* make\_stack(int n)

{

if (n == 0) return NULL;

Stack\* top, \* p;

top = NULL;

p = new Stack;

cout << "Введите 1 элемент стека: ";

cin >> a;

p->data = a;

p->prev = NULL;

top = p;

for (int i = 2; i <= n; i++)

{

Stack\* h = new Stack;

cout << "Введите " << i << " элемент стека: ";

cin >> a;

h->data = a;

h->prev = top;

top = h;

}

return top;

}

Вывод стека организован с помощью функции print. Так как стек читается с конца, то изначально в функции из стека все элементы переписываются в массив, а затем уже в правильном порядке с помощью цикла, который перебирает все элементы массива, выводятся на экран.

void print(Stack\* &last)

{

if (!last) {

cout << "Стек пустой" << endl;

return;

}

vector<char> copy = vector<char>();

while (last)

{

copy.push\_back(pop(&last));

}

for (int i = copy.size() - 1; i >= 0; i--) {

cout << copy[i] << " ";

push(&last, copy[i]);

}

cout << endl;

}

6. Функция replace заменяет значения элементов, стоящих на нечётных местах, на новые.

Сначала все элементы стека переписываются в массив, затем с помощью цикла осуществляется проход по массиву, если индекс элемента нечётный, то программа запрашивает новое значение элемента, и оно записывается в массив на соответствующее место. В этом же цикле осуществляется перенос элементов обратно в стек.

void replace(Stack\* &top)

{

if (!top)

{

cout << "Стек пустой" << endl;

return;

}

vector<char> ar = vector<char>();

while (top)

{

ar.insert(ar.begin(), 1, pop(&top));

}

for (int j = 0; j < ar.size(); j++) {

if (j % 2 == 0) {

cout << "Введите " << j + 1 << " элемент стека: ";

cin >> ar[j];

}

push(&top, ar[j]);

}

cout << endl;

}

3. Код программы

#include <iostream>

#include <stack>

using namespace std;

char a;

stack<char> MakeSTL(int size)

{

stack<char> res;

char tmp;

for (int i = 0; i < size; i++)

{

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

cin >> tmp;

res.push(tmp);

}

return res;

}

void PrintSTL(stack<char> st)

{

if (st.size() > 0)

{

stack<char> t;

while (!st.empty())

{

t.push(st.top());

st.pop();

}

while (!t.empty())

{

cout << t.top() << " ";

t.pop();

}

cout << endl;

}

else

{

cout << "Стек пустой" << endl;

}

}

stack<char> ReplaceSTL(stack<char> st)

{

int n = st.size();

stack<char> per, upend;

char\* ar = new char[n];

char tmp;

while (!st.empty())

{

upend.push(st.top());

st.pop();

}

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

{

if (i % 2 == 0)

{

cout << "Введите " << i+1 << " элемент: " << endl;

cin >> tmp;

ar[i] = tmp;

upend.pop();

}

else

{

ar[i] = upend.top();

upend.pop();

}

}

for (int j = 0; j < n; j++)

{

per.push(ar[j]);

}

return per;

}

struct Stack

{

char data;

Stack\* prev;

};

void push(Stack\*\* last, char element)

{

Stack\* newElement = new Stack;

newElement->data = element;

newElement->prev = \*last;

(\*last) = newElement;

}

char pop(Stack\*\* last)

{

char el = (\*last)->data;

Stack\* rem = \*last;

\*last = (\*last)->prev;

delete rem;

return el;

}

Stack\* make\_stack(int size)

{

Stack\* res = 0;

char tmp;

for (int i = 0; i < size; i++)

{

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

cin >> tmp;

push(&res, tmp);

}

return res;

}

void print(Stack\* &last)

{

if (!last) {

cout << "Стек пустой" << endl;

return;

}

vector<char> copy = vector<char>();

while (last)

{

copy.push\_back(pop(&last));

}

for (int i = copy.size() - 1; i >= 0; i--) {

cout << copy[i] << " ";

push(&last, copy[i]);

}

cout << endl;

}

void replace(Stack\* &top)

{

if (!top)

{

cout << "Стек пустой" << endl;

return;

}

vector<char> ar = vector<char>();

while (top)

{

ar.insert(ar.begin(), 1, pop(&top));

}

for (int j = 0; j < ar.size(); j++) {

if (j % 2 == 0) {

cout << "Введите " << j + 1 << " элемент стека: ";

cin >> ar[j];

}

push(&top, ar[j]);

}

cout << endl;

}

int main()

{

system("color F0");

setlocale(LC\_ALL, "RUS");

cout << "Стек через STL библиотеку." << endl;

int size;

do

{

cout << "Введите кол-во элементов стека: ";

cin >> size;

} while (size <= 0);

stack<char> st1 = MakeSTL(size);

PrintSTL(st1);

stack<char> st2= ReplaceSTL(st1);

PrintSTL(st2);

cout << "Стек через структуру." << endl;

int size2;

do

{

cout << "Введите кол-во элементов стека: ";

cin >> size2;

} while (size2 <= 0);

Stack\* st3 = make\_stack(size2);

print(st3);

replace(st3);

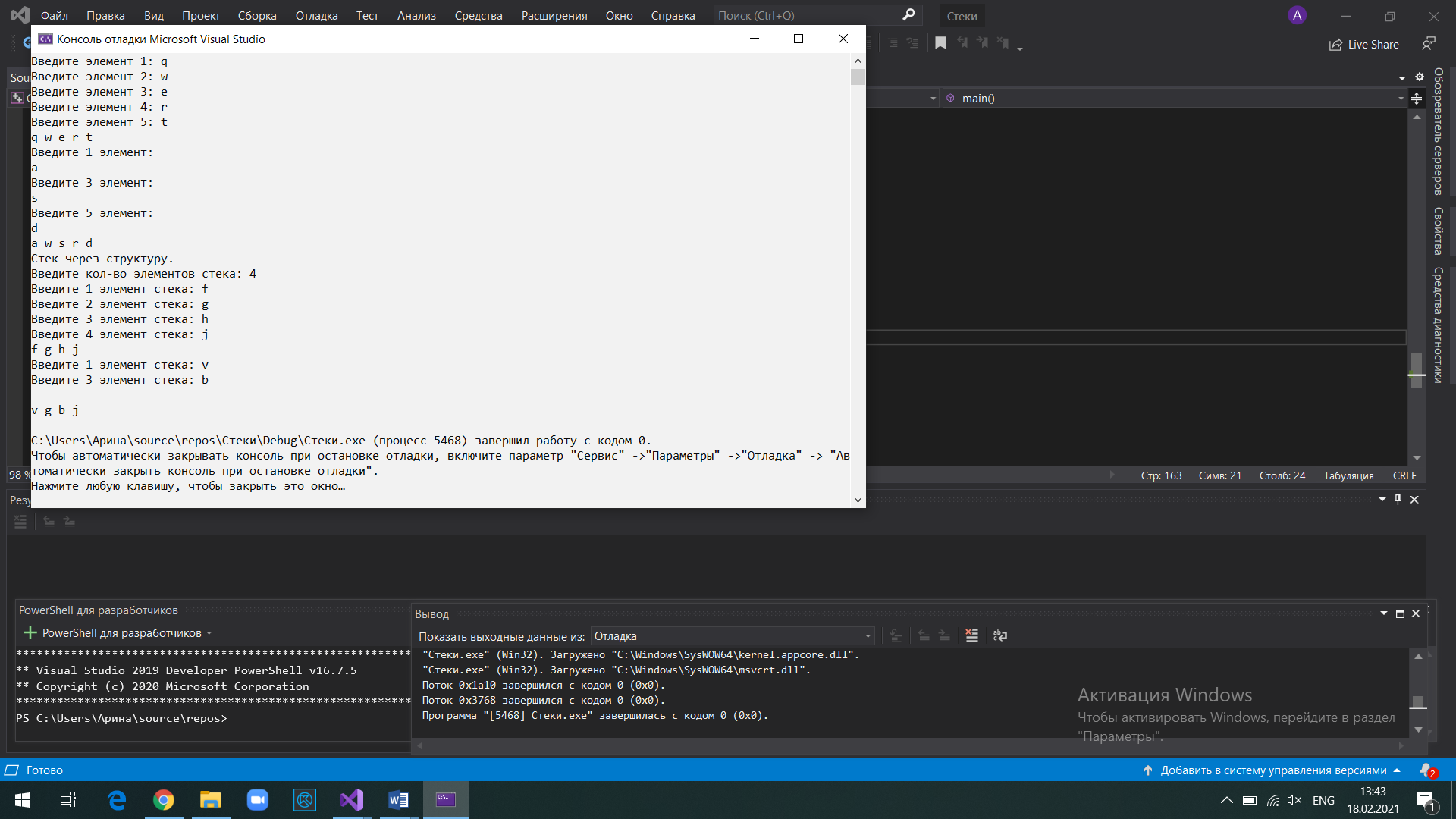
print(st3);

return 0;

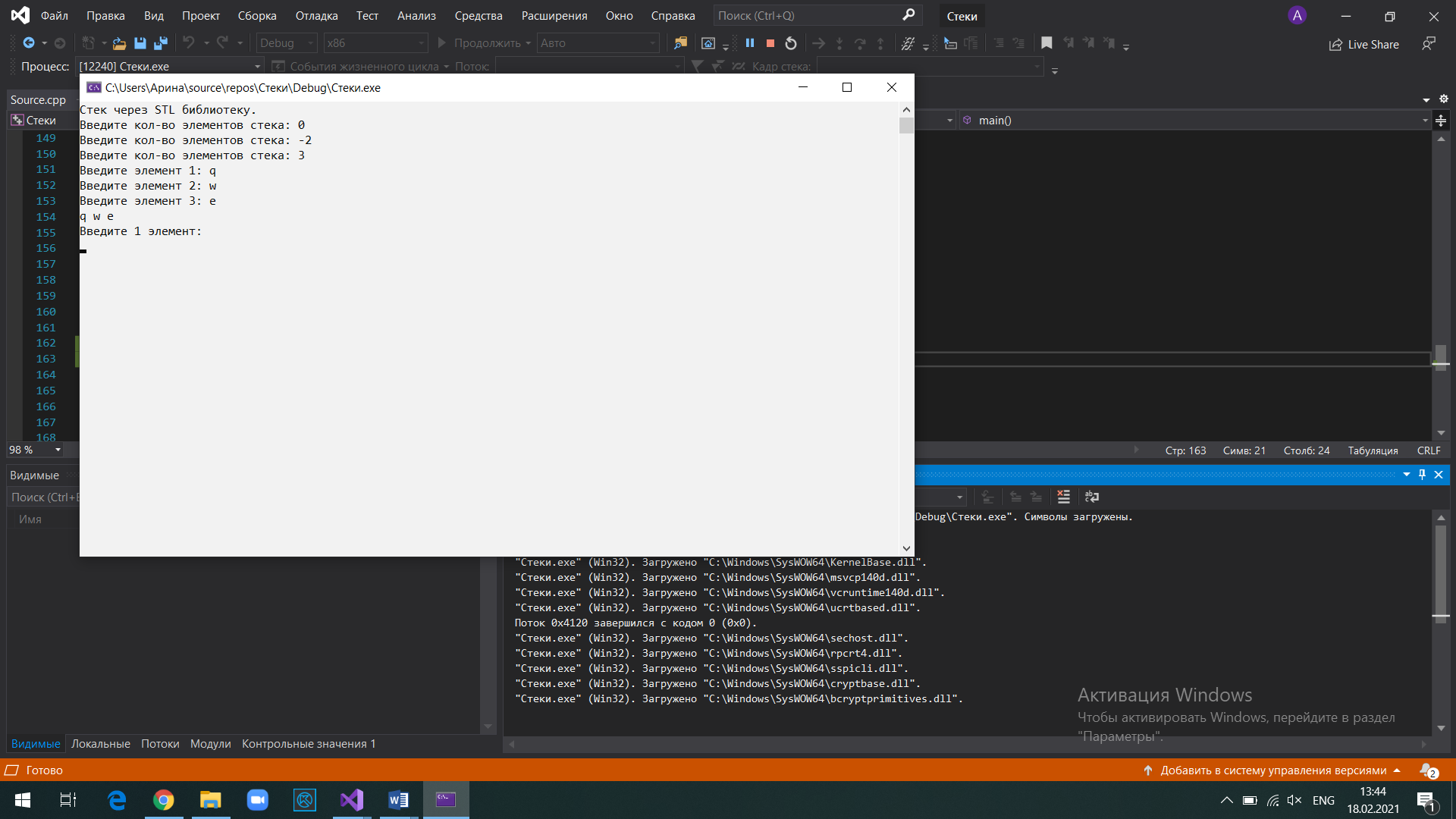
}

4. Выполнение программы.

Работа программы, когда все данные введены корректно.



Если пользователь вводит в качестве кол-ва элементов стека число, меньшее нуля, то программа запрашивает это число до тех пор, пока не будет введено корректное значение.



5. Блок схема

