МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3
по дисциплине «Алгоритмы и структуры данных»
Тема: Линейные структуры данных: стек и очередь
Вариант 4-в

Студент гр. 8383 _____ Шишкин И.В.

Преподаватель _____ Фирсов М.А.

Санкт-Петербург
2019

Цель работы.

Ознакомиться с часто используемыми на практике линейными структурами данных, обеспечивающими доступ к элементам последовательности только через её начало и конец, и способами реализации этих структур, освоить на практике использование стека.

Постановка задачи.

4. Содержимое заданного текстового файла F, разделенного на строки, переписать в текстовый файл G, выписывая литеры каждой строки в обратном порядке.

Описание алгоритма.

Программа считывает каждую строку текста, узнает длину каждой строки и создает стек, размер которого и будет определен длиной строки. Затем, вызывается рекурсивная функция `void inStack(STACK <char>& st, string s, int curr, int max)`, в которой каждый литерал строки s заносится в стек st. После этого вызывается функция `void outStack(STACK <char> &st, ofstream& f)`, в которой литерал из стека st записывается в файл f, в результате чего в этом файле получается вывод изначального текста в обратном порядке. Происходит это потому, что стек работает по принципу Last In First Out.

Описание класса STACK.

В классе STACK в приватном поле хранятся переменные Item* s - элемент в стеке, int size - размер стека и int top - индекс верхнего элемента в стеке. Реализованы функции STACK(int = 100) - конструктор стека, ~STACK() - деструктор стека, inline void push(Item) - функция добавления элемента в стек, inline Item pop() - функция удаления элемента из стека, inline void printStack() - функция печати стека, inline const Item& peek(int) const - функция, возвращающая верхний элемент стека, inline int getTop() const - функция, возвращающая индекс верхнего элемента стека.

Спецификация программы.

Программа считывает содержимое файла F, заносит каждую строку в стек и выводит каждый литерал каждой строки в обратном порядке в файл G.

Программа написана на языке C++. Входные данные считываются из файла. Выходные данные записываются файл и выводятся на экран.

Тестирование.

```
Доступные файлы для выбора:
t.txt
test1.txt
test2.txt
test3.txt
Введите файл, который нужно считать
test3.txt
Содержимое начального файла:
ABCde 12
3 4 %%7583
Введите файл, в который нужно записать результат
res.txt
push for element A
 push for element B
  push for element C
   push for element d
    push for element e
     push for element
      push for element 1
       push for element 2
       pop for element 2
      pop for element 1
     pop for element
    pop for element e
   pop for element d
  pop for element C
 pop for element B
pop for element A
push for element 3
 push for element
  push for element 4
   push for element
    push for element %
     push for element %
      push for element 7
       push for element 5
        push for element 8
         push for element 3
         pop for element 3
        pop for element 8
       pop for element 5
      pop for element 7
     pop for element %
    pop for element %
   pop for element
  pop for element 4
 pop for element
pop for element 3

Содержимое конечного файла:
21 edCBA
3857%% 4 3
```

Input	Output
1	1

qwerty WWRQR vzxbn d reuia $&@678 vcz	RQRWW ytrewq nbxzv d aiuer 876@&$ zcv
qwertyuiop[]asdfghjkl;'\zxcvbnm,./1234560899- yfdashf vhcxj	-9980654321/.,mnbvcxz\';lkjhgfdsa][poiuytrewq fhsadfy jxchv
hello hi hello kl;;ldlvzk 1234567890-	olleh ih olleh kzvldl;;lk -0987654321

Вывод.

В ходе выполнения лабораторной работы были усовершенствованы навыки работы со стеком на базе массива, а также реализован алгоритм инверсии каждой строки текста на языке программирования C++.

Приложение.

Код программы.

Содержимое файла headder.h

```
#include <iostream>
#include <fstream>
#include <Windows.h>
#include <stdlib.h>
#include <string>

#pragma once

template <class Item>
class STACK {
private:
	Item* s;	//элемент в стеке
	int size;	//размер всего стека
	int top;	//индекс верхнего элемента стека
public:
	STACK(int = 100);
	~STACK();

	inline void push(Item);	//функция добавления элемента в стек
	inline Item pop();		//функция удаления элемента из стека
	inline void printStack();	//функция печати стека
	inline const Item& peek(int) const;	//функция, возвращающая верхний элемент стека
	inline int getTop() const;	//функция, возвращающая индекс верхнего элемента
стека
};

template <class Item>
STACK<Item>::STACK(int currSize): size(currSize){
	s = new Item[size];

	if (s == NULL) {
		delete []s;
		exit(0);
	}
	top = 0;
}

template <class Item>
STACK<Item>::~STACK() {
	delete []s;
}

template <class Item>
inline void STACK<Item>::push(Item elem) {
	for (int i = 0; i < top; i++) std::cout << " ";
	std::cout << "push for element " << elem << std::endl;
	s[top++] = elem;
}

template <class Item>
inline Item STACK<Item>::pop() {
	for (int i = 0; i < top - 1; i++) std::cout << " ";
	std::cout << "pop for element " << s[top-1] << std::endl;
	return s[--top];
}
```

```
template <class Item>
inline void STACK<Item>::printStack() {
	for (int i = top - 1; i >= 0; i--) {
		std::cout << s[i] << std::endl;
	}
}

template <class Item>
inline int STACK<Item>::getTop() const {
	return top;
}

template <class Item>
inline const Item& STACK<Item>::peek(int num) const{
	if (num >= top)
		exit(3);

	return s[top - num - 1]; // вернуть n-й элемент стека
}
```

Содержимое файла source.cpp

```
#include "Headder.h"

using namespace std;

void inStack(STACK <char>& st, string s, int curr, int max);		//функция для записывания каждого литерала строки в стек
void outStack(STACK <char> &st, ofstream& f);		//функция для записывания каждого литерала из стека в файл

int main() {
	setlocale(LC_ALL, "RUS");
	int check = 1;
	string file;	//переменная, хранящая в себе путь к файлу, из которого будут считаны строки
	string file2;	//переменная, хранящая в себе путь к файлу, в который будет выведен результат
	string fileContents;	//переменная, предназначенная для вывода содержимого начального файла и результата на экран
	ifstream in;	//начальный файл
	ofstream out;	//файл с результатом
	int t = 0;
	int curr = 0;

	cout << "Доступные файлы для выбора:\n";
	WIN32_FIND_DATA FindFileData;
	HANDLE hf;
	hf = FindFirstFile(".\\files\\*.txt", &FindFileData);
	if (hf != INVALID_HANDLE_VALUE) {
		do {
			cout << FindFileData.cFileName << endl;
		} while (FindNextFile(hf, &FindFileData) != 0);
		FindClose(hf);
	}

	cout << "Введите файл, который нужно считать\n";
	cin >> file;
	file.insert(0, ".\\files\\");
	in.open(file);
	if (!in) {
		cout << "WRONG FILE!\n";
		return 0;
	}
	else {
```

```
		cout << "Содержимое начального файла:\n";
		while (!in.eof()) {
			getline(in, fileContents);
			cout << fileContents << endl;
		}
		in.close();
		in.open(file);
	}

	fileContents.clear();
	cout << "Введите файл, в который нужно записать результат\n";
	cin >> file2;
	file2.insert(0, ".\\files\\");
	out.open(file2);

	while (!in.eof()) {
		getline(in, fileContents);
		STACK <char> st(fileContents.size());
		inStack(st, fileContents, curr, fileContents.size());
		outStack(st, out);
		t = st.getTop();
		for (int i = 0; i < t; i++) st.pop();
	}

	out.close();
	in.close();
	in.open(file2);
	cout << "\nСодержимое конечного файла:\n";
	while (!in.eof()) {
		getline(in, fileContents);
		cout << fileContents << endl;
	}

	return 0;
}

void inStack(STACK <char>& st, string s, int curr, int max) {
	if (curr == max) return;
	st.push(s[curr]);
	inStack(st, s, curr + 1, max);
	//for (int i = 0; i < s.size(); i++) st.push(s[i]);
}

void outStack(STACK <char> &st, ofstream &f) {
	for (int i = 0; i < st.getTop(); i++) {
		f.put(st.peek(i));
	}
	f.put('\n');
}
```