Практическая работа № 3

СТЕКИ

Постановка задачи

Составить программу создания стека на базе линейного односвязного списка и реализовать основные алгоритмы работы со стеком, обеспечивающие следующие действия:

1 - Добавить элемент

2 – Вывести верхний элемент

2.1 – С удалением

2.2 – Без удаления

3 – Проверить наличие элементов

4 - Вычислить длину стека

5 - Вывести (распечатать) стек на экран

Перечисленные действия оформить в виде самостоятельных режимов работы созданного стека . Выбор режимов производить с помощью пользовательского меню.

Провести полное тестирование (всех режимов работы) программы на стеке, сформированном вводом с клавиатуры. Тест-примеры определить самостоятельно. Результаты тестирования в виде скриншотов экранов включить в отчет по выполненной работе.

Оформить отчет с подробным описанием созданного стека, принципов программной реализации алгоритмов работы со стеком, описанием текста исходного кода и проведенного тестирования программы.

Сделать выводы о проделанной работе, основанные на полученных результатах.

1. Описание алгоритма

Алгоритм программы состоит из функции main и вызываемых в ней вспомогательных функций:

- **void add**– функция добавления элемента в стек
- **void pop** – функция извлечения элемента из стека (удаление)
- **void peek** – функция возврата верхнего элемента (без удаления)
- **void show –** функция вывода стека на экран
- **void not_empty** – функция проверки наличия элементов
- **void len –** функция вычисления длинны стека

Стек (англ. stack — стопка; читается стэк) — абстрактный тип данных, представляющий собой список элементов, организованных по принципу LIFO (англ. last in — first out, «последним пришёл — первым вышел»). Чаще всего принцип работы стека сравнивают со стопкой тарелок: чтобы взять вторую сверху, нужно снять верхнюю. В цифровом вычислительном комплексе стек называется магазином — по аналогии с магазином в огнестрельном оружии (стрельба начнётся с патрона, заряженного последним). В 1946 Алан Тьюринг ввёл понятие стека. А в 1957 году немцы Клаус Самельсон и Фридрих Л. Бауэр запатентовали идею Тьюринга.

Рис.1 Принцип работы стека

Функция main создает класс Stack и вызывает функцию menu.

Функция peek возвращает значение верхнего элемента.

Функция not_empty проверяет не является ли головной элемент NULL.

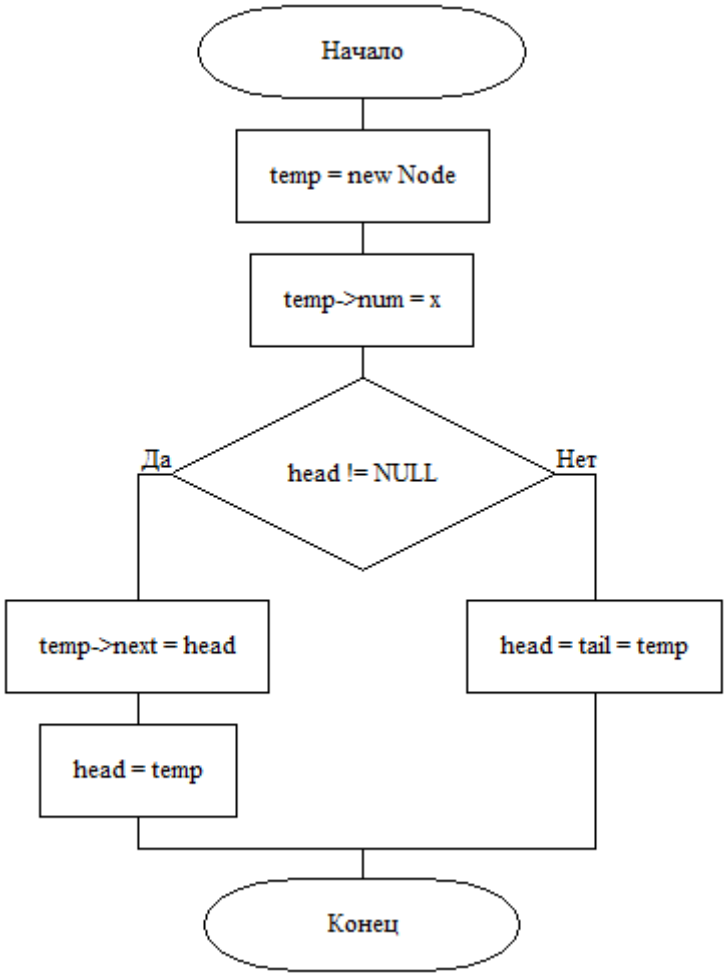

Рис.2 Схема алгоритма функции add

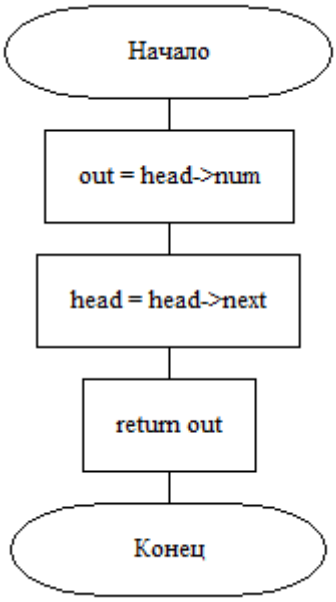

Рис.3 Схема алгоритма функции pop

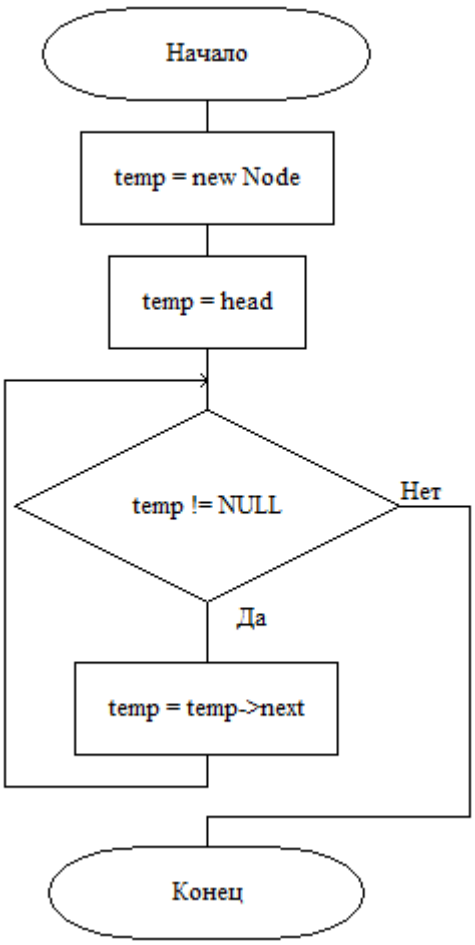

Рис.4 Схема алгоритма функции show

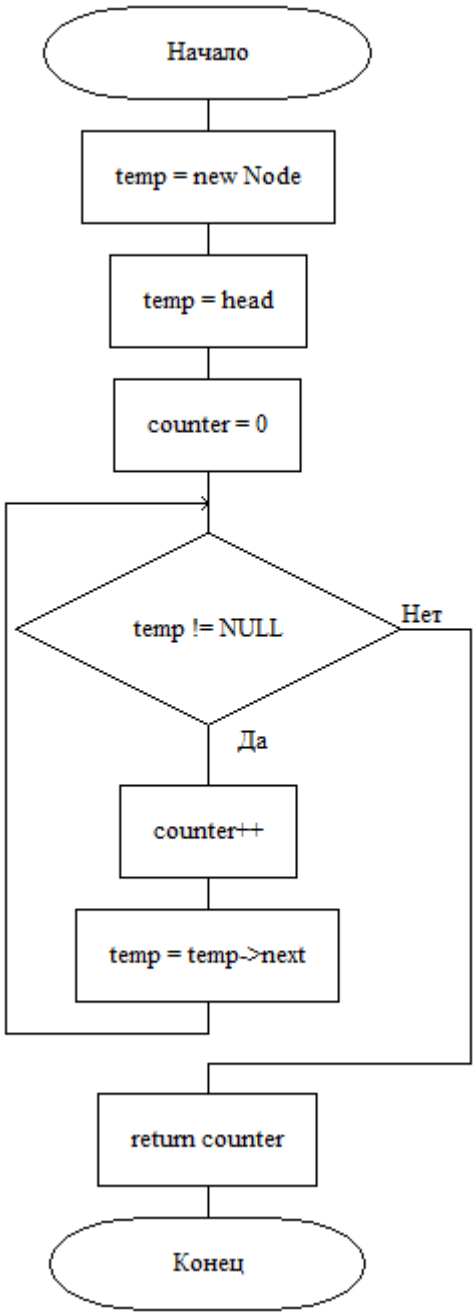

Рис.5 Схема алгоритма функции len

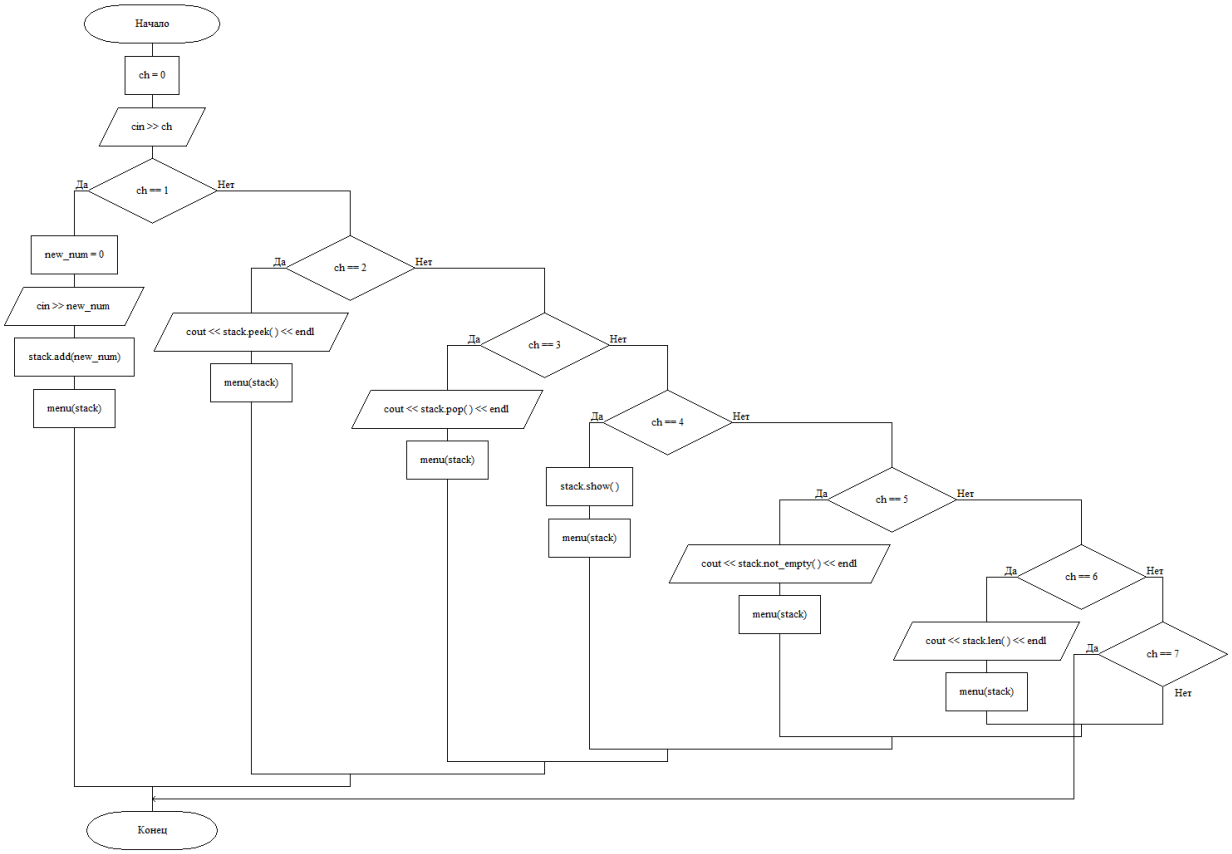

Рис.6 Схема алгоритма функции menu

Реализация алгоритма

Текст исходного кода программы

main.cpp

```
#include "Stack.h"
#include <iostream>

void menu(Stack stack) {
	cout << "Выберите команду:" << endl;
	cout << "[1] - Добавить элемент." << endl;
	cout << "[2] - Вывести верхний элемент (без удаления)." << endl;
	cout << "[3] - Вывести верхний элемент (с удалением)." << endl;
	cout << "[4] - Вывести стек на экран." << endl;
	cout << "[5] - Проверить пустой ли стек." << endl;
	cout << "[6] - Вывести длинну стека." << endl;
	cout << "[7] - Завершить программу." << endl;
	cout << "----> ";
	int ch = 0;
	cin >> ch;
	if (ch == 1) {
		int new_num = 0;
		cout << "Введите число: ";
		cin >> new_num;
		stack.add(new_num);
		cout << "Элемент "<<new_num<<" добавлен." << endl;
		menu(stack);
	}
	else if(ch == 2)
	{
		cout<<stack.peek()<<endl;
		menu(stack);
	}
	else if (ch == 3)
	{
		cout << stack.pop() << endl;
		menu(stack);
	}
	else if (ch == 4)
	{
		stack.show();
		menu(stack);
	}
	else if (ch == 5)
	{
		cout << stack.not_empty()<<endl;
		menu(stack);
	}
	else if (ch == 6)
	{
		cout<<stack.len()<<endl;
		menu(stack);
	}
	else if (ch == 7)
	{
		cout << "Программа завершена.";
		return;
	}
```

```
}


int main()
{
	setlocale(LC_ALL, "Russian");
	Stack *st = new Stack;
	menu(*st);

}
```

Stack.h

```
#include <iostream>
using namespace std;
#pragma once

struct Node
{
	int num;
	Node* next = NULL;
};

class Stack
{
private:
	Node* head = NULL, * tail = NULL;
public:
	void add(int x);
	int pop();
	void show();
	int peek();
	bool not_empty();
	int len();

};
```

Stack.cpp

```
#include "Stack.h"
void Stack::add(int x) {
	Node* temp = new Node;
	temp->num = x;
	if (head != NULL) {
		temp->next = head;
		head = temp;
	}
	else head = tail = temp;
}

int Stack::pop() {
	int out = head->num;
	head = head->next;
	return out;
}
```

```
void Stack::show() {
	Node* temp = new Node;
	temp = head;
	while (temp != NULL) {
		cout << temp->num << endl;
		temp = temp->next;
	}
}

int Stack::peek() {
	return head->num;
}

bool Stack::not_empty() {
	return head != NULL;
}

int Stack::len() {
	Node* temp = new Node;
	temp = head;
	int counter = 0;
	while (temp != NULL) {
		counter++;
		temp = temp->next;
	}
	return counter;
}
```

2. Тестирование программы

Ниже представлен результат работы программы с введённым стеком

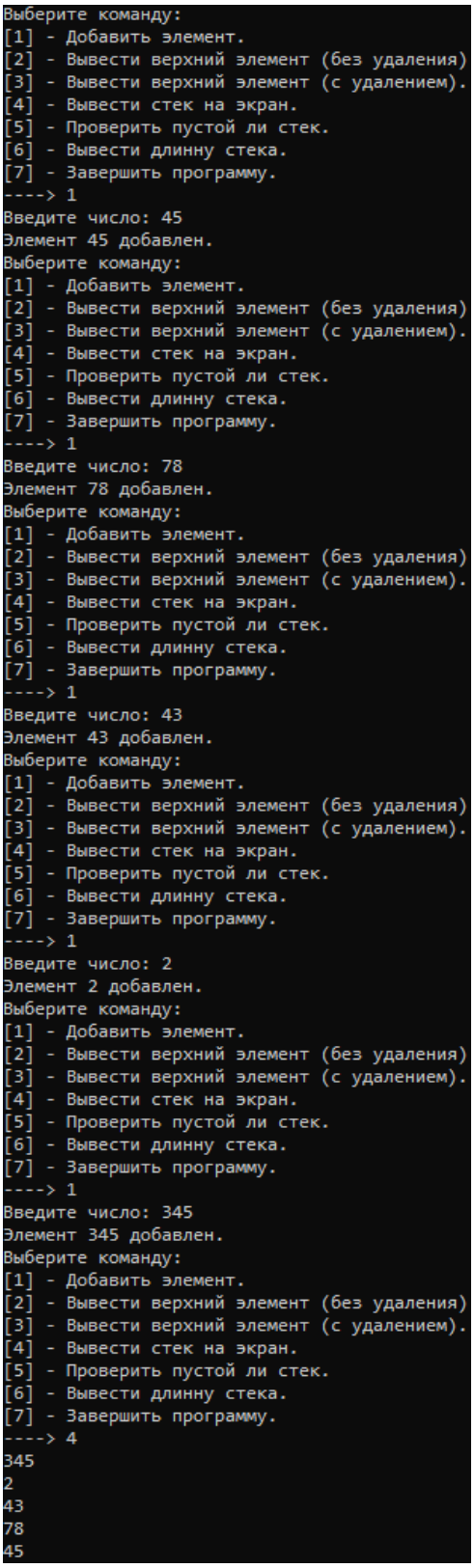

Рис.7 Скриншот добавления элементов в стек и вывод стека

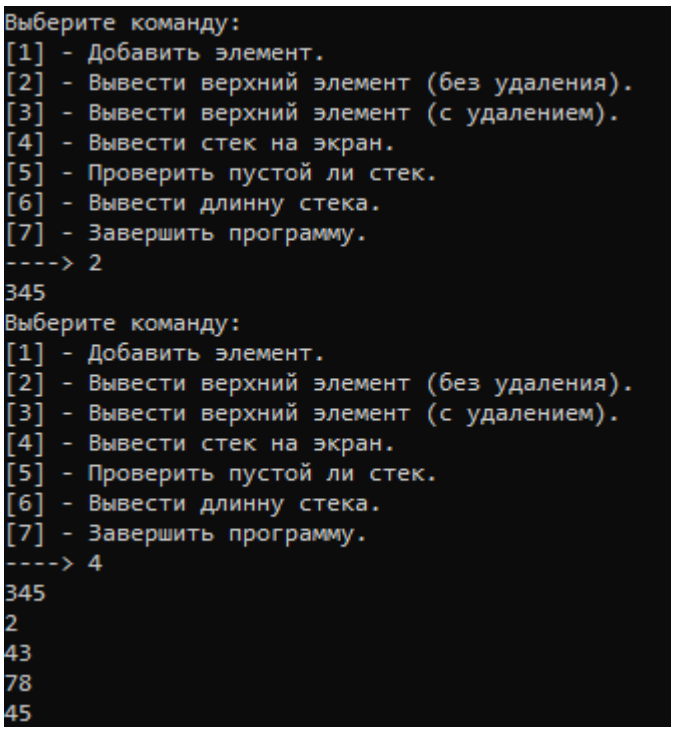

Рис.8 Скриншот вывода верхнего элемента (без удаления) и всего стека

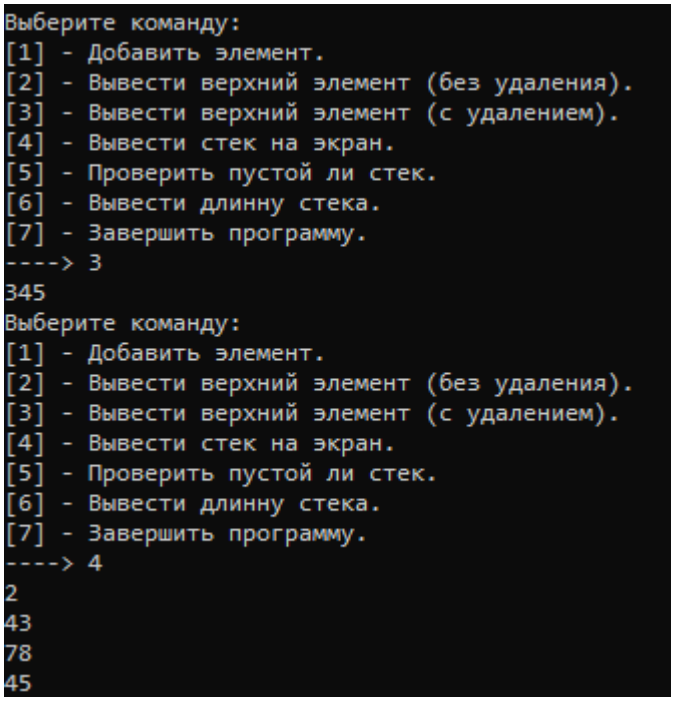

Рис.9 Скриншот вывода верхнего элемента (с удалением) и вывод стека

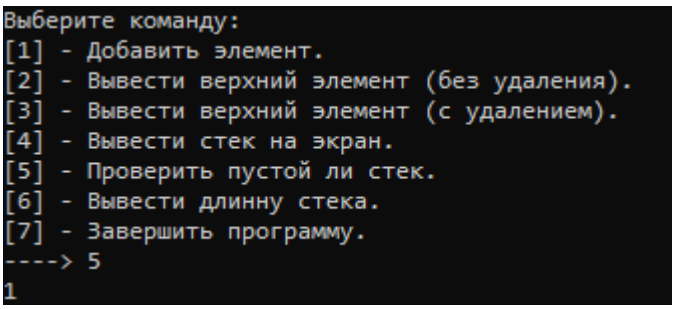

Рис.10 Скриншот проверки стека на пустоту

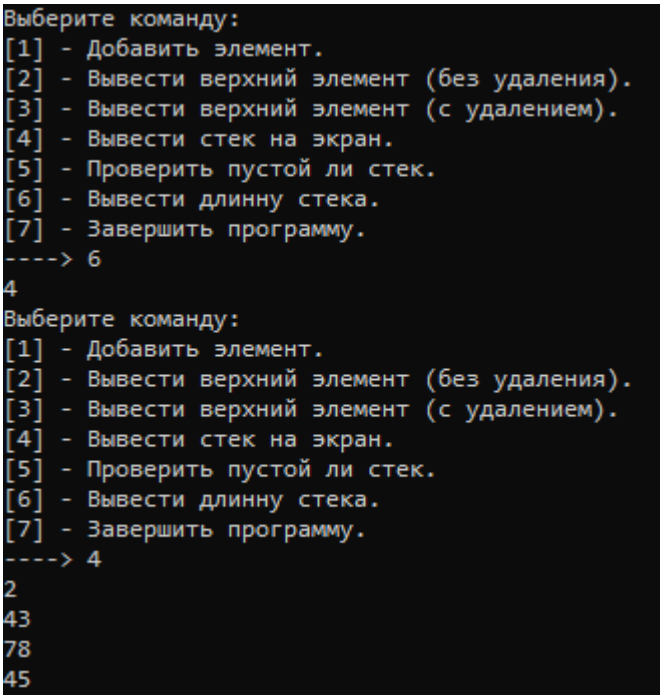

Рис.11 Скриншот вывода длинны стека и всего стека

3. Выводы

1. В ходе работы был создан стек на основе односвязного списка, ссылки между элементами которого осуществляются при помощи указателей на объекты класса Node.
2. Также были реализованы функции работы со стеком: добавление элементов, удаление, вывод всех элементов списка, вывод длины.
3. Были изучены положительные и негативные стороны стека:
4. Преимущества: стек может хранить неограниченное количество данных любого типа.
5. Недостатки: элементы можно извлекать только сверху, что усложняет взаимодействие с элементами.
6. Таким образом, была изучена работа стека на базе линейного связного списка и функций работы над ними и их реализация.

Список используемых информационных источников

1. Сыромятников В.П. Структуры и алгоритмы обработки данных, лекции, РТУ МИРЭА, Москва, 2020/2021 уч./год.
2. Документация по языку программирования C++, интернет-ресурс: https://en.cppreference.com/w/ (Дата обращения – 02.11.2020)
3. Интегрированная среда разработки для языков программирования C и C++, разработанная компанией JetBrains - CLion / Copyright © 2000-2020 JetBrains s.r.o., интернет-ресурс: https://www.jetbrains.com/clion/learning-center/ (Дата обращения – 02.11.2020).
4. ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения. Интернет-ресурс: http://docs.cntd.ru/document/gost-19-701-90-espd (Дата обращения – 02.11.2020).
5. Описание Стека. интернет-ресурс: https://ru.wikipedia.org/wiki/Стек (Дата обращения – 02.11.2020).