**Сопроводительный лист на издание в авторской редакции**

Название работы Динамическая структура данных «стек». Реализация в связном списке Методические указания к выполнению лабораторной работы № 31 для студентов очной формы обучения по направлениям подготовки 09.03.04 – «Программная инженерия», 09.03.01 «Информатика и вычислительная техника», 02.03.03 «Математическое обеспечение и администрирование информационных систем»

Актуальность и соответствующий   
научно-методический уровень подтверждаю \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись научного редактора)

Рукопись сверена и проверена автором \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись автора)

Рекомендуется к изданию \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись заведующего кафедрой)



**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

Брянский государственный технический университет

Утверждаю

Ректор университета

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.Н. Федонин

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2017

ПРОГРАММИРОВАНИЕ

ДИНАМИЧЕСКАЯ СТРУКТУРА ДАННЫХ «СТЕК». РЕАЛИЗАЦИЯ В СВЯЗНОМ СПИСКЕ

**Методические указания**

**к выполнению лабораторной работы № 31**

**для студентов очной формы обучения**

**по направлениям подготовки**

**09.03.04 – «Программная инженерия»,**

**09.03.01 «Информатика и вычислительная техника»,**

**02.03.03 «Математическое обеспечение и администрирование информационных систем»**

**Брянск 2017**

**УДК 004.**

Динамическая структура данных «стек». Реализация в связном списке [Текст] + [Электронный ресурс]: методические указания к выполнению лабораторной работы № 31 для студентов очной формы обучения по направлениям подготовки 09.03.04 – «Программная инженерия», 09.03.01 «Информатика и вычислительная техника», 02.03.03 «Математическое обеспечение и администрирование информационных систем». – Брянск, 2017. – 7 с.

Разработали:

Д. И. Булатицкий

канд. техн. наук, доц.,

Е..В. Коптенок

ассистент

Рекомендовано кафедрой «Информатика и программное обеспечение» БГТУ

**Методические издания публикуются в авторской редакции**

# Цель работы

Цель работы – научиться реализации динамической структуры «стек» с помощью связанного списка.

Продолжительность работы – 2 часа.

# ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

**Динамические структуры данных** – это структуры данных, память под которые выделяется и освобождается не на этапе компиляции, а в процессе работы программы.

**Динамический элемент** – это элемент динамической структуры, который в конкретный момент выполнения программы может либо существовать, либо отсутствовать в памяти.

**Динамическое распределение памяти** – это выделение памяти под отдельные элементы в тот момент, когда они "начинают существовать" в процессе выполнения программы.

**Информационное поле (поле данных)** – это поле структуры, в котором содержатся непосредственно обрабатываемые данные.

**Связное представление данных** – это установление связи между элементами динамической структуры через указатели.

По способу обхода (доступа) можно выделить следующие виды ДСД (динамических структур данных):

* Список;
* Стек;
* Очередь;
* Очередь с приоритетом;
* Кольцевая очередь;
* Ассоциативный массив;

По количеству связей элементов с «соседними» элементами:

* Односвязные;
* Двухсвязные;
* Многосвязные.

По характеру связи:

* Линейные. В линейной динамической структуре данные связываются в цепочку. К линейным структурам относятся списки (односвязные, двухсвязные, кольцевые), стеки, очереди (односторонние, двухсторонние, очереди с приоритетами).
* Нелинейные (деревья, графы).

Из всего многообразия ДСД в рамках данной лекции рассмотрим только наиболее распространённые из них.

## Cтек (LIFO)

**Стек** - такой последовательный список с переменной длиной, включение и исключение элементов из которого выполняются только с одной стороны списка, называемого вершиной стека (Рис.2.1.).

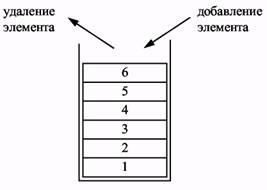


Рис.2.1. Динамическая структура «Стек»

Основные операции над стеком - включение нового элемента (английское название push - заталкивать) и исключение элемента из стека (англ. pop - выскакивать).

Полезными могут быть также вспомогательные операции: определение текущего числа элементов в стеке; очистка стека; неразрушающее чтение элемента из вершины стека, которое может быть реализовано, как комбинация основных операций.

# ХОД РАБОТЫ

**Задание 1.** Интерфейс ДСД «Стек» на основе динамических массивов.

Как было показано ранее, основными операциями со над стеком являются вставка в стек и извлечение из него. К вспомогательным операциям относят:

* определение текущего числа элементов в стеке или проверка наличия элементов в стеке;
* очистку стека и неразрушающее чтение элемента из вершины стека;
* печать стека.

С учётом необходимости управления динамической памятью необходимо также добавить следующие служебные операции: инициализация и уничтожение стека.

Схематично работа со стеком представлена на Рис.3.1.

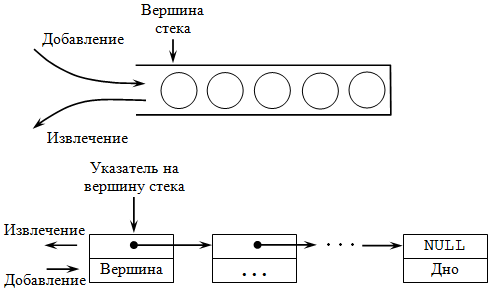


Рис.3.1. Работа со структурой «Стек»

Иногда удобнее представить список в виде связной структуры. Для возможности в программе одновременно работать со списками различной реализации, переопределим также и интерфейс с другим идентификатором типа.

**Шаг 1.** Поместим описание интерфейса стека в заголовочный файл «StackL.h»:

#ifndef \_STACK\_L\_H

#define \_STACK\_L\_H

struct LE

{

double v;

struct LE \*next;

};

typedef struct

{

struct LE \*top;

} StackL;

int Push(StackL \*s, double v);

double Pop(StackL \*s);

double Peek(StackL const \*s);

int IsEmptyStack(StackL const \*s);

void Clear(StackL \*s);

void InitStack(StackL \*s);

void PrintStack(StackL const \*s);

void Destruct(StackL \*s);

#endif

**Шаг 2.** Реализацию поместим его в файл «StackL.c»:

#include <stdio.h>

#include "StackL.h"

void InitStack(StackL \*s)

{

s->top=NULL;

}

int Push(StackL \*s, double v)

{

struct LE \*p=(struct LE \*)malloc(sizeof(struct LE));

if(!p) return 0;

p->next=s->top;

p->v=v;

s->top=p;

return 1;

}

double Pop(StackL \*s)

{

struct LE \*p;

double x=1e300;

if(s->top)

{

p=s->top;

s->top=s->top->next;

x=p->v;

free(p);

}

return x;

}

double Peek(StackL const \*s)

{

if(s->top)

return s->top->v;

}

int IsEmptyStack(StackL const \*s)

{

return s->top==NULL;

}

void Clear(StackL \*s)

{

while(!IsEmptyStack(s)) Pop(s);

}

void PrintStack(StackL const \*s)

{

struct LE \*p;

for(p=s->top; p; p=p->next)

printf("%-7.3lf ", p->v);

}

void Destruct(StackL \*s)

{

Clear(s);

}

**Шаг 3.** Пример работы со стеком:

#include <stdio.h>

#include <stdlib.h>

#include "stackL.h"

int main(int argc, char \*argv[])

{

StackL a, b;

int i;

double x;

InitStack(&a);

Push(&a, 1.1);

Push(&a, 2.2);

printf ("Enter value to push: ");

scanf("%lf", &x);

Push(&a, x);

PrintStack(&a);

for (i=0; i<5; i++) printf("\n Extracted %-7.3lf\n", Pop(&a));

PrintStack(&a);

Destruct(&a);

system("PAUSE");

return 0;

}

# ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ

Самостоятельно реализуйте программу, реализующую работу с представленной библиотекой. Формулировку задачи взять согласно варианту Nзадания=Nвар%6+1:

1. Ввести некоторое число и записать его цифры в стек. Вывести число, у которого цифры идут в обратном порядке. Цифры из записи числа считывать поочередно (например, с помощью getch()).
2. Ввести некоторое число и записать его цифры в стек. Цифры из записи числа считывать поочередно (например, с помощью getch()). Вычислить сумму и произведение его цифр.
3. Ввести некоторое число и записать его цифры в стек. Вывести число, у которого цифры идут в обратном порядке. Цифры из записи числа считывать поочередно (например, с помощью getch()). Определить, верно ли, что цифры, стоящие на нечетных позициях являются четными, а на четных позициях – нечетными.
4. Ввести некоторое число и записать его цифры в стек. Вывести число, у которого цифры идут в обратном порядке. Цифры из записи числа считывать поочередно (например, с помощью getch()). Вывести все цифры, стоящие на нечетных позициях.
5. Ввести некоторое число и записать его цифры в стек. Вывести число, у которого цифры идут в обратном порядке. Цифры из записи числа считывать поочередно (например, с помощью getch()). Вычислить наименьшую и наибольшую цифры в записи числа.
6. Ввести некоторое число и записать его цифры в стек. Вывести число, у которого цифры идут в обратном порядке. Цифры из записи числа считывать поочередно (например, с помощью getch()). Вывести число, которое получится, если поменять цифры стоящие на четных и нечетных позициях местами.

# КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое динамические структуры данных?
2. Какие виды динамических структур данных существуют?
3. Что такое стек?
4. Какие операции для работы со стеком необходимы?

# Список рекомендуемой литературы

## Основная литература

1. Павловская Т.А. C#. Программирование на языке высокого уровня:учеб. Для вузов.-м. [и др.]:Питер,2014.-432
2. Давыдова Н.А.,боровская Е.В. Программирование: учеб. Пособие. М.:Бином. Лаб. Знаний,2011.-238 с.
3. Булатицкий Д.И. Объектно-ориентированное программирование: учеб. пособие.- Брянск: изд-во БГТУ,2008.-120 с.

## Дополнительная литература

1. Тюльпинова Н.В. Алгоритмизация и программирование: учеб. Пособие/Брян. Гос. Техн. Ун-т.- Брянск:изд-во БГТУ,2012.-199 с.-[+ электронная копия].
2. Иванова Г.С. Программирование: основы алгоритмизации и процедурное программирование, объектно ориентированное программирование: учеб. Для вузов.-2-е изд., стер.-м.:Кнорус, 2014.-425 с.

Динамическая структура данных «стек». Реализация в связном списке [Текст] + [Электронный ресурс]: методические указания к выполнению лабораторной работы № 31 для студентов очной формы обучения по направлениям подготовки 09.03.04 – «Программная инженерия», 09.03.01 «Информатика и вычислительная техника», 02.03.03 «Математическое обеспечение и администрирование информационных систем». – Брянск, 2017. – 7 с.

БУЛАТИЦКИЙ ДМИТРИЙ ИВАНОВИЧ

КОПТЕНОК ЕЛИЗАВЕТА ВИКТОРОВНА

Научный редактор В.Я. Израилев

Компьютерный набор Е.В. Коптенок

Иллюстрации Е.В. Коптенок

Подписано в печать 20.02.2017. Формат 60х84 1/15 Бумага офсетная. Офсетная печать. Усл.печ.л. \_\_\_ Уч.-изд.л. \_\_\_ Тираж 1 экз. Бесплатно

Брянский государственный технический университет

Кафедра «Информатика и программное обеспечение», тел. 56-09-84

241035, Брянск, бульвар 50 лет Октября, 7 БГТУ, тел. 58-82-49