**Сопроводительный лист на издание в авторской редакции**

Название работы Динамическая структура данных «очередь» Методические указания к выполнению лабораторной работы № 32 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 «ОЧЕРЕДЬ»

**Методические указания**

**к выполнению лабораторной работы № 32**

**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 «очередь» [Текст] + [Электронный ресурс]: методические указания к выполнению лабораторной работы № 32 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 «очередь» и применить поученную библиотеку для решения задач.

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.

## Очередь (Queue, FIFO)

**Очередью** FIFO (First - In - First- Out - "первым пришел - первым исключается") называется такой последовательный список с переменной длиной, в котором включение элементов выполняется только с одной стороны списка (эту сторону часто называют **концом** или **хвостом очереди**), а исключение - с другой стороны (называемой **началом** или **головой очереди**) (Рис.2.1.).

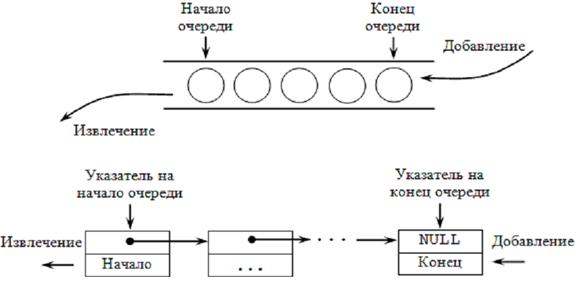
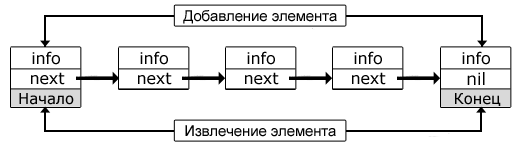


Рис.2.1. Динамическая структура «Очередь»

Основные операции над очередью - те же, что и над стеком - включение, исключение, определение размера, очистка, неразрушающее чтение.

## Дек (DEQ)

Дек - особый вид очереди. **Дек** (от англ. deq - double ended queue,т.е очередь с двумя концами) - это такой последовательный список, в котором как включение, так и исключение элементов может осуществляться с любого из двух концов списка (Рис.2.2.).



Ри.2.2. Динамическая структура «Дек»

# ХОД РАБОТЫ

Описание очереди выглядит следующим образом:

struct имя\_типа {

информационное поле;

адресное поле1;

адресное поле2;

};

* информационное поле – это поле любого, ранее объявленного или стандартного, типа;
* адресное поле1, адресное поле2 – это указатели на объекты того же типа, что и определяемая структура, в них записываются адреса первого и следующего элементов очереди.

Например, адресное поле ссылается на ранее объявленную структуру.

struct list1 {

type pole1;

list1 \*pole2;

}

struct ch3 {

list1 \*beg, \*next ;

}

Очередь как динамическую структуру данных легко организовать на основе линейного списка. Поскольку работа идет с обоими концами очереди, то предпочтительно будет использовать линейный двунаправленный список. Хотя для работы с таким списком достаточно иметь один указатель на любой элемент списка, здесь целесообразно хранить два указателя – один на начало списка (откуда извлекаем элементы) и один на конец списка (куда добавляем элементы). Если очередь пуста, то списка не существует, и указатели принимают значение NULL.

Описание элементов очереди аналогично описанию элементов линейного двунаправленного списка. Поэтому объявим очередь через объявление линейного двунаправленного списка:

struct Queue {

Double\_List \*Begin;//начало очереди

Double\_List \*End; //конец очереди

};

. . . . . . . . . .

Queue \*My\_Queue;//указатель на очередь

Основные операции, производимые с очередью:

* создание очереди;
* печать (просмотр) очереди;
* добавление элемента в конец очереди;
* извлечение элемента из начала очереди;
* проверка пустоты очереди;
* очистка очереди.

Реализацию этих операций приведем в виде соответствующих функций, которые, в свою очередь, используют функции операций с линейным двунаправленным списком.

//создание очереди

void Make\_Queue(int n, Queue\* End\_Queue){

Make\_Double\_List(n,&(End\_Queue->Begin),NULL);

Double\_List \*ptr; //вспомогательный указатель

ptr = End\_Queue->Begin;

while (ptr->Next != NULL)

ptr = ptr->Next;

End\_Queue->End = ptr;

}

//печать очереди

void Print\_Queue(Queue\* Begin\_Queue){

Print\_Double\_List(Begin\_Queue->Begin);

}

//добавление элемента в конец очереди

void Add\_Item\_Queue(int NewElem, Queue\* End\_Queue){

End\_Queue->End = Insert\_Item\_Double\_List(End\_Queue->End,

0, NewElem)->Next;

}

//извлечение элемента из начала очереди

int Extract\_Item\_Queue(Queue\* Begin\_Queue){

int NewElem = NULL;

if (Begin\_Queue->Begin != NULL) {

NewElem = Begin\_Queue->Begin->Data;

Begin\_Queue->Begin=Delete\_Item\_Double\_List(Begin\_Queue->Begin,0);

//удаляем вершину

}

return NewElem;

}

//проверка пустоты очереди

bool Empty\_Queue(Queue\* Begin\_Queue){

return Empty\_Double\_List(Begin\_Queue->Begin);

}

//очистка очереди

void Clear\_Queue(Queue\* Begin\_Queue){

return Delete\_Double\_List(Begin\_Queue->Begin);

}

На основе полученных функций необходимо сформировать библиотеку.

# ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ

**Задание.** Разработать библиотеку для работы с очередью. С помощью полученной библиотеки решить следующую задачу согласно варианту: Nзадания=Nвар%6+1:

1. Используя динамическую структуру "очередь", зашифровать содержимое текста: каждый символ заменить его кодом+1.
2. Сформировать очередь из элементов целого типа. Четные элементы возвести в квадрат. Распечатать исходную и результирующую очереди.
3. Используя динамическую структуру очередь, перевести введенную последовательность цифр в слово, состоящее из кодов ASCII.
4. Сформировать динамическую структуру «очередь», элементами которой являются цифры. Извлекая элементы из очереди, напечатать их двоичные эквиваленты.
5. Дан текстовый файл. Используя очередь, переписать содержимое его строк в новый текстовый файл, перенося при этом в конец каждой строки все входящие в нее цифры, сохраняя исходный порядок следования среди цифр и среди остальных символов строки.
6. Дан файл из целых чисел. Используя очередь, за один просмотр файла напечатать сначала все отрицательные числа, затем все положительные числа, сохраняя исходный порядок в каждой группе.

# КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое динамические структуры данных?
2. Какие виды динамических структур данных существуют?
3. Что такое очередь?
4. Какие операции для работы с очередью 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 «очередь» [Текст] + [Электронный ресурс]: методические указания к выполнению лабораторной работы № 32 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