**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ   
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ   
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА (МИИТ)»  
(РУТ (МИИТ)**

РОССИЙСКАЯ ОТКРЫТАЯ АКАДЕМИЯ ТРАНСПОРТА

Кафедра «Системы управления транспортной инфраструктурой»

Кафедра: «Системы управления транспортной инфраструктурой»

Авторы: Ридель В.В., д.физ.-мат.н., ст.н.с.

**АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ**

**Методические указания   
к выполнению лабораторных работ в интерактивной форме для студентов специальности**

*Направление/специальность:*  **09.03.03 Прикладная информатика**

*Профиль/специализация:*  **«Прикладная информатика в информационной сфере» (ИИ)**

*Квалификация выпускника:* **бакалавр**

*Форма обучения:* **заочная**

**Москва - 2025**

**Содержание**

Введение 3

Рекомендуемая литература 3

Требования к оформлению отчетов по лабораторным работам 3

Работа №1. Массивы автоматические, статические и динамические 3

Работа №2. Массивы и структурированные типы данных 7

Работа №3. Файлы 9

Работа №4. Очереди и стеки 11

Работа №5. Связные списки 12

Работа №6. Сортировка массивов 13

Работа №7. Сортировка списков 14

Работа №8. Поиск в массивах и списках 16

Работа №9. Двоичные деревья 17

# Введение

Первая часть лабораторного цикла включает в себя 9 работ, направленных на получение навыков использования основных структур данных и алгоритмов их обработки.

Лабораторные работы ориентированы на применение языка С++, хотя может быть использован и другой язык программирования.

# Рекомендуемая литература

1. Вирт Н. Алгоритмы и структуры данных.: Пер. С англ. - М.: Мир, 2001.

2. Топп У., Форд У. Структуры данных в С++. 1999.

3. Седжвик Р. Фундаментальные алгоритмы на C++. Части 1-4. Анализ. Структуры данных. Сортировка. Поиск. 2001.

4. Хэзфилд Р., Кирби Л. Искусство программирования на C. Фундаментальные алгоритмы, структуры данных и примеры приложений. 2001.

5. Подбельский В.В. Язык Си++. - М.: Финансы и статистика, 1995.

# Требования к оформлению отчетов по лабораторным работам

Отчет по лабораторной работе должен содержать название работы, предварительно составленный в соответствии с заданием текст программы, необходимый для допуска к выполнению работы, текст отлаженной работающей программы, результаты выполнения программы и выводы для тех пунктов задания, для которых это требуется.

Лабораторная работа №1

# Массивы автоматические, статические и динамические

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить темы «Типы данных», «Массивы», «Указатели», «Операции с динамической памятью».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Определить в функции main() переменные и массивы по таблице 1 в соответствии с вариантом:

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Переменные | № | Переменные |
| 1 | 1. Одномерный символьный массив;  2. Указатель на тип **char**;  3. Статический одномерный массив целых чисел;  4. Указатель на массив целых чисел;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. | 9 | 1. Одномерный массив плавающих чисел;  2. Указатель на тип **float**;  3. Статический одномерный массив беззнаковых целых чисел;  4. Указатель на массив **unsigned int**;  5. Трехмерный массив символов;  6. Указатель на двумерный массив символов. |
| 2 | 1. Одномерный массив целых чисел;  2. Указатель на тип **int**;  3. Статический одномерный символьный массив;  4. Указатель на массив символов;  5. Трехмерный массив плавающих чисел;  6. Указатель на двумерный массив плавающих чисел. | 10 | 1. Одномерный символьный массив;  2. Указатель на тип **char**;  3. Статический одномерный массив типа **double**;  4. Указатель на массив типа **double**;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. |
| 3 | 1. Одномерный символьный массив;  2. Указатель на тип **char**;  3. Статический одномерный массив длинных чисел;  4. Указатель на массив длинных целых чисел;  5. Трехмерный массив плавающих чисел;  6. Указатель на двумерный массив плавающих чисел. | 11 | 1. Одномерный массив целых чисел;  2. Указатель на тип **int**;  3. Статический одномерный массив плавающих чисел;  4. Указатель на массив плавающих чисел;  5. Трехмерный массив символов;  6. Указатель на двумерный массив символов. |
| 4 | 1. Одномерный массив длинных целых чисел;  2. Указатель на тип **long int**;  3. Статический одномерный массив символов;  4. Указатель на массив символов;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. | 12 | 1. Одномерный массив типа **double**;  2. Указатель на тип **double**;  3. Статический одномерный массив беззнаковых целых чисел;  4. Указатель на массив **unsigned int**;  5. Трехмерный массив символов;  6. Указатель на двумерный массив символов. |
| 5 | 1. Одномерный массив типа **float**;  2. Указатель на тип **float**;  3. Статический одномерный массив символов;  4. Указатель на массив символов;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. | 13 | 1. Одномерный массив беззнаковых целых чисел;  2. Указатель на тип **unsigned int**;  3. Статический одномерный массив символов;  4. Указатель на массив символов;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. |
| 6 | 1. Одномерный символьный массив;  2. Указатель на тип **char**;  3. Статический одномерный массив коротких целых чисел;  4. Указатель на массив коротких целых чисел;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. | 14 | 1. Одномерный массив типа **float**;  2. Указатель на тип **float**;  3. Статический одномерный массив символов;  4. Указатель на массив символов;  5. Трехмерный массив **unsigned int**;  6. Указатель на двумерный массив беззнаковых целых чисел. |
| 7 | 1. Одномерный массив коротких целых чисел;  2. Указатель на тип **short int**;  3. Статический одномерный массив символов;  4. Указатель на массив символов;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. | 15 | 1. Одномерный символьный массив;  2. Указатель на тип **char**;  3. Статический одномерный массив плавающих чисел;  4. Указатель на массив плавающих чисел;  5. Трехмерный массив целых чисел;  6. Указатель на двумерный массив целых чисел. |
| 8 | 1. Одномерный массив типа **double**;  2. Указатель на тип **double**;  3. Статический одномерный массив целых чисел;  4. Указатель на массив целых чисел;  5. Трехмерный массив символов;  6. Указатель на двумерный массив символов. | 16 | 1. Одномерный массив целых чисел;  2. Указатель на тип **int**;  3. Статический одномерный массив типа **double**;  4. Указатель на массив типа **double**;  5. Трехмерный массив символов;  6. Указатель на двумерный массив символов. |

2. В функции main() выполнить следующие действия:

1. Проверить содержимое массива №1 (с помощью цикла for и операции вывода cout<<).
2. Ввести данные в массив №1 (с помощью цикла for и операции ввода cin>>).
3. Еще раз проверить содержимое этого массива, сделать выводы.
4. Присвоить указателю №2 адрес массива №1, вывести на экран адреса массива и указателя и содержимое указателя. Сделать выводы.
5. Повторить пункт 3 для указателя, содержащего адрес массива. Сделать выводы.
6. Повторить пункты 1 – 3 для статического массива №3. Сделать выводы.
7. Используя имеющийся указатель №2, создать динамический массив и повторить для него пункты 1 – 3. Сделать выводы.
8. Удалить динамический массив. Используя указатель №4, создать двумерный динамический массив и повторить для него пункты 2, 3. Сделать выводы. Удалить двумерный динамический массив.
9. Вывести на экран любой из элементов трехмерного массива №5, используя операцию индексации.
10. Повторить пункт 9, используя имя массива как указатель и операцию доступа по указателю.
11. Присвоить указателю №6 на двумерный массив адрес трехмерного массива №5. Повторить для этого указателя пункт 10. Сделать выводы.

Текст программы следует дополнить сообщениями о вводе и выводе данных. Такие сообщения облегчат контроль выполнения программы. По мере выполнения работы отлаженные фрагменты текста можно закомментировать, предварительно занеся в отчет полученные результаты.

Сохранить файл с текстом программы для следующей работы.

**Контрольные вопросы.**

1. По каким признакам классифицируются структуры данных?
2. К какой группе структур данных относятся автоматические массивы?
3. Что означает понятие «тип данных»?
4. Какую информацию можно извлечь из типа данных?
5. К какой группе структур данных относятся статические массивы?
6. К какой группе структур данных относятся динамические массивы?
7. Что такое указатели?
8. Какие операции можно выполнять над указателями?
9. В чем заключается связь между указателями и массивами?
10. Какие операции обязательны при работе с динамическими массивами?
11. Свойства динамических массивов.
12. В чем заключается отличие между автоматическими и статическими массивами?
13. Можно ли изменить размер динамического массива при исполнении программы? Если да, то как это сделать?
14. Какое требование нужно соблюдать при присваивании адреса массива указателю?
15. Какие ограничения накладываются на определение многомерных динамических массивов?
16. В чем заключается отличие между именем массива и указателем?

Лабораторная работа №2

# Массивы и структурированные типы данных

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить темы «Массивы», «Указатели», «Структуры», «Операции с динамической памятью».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Определить структурный тип, содержащий следующие поля:

* символьный массив, используемый для хранения строки, например, с именем студента,
* указатель на тип **char** – для организации динамического массива, хранящего строку, например, с фамилией студента.

Остальные поля выбрать по варианту, приведенному в таблице 2.

Использовать одну из переменных для хранения некоторого идентификатора (номера); указатель на несимвольный тип – для организации динамического массива целых или плавающих чисел; другую переменную – для хранения размера этого массива;

Дополнить структурный тип любыми полями по своему выбору.

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
| № | Переменные | № | Переменные |
| 1 | 1. Переменная длинного целого типа;  2. Указатель на целый тип;  3. Переменная целого типа. | 9 | 1. Переменная плавающего типа;  2. Указатель на тип **float**;  3. Переменная беззнакового целого типа. |
| 2 | 1. Переменная целого типа;  2. Указатель на тип **int**;  3. Переменная плавающего типа. | 10 | 1. Переменная типа **double**;  2. Указатель на тип **double**;  3. Переменная целого типа. |
| 3 | 1. Переменная беззнакового длинного целого типа;  2. Указатель на длинный целый тип;  3. Переменная плавающего типа. | 11 | 1. Переменная целого типа;  2. Переменная плавающего типа;  3. Указатель на плавающий тип. |
| 4 | 1. Переменная длинного целого типа;  2. Указатель на тип **long int**;  3. Переменная целого типа. | 12 | 1. Переменная беззнакового целого типа;  2. Указатель на тип **unsigned int**;  3. Переменная символьного типа. |
| 5 | 1. Переменная типа **float**;  2. Указатель на тип **float**;  3. Переменная символьного типа. | 13 | 1. Переменная беззнакового целого типа;  2. Указатель на тип **unsigned int**;  3. Переменная целого типа. |
| 6 | 1. Переменная символьного типа;  2. Указатель на короткий целый тип;  3. Переменная короткого целого типа. | 14 | 1. Переменная типа **float**;  2. Указатель на тип **float**;  3. Переменная типа **unsigned int**. |
| 7 | 1. Переменная короткого целого типа;  2. Указатель на тип **short int**;  3. Переменная символьного типа. | 15 | 1. Переменная плавающего типа;  2. Указатель на плавающий тип;  3. Переменная целого типа. |
| 8 | 1. Переменная типа **double**;  2. Указатель на тип **double**;  3. Переменная символьного типа. | 16 | 1. Указатель на тип **int**;  2. Переменная типа **double**;  3. Переменная символьного типа. |

2. Определить функции:

* инициализации структуры;
* заполнения массива чисел;
* вывода на экран массива чисел;
* ввода информации в строки имени и фамилии и другие поля;
* вывода на экран всех полей структуры, кроме массива чисел;
* функцию освобождения динамической памяти.

У половины функций, по выбору студента, одним из аргументов должен быть указатель на структуру, у второй половины – ссылка на структуру.

3. Определить функцию main(), в которой создать:

* объект ранее определенного структурного типа
* указатель на этот структурный тип.

С помощью указателя создать динамический массив объектов структурного типа из 3-х – 4-х элементов.

Для объекта последовательно вызывать функции инициализации, заполнения массива чисел, ввода данных в остальные поля, показа массива, показа полей.

Для каждого элемента массива структур выполнить в цикле (for) функции инициализации, заполнения массива и ввода данных.

Вывести на экран содержимое полей каждого элемента массива структур в цикле (for) с помощью соответствующих функций.

В конце функции main() вызвать функцию освобождения памяти для объекта структурного типа и в цикле для каждого элемента массива объектов.

Удалить динамический массив.

Ход выполнения программы контролировать по выводимым на экран сообщениям.

Сохранить файл с текстом программы для следующей работы.

**Дополнительное задание.**

Создать в функции main() блок, в котором определить локальный объект структурного типа. Ввести данные в поля локального объекта.

Попытаться вывести на экран содержимое полей локального объекта за пределами блока. Сделать выводы.

**Контрольные вопросы.**

1. Что представляет собой структурный тип данных?
2. Данные каких типов могут входить в состав структур?
3. Данные каких типов не могут входить в состав структур?
4. По каким признакам классифицируются структуры данных?
5. К какой группе структур данных относятся автоматические массивы?
6. Что означает понятие «тип данных»?
7. Какую информацию можно извлечь из типа данных?
8. К какой группе структур данных относятся статические массивы?
9. К какой группе структур данных относятся динамические массивы?
10. Что такое указатели?
11. Какие операции можно выполнять над указателями?
12. В чем заключается связь между указателями и массивами?
13. Какие операции обязательны при работе с динамическими массивами?
14. Каковы свойства динамических массивов?
15. Как обеспечить связь между массивами или структурами и функциями?

Лабораторная работа №3

# Файлы

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить темы «Массивы», «Структуры», «Файлы».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Определить функции записи и считывания структуры для двоичного и текстового файлов. У всех функций одним из аргументов должна быть ссылка на структурный тип, у функций работы с двоичными файлами еще одним аргументом должен быть указатель на тип FILE.

Использовать структурный тип и функции ввода и вывода на экран данных, определенные в программе к лабораторной работе №2.

2. Определить глобальные указатели на тип FILE для функций работы с двоичными файлами.

3. Определить функцию main(), в которой создать:

* указатели на тип FILE для функций работы с текстовыми файлами;
* указатель на структурный тип для организации динамического массива структур;
* первый динамический массив структур небольшого размера (3 – 4 элемента) – для ввода данных и их сохранения в файлах;
* второй динамический массив структур такого же размера – для считывания данных из текстового файла;
* третий динамический массив структур – для считывания данных из двоичного файла.

4. Ввести данные в элементы первого динамического массива с помощью функций, разработанных в лабораторной работе №2.

5. Сохранить содержимое элементов массива структур в текстовом и двоичном файлах.

6. Считать содержимое текстового файла в элементы второго массива и вывести на экран. Сравнить содержимое первого и второго массивов.

7. Считать содержимое двоичного файла в элементы третьего массива и вывести на экран. Сравнить содержимое первого и третьего массивов.

8. Просмотреть содержимое двоичного и текстового файлов с помощью любого тестового редактора или программы-просмотрщика (для просмотра тестового файла можно использовать «Блокнот» Windows, для просмотра двоичного файла рекомендуется Norton Commander или аналогичная программа). Установить соответствие между элементами данных в первом массиве и текстовом файле. Попытаться сделать то же самое для двоичного файла.

Сохранить файл с тестом программы для последующих работ.

**Контрольные вопросы.**

1. Что такое файл?
2. К какой группе структур данных относятся файлы?
3. Какие действия необходимо выполнить для работы с файлом?
4. Различаются ли файлы по типам?
5. Как в программах устанавливается связь с файлами?
6. Какие способы организации связи с файлами вам известны?
7. Какие операции можно выполнять над файлами?
8. Как открыть файл для записи?
9. Как открыть файл для считывания?
10. Какая функция позволяет узнать длину файла?
11. Как проверить, можно ли произвести запись в выбранный файл?
12. Можно ли считать данные из произвольного места в файле? Если да, то как это сделать?
13. Можно ли перемещаться по файлу? Если да, то с помощью какой функции?
14. Чем отличается запись действительных чисел в текстовый и двоичный файлы?
15. Как обеспечить связь между файлами и функциями?

Лабораторная работа №4

# Очереди и стеки

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить темы «Очереди», «Стеки».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Определить массив, который будет использован для организации очереди и стека. Тип массива выбрать по таблице №1 – массив №1.

Определить структурный тип, используемый для представления элементов стека, организованного в виде динамической цепочки звеньев.

2. Разработать функции занесения и извлечения данных для простой очереди, циклической очереди, стека на основе массива и на основе связного списка. Обратить внимание на аргументы функций. Можно использовать перегрузку функций.

3. В функции main() проверить работу простой и циклической очередей и обоих вариантов организации стека. Для простой очереди и стека на основе массива добиться их переполнения. Сделать то же самое для циклической очереди. Сделать выводы.

4. Проконтролировать при операциях занесения и извлечения данных изменение индексов записи и считывания для простой и циклической очередей. Проконтролировать изменение индекса вершины стека для массива и адреса вершины стека для списка. Сделать выводы.

Сохранить файл с тестом программы для последующих работ.

**Дополнительное задание.**

Разработать функции занесения и извлечения данных для приоритетной очереди. Проверить работу приоритетной очереди с помощью этих функций.

**Рекомендации к выполнению работы.**

Функция main() может содержать цикл, например, с постусловием, внутри которого может происходить опрос клавиатуры и выполнение вставки в очередь (стек) или извлечение, в зависимости от нажатой клавиши.

**Контрольные вопросы.**

1. Что представляет собой очередь?
2. Что представляет собой стек?
3. Какие известны виды очередей?
4. На основе каких структур данных могут организовываться стеки?
5. Какие операции допустимы для очередей?
6. Какие операции допустимы для стеков?
7. Какой характер имеет операция считывания для очередей и стеков?
8. Какими свойствами обладают очереди?
9. Каким недостатком обладает простая очередь? Каков способ борьбы с этим недостатком?
10. Чем отличается циклическая очередь от простой?
11. Чем отличается стек на основе массива от стека на основе связного списка?
12. Чем отличается приоритетная очередь от простой?
13. Чем отличается стек на основе связного списка от собственно связного списка?
14. К каким структурам данных относятся очереди и стеки?
15. Что такое уровни представления структур данных?
16. Пояснить различие между логическим и физическим уровнями представления структур данных на примере очередей или стеков.
17. Каковы области применения очередей и стеков?
18. Что представляет собой дек?

Лабораторная работа №5

# Связные списки

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить тему «Связные списки».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Для организации односвязного списка определить структурный тип, содержащий указатель на свой тип и поля, использованные в работе №2.

2. Определить функции вставки нового звена в односвязный линейный список, удаления звена из списка, просмотра содержимого списка.

3. В функции main() создать заглавное звено списка и проверить работу функций вставки, удаления и просмотра. В функции просмотра предусмотреть вывод адресов каждого звена списка. Сделать выводы.

4. Преобразовать односвязный список в двусвязный. Внести соответствующие изменения в функции работы со списком. Проверить работу функций.

5. Преобразовать линейный список в кольцевой. Внести необходимые изменения в функции работы со списком. Проверить работу функций. Сделать выводы.

Сохранить файл с тестом программы для последующих работ.

**Рекомендации к выполнению работы.**

Функция main() может содержать цикл, похожий на тот, который использован в работе №4. В этом случае вызов функций вставки, удаления и просмотра списка будет выполняться аналогично вызову соответствующих функций для очереди или стека.

Для упрощения создания программы можно использовать программу из предыдущей работы, внеся в нее необходимые изменения и удалив ненужные фрагменты.

**Контрольные вопросы.**

1. Что представляют собой списки?
2. К каким классификационным группам структур данных относятся списки?
3. Какие существуют разновидности списков?
4. В чем состоит отличие несвязного списка от массива?
5. В чем состоит отличие связного списка от массива?
6. В чем состоит отличие линейного списка от кольцевого?
7. В чем заключаются недостатки односвязного списка?
8. В чем состоит отличие односвязного списка от двусвязного?
9. Какие операции применяются для связных списков?
10. В чем отличие считывания информации из списка от считывания из очереди или стека?
11. Особенности операций вставки и удаления для связных списков.
12. В чем отличие операции вставки в двусвязный список от вставки в односвязный список?
13. В чем отличие операции удаления из двусвязного списка от удаления из односвязного списка?
14. В чем заключаются особенности работы с кольцевыми списками?
15. Что означает понятие «динамическая структура данных»?
16. Какой тип должно иметь звено связного списка? Почему?
17. Что обязательно должно содержать звено связного списка?
18. В чем состоит отличие звена двусвязного списка от звена односвязного списка?
19. В чем состоит отличие связного списка от стека, организованного в виде связного списка?

Лабораторная работа №6

# Сортировка массивов

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить тему «Сортировка».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Определить массив, элементы которого будут упорядочиваться. Тип массива выбрать по таблице №1 – массив №1.

2. Разработать функцию сортировки массива методами, выбранными по таблице 3 в соответствии с вариантом.

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| № | Алгоритмы сортировки | № | Алгоритмы сортировки |
| 1 | 1). Пузырьковая; 2). Быстрая. | 9 | 1). Быстрая; 2). Отбором. |
| 2 | 1). Отбором; 2). Шелла. | 10 | 1). Шелла; 2). Вставками. |
| 3 | 1). Вставками; 2). Быстрая. | 11 | 1). Пузырьковая; 2). Отбором. |
| 4 | 1). Пузырьковая; 2). Шелла. | 12 | 1). Быстрая; 2). Пузырьковая. |
| 5 | 1). Отбором; 2). Быстрая. | 13 | 1). Шелла; 2). Отбором. |
| 6 | 1). Вставками; 2). Шелла. | 14 | 1). Вставками; 2). Пузырьковая. |
| 7 | 1). Отбором; 2). Пузырьковая. | 15 | 1). Быстрая; 2). Вставками. |
| 8 | 1). Пузырьковая; 2). Вставками. | 16 | 1). Шелла; 2). Пузырьковая. |

3. Любым способом заполнить элементы массива значениями.

4. Выполнить сортировку массива первым алгоритмом и проконтролировать ее результат. Проверить все варианты исходного заполнения массива: случайным образом, отсортированного в обратном порядке, отсортированного в требуемом порядке. Убедиться в правильности сортировки во всех случаях. Сделать выводы.

5. Повторить пункты 3 и 4 для второго алгоритма сортировки.

Сохранить файл с тестом программы для последующих работ.

**Контрольные вопросы.**

1. Что такое «сортировка»?
2. Сколько существует групп алгоритмов сортировки?
3. Сколько существует алгоритмов сортировки?
4. По каким признакам характеризуются алгоритмы сортировки?
5. Что нужно учитывать при выборе алгоритма сортировки?
6. Какой алгоритм сортировки считается самым простым?
7. Какой алгоритм сортировки считается самым эффективным?
8. Что означает понятие «скорость сортировки»?
9. В чем заключается метод пузырьковой сортировки?
10. В чем заключается метод сортировки отбором?
11. В чем заключается метод сортировки вставками?
12. В чем заключается метод сортировки разделением?
13. В чем заключается метод быстрой сортировки?
14. В чем заключается метод сортировки Шелла?
15. В чем заключается метод сортировки Бэтчера?
16. Как зависит скорость сортировки от размера структуры данных для разных алгоритмов?
17. Почему метод Бэтчера называется параллельным?

Лабораторная работа №7

# Сортировка списков

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить тему «Сортировка», «Списки».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Разработать функции сортировки списков методами, выбранными по таблице 4 в соответствии с вариантом.

Таблица 4

|  |  |  |  |
| --- | --- | --- | --- |
| № | Алгоритмы сортировки | № | Алгоритмы сортировки |
| 1 | Линейный односвязный список, содержащий целые числа.  1). Отбором с процедурами вставки и удаления;  2). Вставками. | 9 | Кольцевой двусвязный список, содержащий символы.  1). Пузырьковая с заменой указателей; 2). Отбором с процедурами вставки и удаления. |
| 2 | Кольцевой односвязный список, содержащий символы.  1). Вставками; 2). Отбором с заменой указателей. | 10 | Кольцевой односвязный список, содержащий данные типа **float**.  1). Пузырьковая с заменой указателей; 2). Вставками. |
| 3 | Линейный односвязный список, содержащий данные типа **float**.  1). Пузырьковая с процедурами вставки и удаления;  2). Отбором с заменой указателей. | 11 | Кольцевой двусвязный список, содержащий целые числа.  1). Отбором с процедурами вставки и удаления;  2). Пузырьковая с переносом данных. |
| 4 | Линейный двусвязный список, содержащий целые числа.  1). Вставками; 2). Пузырьковая с переносом данных. | 12 | Линейный односвязный список, содержащий данные типа **double**.  1). Пузырьковая c процедурами вставки и удаления; 2). Отбором c процедурами вставки и удаления. |
| 5 | Линейный двусвязный список, содержащий символы.  1). Вставками;  2). Пузырьковая с заменой указателей. | 13 | Линейный двусвязный список, содержащий данные типа **double**.  1). Вставками; 2). Пузырьковая c процедурами вставки и удаления. |
| 6 | Линейный односвязный список, содержащий символы.  1). Отбором с заменой указателей; 2). Пузырьковая с переносом данных. | 14 | Кольцевой односвязный список, содержащий данные типа **double**.  1). Отбором с заменой указателей; 2). Вставками. |
| 7 | Кольцевой односвязный список, содержащий целые числа.  1). Пузырьковая с переносом данных; 2). Вставками. | 15 | Кольцевой двусвязный список, содержащий данные типа **double**.  1). Вставками; 2). Пузырьковая с заменой указателей. |
| 8 | Линейный двусвязный список, содержащий данные типа **float**.  1). Отбором с заменой указателей; 2). Пузырьковая с заменой указателей. | 16 | Кольцевой двусвязный список, содержащий данные типа **float**.  1). Вставками; 2). Отбором с заменой указателей. |

2. Ввести информацию в элементы списка.

3. Выполнить сортировку списка первым алгоритмом и проконтролировать ее результат. Проверить все варианты исходного заполнения списка: случайным образом, отсортированного в обратном порядке, отсортированного в требуемом порядке. Убедиться в правильности сортировки во всех случаях. Сделать выводы.

4. Повторить пункты 2 и 3 для второго алгоритма сортировки. Для ввода данных в существующие элементы списка может потребоваться соответствующая функция (редактирования элемента). Ее необходимо разработать самостоятельно.

Сохранить файл с тестом программы для последующих работ.

**Контрольные вопросы.**

1. Что такое «сортировка»?
2. Сколько существует алгоритмов сортировки?
3. Сколько существует групп алгоритмов сортировки?
4. Чем сортировка списка отличается от сортировки массива?
5. Что нужно учитывать при выборе алгоритма сортировки?
6. По каким признакам характеризуются алгоритмы сортировки?
7. Какой алгоритм сортировки считается самым эффективным?
8. Какой алгоритм сортировки считается самым простым?
9. В чем заключается метод сортировки вставками?
10. В чем заключается метод сортировки отбором?
11. В чем заключается метод пузырьковой сортировки?
12. В чем заключается пузырьковая сортировка списка с использованием процедур вставки в список и удаления?
13. В чем заключается пузырьковая сортировка списка с переносом данных между звеньями?
14. В чем заключается пузырьковая сортировка списка с заменой значений указателей?
15. В чем заключается сортировка списка отбором с использованием процедур вставки в список и удаления?
16. В чем заключается сортировка списка отбором с заменой значений указателей?
17. В чем заключается метод сортировки списка вставками?
18. Что означают термины «внешняя сортировка» и «внутренняя сортировка»?

Лабораторная работа №8

# Поиск в массивах и списках

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить тему «Поиск», «Массивы», «Списки».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Определить массив и список, в которых будет выполняться поиск. Использовать массив, созданный в работе №6, и список, созданный в работе №7.

2. Разработать функции линейного поиска в массиве, двоичного поиска и поиска в списке. Для обеспечения двоичного поиска в массиве использовать одну из функций сортировки массива, созданных в работе №6.

3. Ввести информацию в массив и список.

4. Выполнить линейный поиск в массиве.

5. Упорядочить элементы массива функцией сортировки и выполнить двоичный поиск.

6. Выполнить поиск в связном списке.

Для всех случаев проверить варианты успешного и неуспешного поиска. Сделать выводы.

Сохранить файл с тестом программы для последующих работ.

**Контрольные вопросы.**

1. Что такое поиск?
2. Что называется ключом поиска?
3. Какие известны методы поиска?
4. Какой алгоритм поиска является наиболее эффективным?
5. Какое требование предъявляется к структуре данных, в которой выполняется двоичный поиск?
6. Чем отличается поиск в массиве от поиска в списке?
7. Чем отличаются процедуры поиска в односвязном и двусвязном списках?
8. В чем заключается метод линейного поиска?
9. В чем заключается метод двоичного поиска?
10. В чем заключается поиск в списке?
11. Почему двоичный поиск получил такое название?
12. Какие известны варианты двоичного поиска?
13. Пригоден ли двоичный метод для поиска данных в неупорядоченной структуре?
14. Какой из методов поиска данных в массиве является более универсальным?
15. Существуют ли какие-нибудь недостатки у линейного поиска? Если да, то какие?

Лабораторная работа №9

# Двоичные деревья

**Подготовка к работе.** По указанной литературе и конспекту лекций повторить тему «Двоичные деревья».

Разработать программу в соответствии с заданием к лабораторной работе.

**Задание.**

1. Для организации двоичного дерева определить соответствующий структурный тип, содержащий поля, использованные в работе №2. Для хранения ключа использовать любое несимвольное поле.

2. Разработать функции вставки узла в дерево, удаления узла из дерева, прохождения дерева в восходящем, нисходящем, последовательном порядке, создания идеально сбалансированного дерева с заданным числом узлов, поиска в дереве.

3. Создать двоичное дерево с помощью функции вставки узла.

4. Выполнить просмотр дерева в восходящем, нисходящем и последовательном порядке с помощью соответствующих функций.

5. Удалить несколько узлов из дерева, проверяя состояние дерева после каждого удаления с помощью просмотра в нисходящем порядке.

6. Создать идеально сбалансированное дерево и выполнить его просмотр в нисходящем порядке.

7. Дополнить идеально сбалансированное дерево несколькими узлами и проверить его состояние, просматривая дерево в нисходящем порядке.

8. Выполнить поиск информации в дереве. Проверить варианты успешного и неуспешного поиска.

**Контрольные вопросы.**

1. Что представляют собой древовидные структуры данных?
2. Какие существуют виды деревьев?
3. Что представляет собой двоичное дерево?
4. Чем отличается двоичное дерево от двусвязного списка?
5. Что означает термин «вырожденное дерево»?
6. Чем вырожденное дерево отличается от односвязного списка?
7. Что означает термин «идеально сбалансированное дерево»?
8. В чем заключается особенность дерева как структуры данных?
9. Каковы области применения древовидных структур данных?
10. Процедуры какого характера наиболее эффективны при работе с деревьями?
11. В чем заключается вставка узла в дерево?
12. В чем заключается удаление узла из дерева?
13. Каковы особенности удаления элемента из древовидной структуры данных?
14. В чем заключается поиск в дереве?
15. Что такое «прохождение дерева»? Какие возможны варианты прохождения дерева?
16. Что такое высота дерева?
17. Как сохранить сбалансированность дерева при вставке и удалении узлов?