МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательноеучреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра информационных систем и технологий

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_5_» _06_ 2019 г.

СТРУКТУРЫ И АЛГОРИТМЫ КОМПЬЮТЕРНОЙ ОБРАБОТКИ ДАННЫХ

Методические рекомендации по выполнению лабораторных работ для студентов направления подготовки 02.03.03 «Математическое обеспечение и администрирование информационных систем»

Курск 2019

УДК 004
Составитель С.Ю. Сазонов, Е.А. Кулешова

Рецензент
Кандидат технических наук, доцент Ю.А. Халин

Структуры и алгоритмы компьютерной обработки данных: методические рекомендации по выполнению лабораторных работ для студентов направления подготовки 02.03.03 «Математическое обеспечение и администрирование информационных систем» / Юго-Зап. гос. ун-т; сост. С.Ю. Сазонов, Е.А. Кулешова. Курск, 2019. 30с. Библиогр.: с.30.

В методических рекомендациях приведен перечень различных методов поддержки принятия решений довольно широк и включает в себя методы выявления проблем, генерации, выбора и оценки альтернатив, а также методы реализации и оценки качества и эффективности последствий принятия решений. Многие из этих методов являются достаточно сложными для понимания их сущности неподготовленными пользователями, что влечет за собой либо нежелание их применения на практике, либо затрудняет их практическое использование.

Методические рекомендации предназначены для студентов, обучающихся по направлению подготовки 02.03.03 «Математическое обеспечение и администрирование информационных систем».

Текст печатается в авторской редакции.

Подписано в печать 5.03.19 . Формат 60x84 1/16
Усл.печ.л. 1,7. Уч.-изд.л. 1,6. Тираж 100 экз.
Заказ 187 . Бесплатно.
Юго-Западный государственный университет.
305040, г.Курск, ул. 50 лет Октября, 94

Содержание

Введение 4

Лабораторная работа №1Массивы автоматические, статические и динамические 5

Лабораторная работа №2 Массивы и структурированные типы данных 9

Лабораторная работа №3 Файлы 12

Лабораторная работа №4 Очереди и стеки 14

Лабораторная работа №5 Связные списки 16

Лабораторная работа №6Сортировка массивов 18

Лабораторная работа №7 Сортировка списков 20

Лабораторная работа №8 Исследование эффективности алгоритмов сортировки для массивов малого размера 22

Лабораторная работа №9 Поиск в массивах и списках 26

Лабораторная работа №10 Деревья 28

Библиографический список 30

Введение

Лабораторный цикл включает в себя 10 работ, направленных на получение навыков работы с основными структурами данных и алгоритмами их обработки.

Лабораторные работы ориентированы на применение языка C++, хотя может ыть использован и любой другой язык программирования.

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

Таблица 1

№	Переменные	№	Переменные
1	1. Одномерный символьный ив; 2. Указатель на тип char; 3. Статический одномерный ив целых чисел; 4. Указатель на массив целых ; 5. Трехмерный массив целых ; 6. Указатель на двумерный ив целых чисел.	9	1. Одномерный массив ющих чисел; 2. Указатель на тип float; 3. Статический одномерный ив беззнаковых целых чисел; 4. Указатель на массив unsigned 5. Трехмерный массив символов; 6. Указатель на двумерный массив олов.
2	1. Одномерный массив целых ; 2. Указатель на тип int; 3. Статический одномерный ольный массив; 4. Указатель на массив олов; 5. Трехмерный массив ющих чисел; 6. Указатель на двумерный ив плавающих чисел.	10	1. Одномерный символьный ив; 2. Указатель на тип char; 3. Статический одномерный ив типа double; 4. Указатель на массив типа е; 5. Трехмерный массив целых ; 6. Указатель на двумерный массив х чисел.
3	1. Одномерный символьный ив; 2. Указатель на тип char; 3. Статический одномерный ив длинных чисел; 4. Указатель на массив ных целых чисел; 5. Трехмерный массив ющих чисел; 6. Указатель на двумерный ив плавающих чисел.	11	1. Одномерный массив целых ; 2. Указатель на тип int; 3. Статический одномерный ив плавающих чисел; 4. Указатель на массив ющих чисел; 5. Трехмерный массив символов; 6. Указатель на двумерный массив олов.
4	1. Одномерный массив длинных х чисел; 2. Указатель на тип long int; 3. Статический одномерный ив символов; 4. Указатель на массив олов; 5. Трехмерный массив целых ; 6. Указатель на двумерный ив целых чисел.	12	1. Одномерный массив типа е; 2. Указатель на тип double; 3. Статический одномерный ив беззнаковых целых чисел; 4. Указатель на массив unsigned 5. Трехмерный массив символов; 6. Указатель на двумерный массив олов.

5	1. Одномерный массив типа float; 2. Указатель на тип float; 3. Статический одномерный символов; 4. Указатель на массив символов; 5. Трехмерный массив целых 6. Указатель на двумерный целых чисел.	13	1. Одномерный массив беззнаковых чисел; 2. Указатель на тип unsigned int; 3. Статический одномерный массив ов; 4. Указатель на массив символов; 5. Трехмерный массив целых чисел; 6. Указатель на двумерный массив чисел.
6	1. Одномерный символьный ; 2. Указатель на тип char; 3. Статический одномерный коротких целых чисел; 4. Указатель на массив коротких чисел; 5. Трехмерный массив целых 6. Указатель на двумерный целых чисел.	14	1. Одномерный массив типа float; 2. Указатель на тип float; 3. Статический одномерный массив ов; 4. Указатель на массив символов; 5. Трехмерный массив unsigned int; 6. Указатель на двумерный массив ковых целых чисел.
7	1. Одномерный массив коротких чисел; 2. Указатель на тип short int; 3. Статический одномерный символов; 4. Указатель на массив символов; 5. Трехмерный массив целых 6. Указатель на двумерный целых чисел.	15	1. Одномерный символьный массив; 2. Указатель на тип char; 3. Статический одномерный массив ющих чисел; 4. Указатель на массив плавающих 5. Трехмерный массив целых чисел; 6. Указатель на двумерный массив чисел.
8	1. Одномерный массив типа ; 2. Указатель на тип double; 3. Статический одномерный целых чисел; 4. Указатель на массив целых 5. Трехмерный массив символов; 6. Указатель на двумерный символов.	16	1. Одномерный массив целых чисел; 2. Указатель на тип int; 3. Статический одномерный массив ouble; 4. Указатель на массив типа double; 5. Трехмерный массив символов; 6. Указатель на двумерный массив ов.

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. Ограничения, накладываемые 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 структуру (структурный тип), содержащую поля, определяемые вариантом, приведенным в табл. 1. Заменить все двумерные и трехмерные массивы и указатели одномерными. Заменить все несимвольные массивы на переменные тех же типов.

Символьный массив использовать для хранения строки, например, с именем студента, указатель на тип **char** - для организации динамического массива, хранящего строку, например, с фамилией студента.

Использовать одну из переменных для хранения некоторого идентификатора (номера); один из указателей 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.

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. 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 удаления 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 шаш() 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 отличие несвязного списка от массива?
5. В чем отличие связного списка от массива?
6. В чем отличие линейного списка от кольцевого?
7. В чем заключаются недостатки односвязного списка?
8. В чем 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. 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 отличие звена двусвязного списка от звена односвязного списка?
19. В чем 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 2 в соответствии с вариантом из таблицы 2

Таблица 2

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

3. Любым способом заполнить элементы массива значениями.

4. Выполнить сортировку массива первым алгоритмом и проконтролировать ее результат. 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 функцию сортировки списка методами, выбранными по таблице 3 в соответствии с вариантом. Для организации списка использовать структурный тип и функции работы со списками, созданные в работе №5.

Таблица 3

№	Алгоритмы	№	Алгоритмы сортировки
1	1). Отбором; 2).	9	1). Шелла; 2).
2	1). Вставками; 2).	1	1). Пузырьковая; 2).
3	1). Пузырьковая; 2).	1	1). Быстрая; 2).
4	1). Отбором; 2).	1	1). Шелла; 2). Отбором.
5	1). Вставками; 2).	1	1). Вставками; 2).
6	1). Отбором; 2).	1	1). Быстрая; 2).
7	1). Пузырьковая; 2).	1	1). Шелла; 2).
8	1). Шелла; 2).	1	1). Быстрая; 2).

2. Ввести информацию в элементы списка.

3. Выполнить сортировку списка первым алгоритмом и проконтролировать ее результат. Сделать выводы.

4. Повторить пункты 2 и 3 для второго алгоритма сортировки. Для ввода данных в существующие элементы списка может протребоваться 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 разделением?
10. В чем заключается метод быстрой сортировки?
11. В чем заключается метод сортировки Шелла?
12. В чем заключается метод сортировки Бэтчера?
13. В чем заключается метод сортировки вставками?
14. В чем заключается метод сортировки отбором?
15. В чем заключается метод пузырьковой сортировки?
16. Что означают термины «внешняя сортировка» и «внутренняя сортировка»?

Лабораторная работа №8

Исследование эффективности алгоритмов сортировки для массивов малого размера

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

Разработать программу в соответствии с заданием к лабораторной работе.

Пояснения к выполнению работы.

В задачах цифровой обработки сигналов часто требуется выполнить упорядочение массивов малого размера - от 3 до 9 элементов, реже 25 и более элементов. Как известно, наиболее эффективным универсальным методом для массивов большого размера в среднем случае является алгоритм быстрой сортировки. Этот метод достаточно сложен с алгоритмической точки зрения и может оказаться не столь эффективным для небольших массивов. В работе требуется сравнить время сортировки массива алгоритмами сортировки отбором, вставками, быстрой и пузырьковой.

Определить время выполнения какого-либо фрагмента программы можно с помощью библиотечной функции **gettimeO,** прототип которой находится в заголовочном файле **dos.h.** Эта функция должна получать аргумент - адрес структуры типа **time,** в поля которой и заносится информация о текущем времени. В обобщенном виде программа, в которой выполняется контроль времени может выглядеть следующим образом:

```
#include <dos.h> void main()
{
time tl,t2; gettime(&tl);
// Фрагмент, время работы которого измеряется.
gettime(&t2);
И ...
}
```

Для получения интервала времени следует вычесть поля структуры tl из полей структуры **t2.** Представить этот интервал в секундах можно следующим образом:

double delta_t={t2.ti_hour*360000.+t2.ti_min*6000.+ t2.ti_sec*100.+t2.ti_hund-(tl.ti hour*360000.+tl.ti min*6000.+

Современные процессоры выполняют сортировку небольших массивов за время, значительно меньшее, чем тысячная доля секунды (tl.ti_hund), поэтому для получения точного результата нужно выполнить сортировку достаточно большое число раз, например, миллион. При этом необходимо каждый раз сортировать один и тот же неупорядоченный массив. Лучше сортировать второй массив, каждый раз перед сортировкой копируя в него данные из исходного неупорядоченного массива. Копирование занимает некоторое время, которое необходимо учесть.

После завершения всего процесса следует проверить правильность сортировки, выведя содержимое второго массива на экран.

Итоговая программа в обобщенном виде может выглядеть следующим образом:

```
#include <dos.h>
#include <iostream.h>
const int N = 5; // Размер массива.
const unsigned long NN = 1000000; // Число сортировок, void main()
{
char arrl[N]={ /* Элементы неупорядоченного массива */
};
char arr[N];
time tl,t2;
double t_copy,t_sort;

gettime(&tl);
// Копирование из массива arrl в массив arr. gettime(&t2);
```

t_copy = (t2.ti_hour*360000.+t2.ti_min*6000.+ t2.ti_sec*100.+t2.ti_hund-(tl.ti_hour*360000.+ tl.ti_min*6000.+tl.ti_sec*100.+tl.ti_hund))/100.;

gettime(&tl);

// Цикл сортировки из большого числа проходов. gettime(&t2);

t_sort = (t2.ti_hour*360000.+t2.ti_min*6000.+ t2.ti_sec*100.+t2.ti_hund-(tl.ti_hour*360000.+ tl.ti_min*6000.+tl.ti_sec*100.+tl.ti_hund))/100.; t_sort-=t_copy;

// Контроль содержимого массива arr после сортировки.

// Вывод на экран времени сортировки t_sort.

}

Для повышения точности результата следует вынести за пределы цикла сортировки все вспомогательные операции. С этой же целью нужно выбирать число проходов цикла достаточно большим. На процессоре AMD K7 с частотой 700 МГц для символьного массива из 5 элементов 100 миллионов циклов сортировки отбором выполняются за время около 40 секунд, пузырьковой - 80 сек, вставками - 60 сек, быстрой - 100 сек. Для сокращения времени сортировки и получения результата менее точного, но достаточного для качественной оценки алгоритмов, можно уменьшить число проходов цикла.

Следует сравнить эффективность алгоритмов для худшего случая - массив упорядочен в обратном направлении, — и случайного расположения элементов в массиве, а также для массивов разного размера - 5, 7, 9 и 25 элементов.

Задание.

1. Определить массив из 5 элементов и заполнить его значениями, упорядоченными по убыванию. Определить второй массив такого же типа и размера, не инициализированный значениями.
2. Выполнить многократное копирование данных из первого массива во второй и сортировку второго массива методом отбора в соответствии с пояснениями к работе. Можно использовать функцию сортировки, созданную в работе № 6.

3. Зафиксировать время, затраченное на сортировку.
4. Проконтролировать правильность работы алгоритма выводом массива на экран.
5. Повторить пункты 2-4 для сортировки вставками, быстрой и пузырьковой.
6. Повторить пункты 1-5 для массива размером 7, 9 и 25 элементов.
7. Повторить пункты 1-6 для массива заполненного случайным образом.
8. Представить результаты работы в виде графика зависимости времени сортировки от размера массива для разных алгоритмов. Сделать выводы.

Контрольные вопросы.

1. В чем заключается метод сортировки отбором?
2. В чем заключается метод сортировки вставками?
3. В чем заключается метод пузырьковой сортировки?
4. В чем заключается метод быстрой сортировки?
5. Какой алгоритм сортировки считается самым простым?
6. Какой алгоритм сортировки считается самым эффективным?
7. Сколько существует алгоритмов сортировки?
8. Сколько существует групп алгоритмов сортировки?
9. По каким признакам характеризуются алгоритмы сортировки?
10. Что нужно учитывать при выборе алгоритма сортировки?
11. Что такое «сортировка»?
12. Как зависит скорость сортировки от размера структуры данных для разных алгоритмов?
13. Чем сортировка списка отличается от сортировки массива?
14. Почему метод Бэтчера называется параллельным?
15. Предложите другой способ решения задачи, рассматриваемой в этой работе.

Лабораторная работа №9

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 №10
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 соответсвующих 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?

Библиографический список

1. Введение в математическое моделирование [Текст] : учебное пособие / В. Н. Ашихмин [и др.] ; под ред. П. В. Трусова. - Москва : Логос : Университетская книга, 2015. - 440 с.
2. Системы поддержки принятия решений [Текст] : учебник и практикум для бакалавриата и магистратуры / Санкт-Петербургский гос. ун-т; под ред. В. Г. Халина, Г. В. Черновой. - Москва : Юрайт, 2016. - 494 с. - (Бакалавр и магистр. Академический курс).
3. Горелик, В. А. Теория принятия решений [Электронный ресурс : учебное пособие для магистрантов. – Москва : МПГУ, 2016 – 152 с. - Режим доступа: biblioclub.ru
4. Ногин, В. Д. Принятие решений в многокритериальной среде. Количественный подход [Текст] / В. Д. Ногин. – М. : ФИЗМАТЛИТ, 2002. -176 с.
5. Юдин, Д. Б. Вычислительные методы теории принятия решений [Текст] : монография / Д. Б. Юдин. - Москва : URSS ; Москва : Либроком, 2014. - 318 с.
6. Соловьев, Н. Основы теории принятия решений для программистов [Электронный ресурс] : учебное пособие / Н. Соловьев , Е. Чернопрудова, Д. А. Лесовой. - Оренбург : ОГУ, 2012. - 187 с. - Режим доступа: biblioclub.ru
7. Власов, Марк Павлович. Моделирование экономических систем и процессов [Текст] : учебное пособие / М. П. Власов, П. Д. Шимко. – Москва : Инфра-М, 2013. - 336 с.
8. Структуры и алгоритмы компьютерной обработки данных [Электронный ресурс] : методические указания к выполнению курсовых работ / Юго-Зап. гос. ун-т; сост. С.Ю. Сазонов., Е.А. Кулешова – Курск : ЮЗГУ, 2019. - 11 с.
9. Самостоятельная работа студента [Электронный ресурс] : методические указания к самостоятельной работе для студентов направлений 09.03.02, 09.03.03, 02.03.03, 38.03.05, 02.04.03, 09.04.02, 09.04.03, 38.04.05 / Юго-Зап. гос. ун-т; сост. С. Ю. Сазонов. – Курск: ЮЗГУ, 2018. - 35 с.