Лабораторная работа № 1

**Динамические структуры данных, работа с файлами**

**Цель работы:** Закрепить практические навыки, полученные при изучении дисциплины «Основы программирования». Решение задач с использованием динамических структур, работа с файлами.

**Теоретическая подготовка**

Лабораторная работа № 1 представляет собой повторение практических навыков, изучаемых в предыдущем семестре, поэтому новых теоретических материалов не приводится. При подготовке к лабораторной работе смотрите материалы, приведенные в папке «Самостоятельная работа», а также разработки к лабораторным работам и БДЗ прошлого семестра.

**Общие требования к выполнению заданий**

1. Во всех вариантах нужно создать динамические структуры, организованные в виде массива или списка (по своему выбору)
2. Должна быть статическая переменная для учета общего числа созданных элементов данных.
3. Данные размещаются в динамической памяти.
4. Обязательные функции для всех вариантов:

* добавить новый элемент
* распечатка данных в табличном виде
* выход из программы

1. Остальные функции для работы с данными указаны в задании индивидуально.
2. Для выполнения функций, указанных в задании, написать диалоговый интерфейс, позволяющий выполнять функции в произвольном порядке многократно
3. При выполнении функции «выход из программы» нужно сохранить базу на диске
4. Первичное создание базы – ввод данных с клавиатуры
5. Если программа уже запускалась, то данные загружаются из файла перед выходом на диалог. Иными словами вносятся изменения и дополнения в уже существующую базу данных.
6. При реализации функций в параметрах и возвращаемых значениях использовать указатели и ссылки

**Пример диалогового интерфейса**

**(База данных «Склад товаров»)**

Добавить новый элемент …… 1

Распечатать базу товаров …… 2

Поиск товара по названию …… 3

Фильтр по цене ………….. 4

Выход из программы ………….. 5

……………………………………..

Введите номер функции

**Пример распечатки данных в табличном виде**

(**«Склад товаров»)**

**Название Цена Количество Общая сумма**

**Товара (руб) (кг) (руб)**

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

Сыр «Российский» 560 26.5 14840.00

Масло сливочное 380.5 100.25 38145.12

Рис длинный 68 25.0 1700.00

Рис круглый 62 56.75 3518.50

**- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

Всего товаров на сумму 58203.62

Количество элементов 4

### Контрольные вопросы

В качестве контрольного вопроса может быть задан любой вопрос по темам

(смотрите вопросы к лабораторным работам прошлого семестра):

* типы и модификаторы типов данных
* операции и операторы
* массивы и строки
* указатели и ссылки
* пользовательские функции
* структуры
* динамическая работа с памятью
* ввод-вывод на консоль
* ввод-вывод в файл

## Варианты заданий

|  |  |
| --- | --- |
| **Номер компьютера** | **Задание** |
| 1, 11, 21 | **Пищевое производство (расчеты).**  Определить следующую структуру :  char\* fam ; //Название изделия  int type; //тип изделия (1- булочка, 2-пирожок, 3-пирожное)  double weight; // вес  int quant; // количество  double cost; //стоимость  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск изделий по названию * Фильтр по типу (найти все изделия заданного типа) * Сортировать по уменьшению стоимости (сначала более дорогие)   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |
| 2, 12, 22 | **Поезд (заказ билетов)**  Определить следующую структуру :  char\* fio; // ФИО пассажира  char\* point\_from; // пункт отправления  char\* point\_to; // пункт прибытия  double price; // цена билета  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск билета по фио пассажира * Фильтр по цене (найти все билеты с ценой более заданного значения) * Сортировать по алфавиту (поле point\_from)   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |

|  |  |
| --- | --- |
| 3, 13, 23 | **Почта (ценное письмо)**  Определить следующую структуру :  int ind1; // индекс получателя  char\* adr1; //адрес получателя  char\* name1; // имя получателя  char\* adr2; // адрес отправителя  char\* name2; // Фамилия отправителя  double cost; // стоимость письма  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск письма по отправителю * Фильтр по индексу (найти все письма с заданным индексом) * Сортировать по увеличению стоимости   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |
| 4, 14, 24 | **Студент**  Определить следующую структуру :  char\* fam; //Фамилия  char\* name; // Имя  char\* faculty // факультет  int grup // номер группы  Создать обязательные функции, указанные в общих требованиях. Создать функции для данного варианта:   * Поиск студента по фамилии * Фильтр по номеру группы (найти всех студентов заданной группы) * Сортировать по алфавиту (по полю «фамилия»)   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |

|  |  |
| --- | --- |
| 5, 15, 25 | **Производство (поставщики)**  Определить следующую структуру :  char\* firma; // название фирмы поставщика  char\* adr2; // юридический адрес  int type; // 1- поставщик сырья, 2- поставщик оборудования  double balance; // сальдо  // (положительное - должны поставщику, отрицательное-наоборот)  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск поставщика по названию фирмы * Фильтр по типу (найти всех поставщиков заданного типа) * Сортировать по уменьшению сальдо   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка 5. При выходе из программы запомнить измененные данные на диске |
| 6, 16, 26 | **Поликлиника**  Определить следующую структуру :  int specialty; // специальность(1-терапевт, 2-кардиолог, 3- стоматолог)  int qualification; //квалификация (0-высшая, 1- первая, 2- вторая)  char\* fio; // ФИО врача  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск врача по фио * Фильтр по квалификации и специальности (найти всех врачей заданной специальности и квалификации) * Сортировать по алфавиту   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |

|  |  |
| --- | --- |
| 7, 17, 27 | **Склад (товары)**  Определить следующую структуру :  char\* fam ; //Название товара  double price; // цена  int quant; // количество  int num; // номер секции  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск товара по названию * Фильтр по номеру секции (найти все изделия заданной секции) * Сортировать по уменьшению количества   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на дискеоздать массив объектов и расположить его в динамической памяти |
| 8, 18, 28 | **Банк (депозит)**  Определить следующую структуру :  char\* fam; //Клиент: Фамилия  char\* name; // Имя  int account; // номер счета  double sum; // сумма на счете (сумма депозита )    Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск клиента по фамилии * Фильтр по сумме на счете (найти всех клиентов с суммой больше заданной) * Сортировать по алфавиту   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |

|  |  |
| --- | --- |
| 9, 19, 29 | **Работник (зарплата)**  Определить следующую структуру :  char\* fam; //Работник: Фамилия  char\* name; // Имя  int department; // отдел  double salary; // оклад  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск работника по фамилии * Фильтр по отделу (найти всех работников заданного отдела) * Сортировать по возрастанию оклада   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |
| 10, 20, 30 | **Преподаватель**  Определить следующую структуру :  int department; // кафедра (1-ИПОВС, 2-ВМ, 3-ВТ)  char\* fio; //ФИО преподавателя  int status; //ученое звание ( 1- преподаватель, 2- доцент, 3-доктор наук)  Создать обязательные функции, указанные в общих требованиях.  Создать функции для данного варианта:   * Поиск преподавателя по фио * Фильтр по статусу (найти всех преподавателей заданного статуса) * Сортировать по алфавиту   **Тестовая программа:**   1. Создать массив объектов и расположить его в динамической памяти 2. При запуске программы данные ввести с клавиатуры (первый запуск программы) или загрузить с диска (все последующие запуски) 3. Добавить несколько новых элементов (ввод с клавиатуры) 4. Выполнить все реализованные функции (поиск, фильтр, сортировка) 5. При выходе из программы запомнить измененные данные на диске |