1. аннотация

Целью курсового проектирования является систематизация, закрепление и углубление знаний в области основ программирования и совершенствование практических навыков разработки программ на языках С/С++ с использованием методологии структурного программирования. Для достижения цели на разных этапах курсового проектирования должны быть решены следующие задачи:

* выбор варианта задания и детализация поставки задачи; - определение требований к функциям, выполняемых разрабатываемой программой;
* выбор типов и проектирование структур данных, определяющих способы представления, хранения и преобразования входных, выходных и промежуточных данных;
* разработка модульной структуры программы, определение функций модулей и способов их взаимодействия;
* написание текста (кодирование) программных модулей на алгоритмическом языке;
* разработка тестовых примеров;
* тестирование и отладка программы;
* разработка программных документов в соответствии с действующими стандартами.

Выполнение курсовой работы осуществляется по индивидуальному заданию, в соответствии с которым необходимо разработать программу для хранения и обработки информации, представленной в форме таблицы. Подобные программы являются важной составной частью различных автоматизированных информационных систем. Приобретение студентами практического опыта в разработке таких программ обеспечит хороший фундамент для освоения последующих дисциплин, связанных с разработкой информационных систем.

В рамках настоящего курсового проектирования при разработке программы используется методология структурного программирования, которая базируется на методах нисходящего проектирования и модульного программирования, структурного кодирования программ [4, 8]. Соблюдение этой методологии позволяет сократить время разработки программ и повысить их качество.

При выполнении курсовой работы необходимо учитывать, что разработка программных систем – это итеративный процесс. При этом основные фазы создания программы – проектирование, программирование и тестирование – нельзя строго разделить и зачастую они выполняются параллельно либо со значительным перекрытием во времени.

1. Введение

В рамках данного курсового проектирования, представляем программу “Учет и управление данными о материальных ценностях”, разработанную на основе технического задания в организации Севастопольский государственный университет, утвержденного 01.09.2023. Эта программа представляет собой инструмент для эффективного учета и управления информацией о материальных ценностях.

Управление данными о материальных ценностях имеет важное значение для организаций. Это позволяет хранить, обновлять и анализировать информацию о материальных ценностях, что в свою очередь способствует оптимизации процессов управления ресурсами и принятию обоснованных решений в области управления материальными ценностями.

Целью данного курсового проекта является разработка программы "Учет и управление данными о материальных ценностях", предоставляющей средства для эффективного учета информации о материальных ценностях в организации, а также возможность управления этими данными. Программа поможет улучшить организацию и управление материальными ценностями.

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ ПРОГРАММЫ

Программа "Учет и управление материальными ценностями" предназначена для систематизации и эффективного управления информацией о материальных ценностях в организации. Ее основная цель состоит в создании удобного и функционального инструмента, который поможет облегчить процессы учета материальных ценностей и оптимизировать операции, связанные с данными о них.

Программа может успешно применяться в следующих областях:

1. Учет материальных ценностей: Ведение информации о материальных ценностях, включая их характеристики, стоимость, количество и местонахождение.
2. Управление запасами: Оптимизация и контроль за запасами материальных ценностей для предотвращения избыточных закупок или нехватки материалов.
3. Планирование закупок и расходов: Анализ данных о материальных ценностях для точного планирования закупок и оптимизации расходов.
4. Отчетность и аналитика: Генерация отчетов и аналитических данных для принятия информированных решений в управлении материальными ценностями.

Цель программы заключается в автоматизации учета и управления данными о материальных ценностях для повышения эффективности и точности управления материальными ресурсами в организации.

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ ПРОГРАММЫ
   1. Постановка задачи на разработку программы

Вариант задания: Даны сведения за квартал о материальных ценностях в стоимостном выражении по филиалам завода. Структура записи: номер завода, номер филиала, фамилия ответственного за материальные ценности, наличие материальных ценностей на начало периода, получено материальных ценностей на сумму, выбыло на сумму. Получить ведомость движения материальных ценностей за отчетный период, содержащую: стоимость материальных ценностей по каждому филиалу на конец отчетного периода, по всему заводу на конец периода, а также итоговые цифры по каждому виду (на начало периода, получено, выбыло, на конец периода) и по всему заводу в целом. Проконтролировать двумя путями (по строке и столбцу) получение итоговой цифры: наличие материальных ценностей на конец отчетного периода в стоимостном выражении по заводу в целом

Для представления таблицы в коде программы необходимо использовать списки (однонаправленные или двунаправленные) или бинарные деревья. Запрещается использовать классы или готовые контейнеры.

Разрабатываемая программа должна использовать меню-ориентированный интерфейс, обеспечивающий выполнение следующего минимального состава действий.

Программа должна обеспечивать следующие функции:

1. Начальное создание таблицы. При необходимости создания новой таблицы исходные данные считываются из текстового файла. Имя файла должен задавать пользователь.

2. Просмотр таблицы. При этом необходимо предусмотреть возможность скроллинга.

3. Добавление новой записи в таблицу.

4. Удаление записи. Удаляемый элемент выбирается по одному из полей таблицы (ключевому). Ключевое поле выбирает студент.

5. Корректировка записи в таблице. Корректируемую запись выбирают по одному из полей таблицы (ключевому).

6. Сортировка таблицы. Сортировка производится по одному из полей таблицы (ключевому). Метод сортировки выбирает студент.

7. Поиск записи в таблице (по не ключевому полю).

8. Сохранение таблицы в файле. Имя файла должен вводить пользователь. Сохранять таблицу следует в текстовый файл и в типизированный.

9. Чтение данных из файла. Имя файла должен вводить пользователь. При чтении данных необходимо создавать новый список.

10. Обработка таблицы и просмотр результатов обработки. Результат обработки необходимо вывести на экран и в текстовый файл. Имя файла вводит пользователь.

11. Выход ― завершение работы программы.

* 1. Описание и обоснование выбора метода организации входных и выходных данных

В качестве структуры данных для программы был выбран односвязный список. Реализация динамической структуры в виде односвязного списка имеет следующие преимущества:

* **Экономия памяти при добавлении элементов:** Односвязный список выделяет дополнительную память только для каждого нового элемента. В отличие от динамического массива, где требуется выделение новой области памяти для всех элементов, копирование старых данных в новую область и только потом освобождение памяти, выделенной под старую область. При увеличении количества элементов в структуре данных, это преимущество односвязного списка становится более заметным.
* **Меньшее количество дополнительных операций при манипулировании данными:** Односвязный список требует меньше операций при добавлении нового элемента или удалении существующего. Это обусловлено отсутствием необходимости перемещать большие блоки данных, как это требуется в случае динамического массива.

В качестве входных и выходных данных используется текстовый файл, поля которого задаёт пользователь. Данные легко читаются и изменяются в файле без необходимости кодирования. Это обеспечивает удобство в работе с информацией, а также улучшает ее портируемость и доступность для редактирования.

* 1. Обоснование выбора языка и среды программирования

В качестве языка программирования был выбран язык C с элементами С++ в среде разработки Embarcadero Dev-C++ 6.3.

Компилируемый статически типизированный язык программирования С был разработан сотрудником Bell Labs, Деннисом Ритчи, в период с 1969 по 1973 годы. Изначально созданный для реализации операционной системы UNIX, он был адаптирован для работы на различных платформах. Дизайн языка был ориентирован на соответствие машинным инструкциям, что обеспечило его широкое использование в проектах, где требовался язык программирования, близкий к ассемблеру.

Язык С применяется в различных областях, включая операционные системы и прикладное программное обеспечение для широкого спектра устройств: от суперкомпьютеров до встраиваемых систем. Эта универсальность и мощь делают его популярным выбором для разработчиков, работающих в сферах, где требуется эффективное управление ресурсами и высокая производительность.

С является одним из наиболее влиятельных языков программирования и оказал значительное влияние на развитие компьютерной индустрии. Его особенности, такие как эффективность, портируемость и близость к аппаратному уровню, продолжают делать его популярным средством разработки программного обеспечения в различных областях информационных технологий.

К преимуществам языка можно отнести:

* Универсальность (используется практически во всех существующих ЭВМ)
* Малое потребление ОЗУ
* Малый размер исполняемого файла
* Быстрота компилирования и выполнения программ
* Гибкость написания сложных многоструктурных программ
* Высокая стурктурированность

Но главным критерием выбора данного языка программирования стало изучение данного языка программирования и освоение его на практике в течение первого курса обучения и именно на нём в соответствии с техническим заданием, должна быть написана программа.

* 1. Описание алгоритмов функционирования программы

Для выполнения КП были разработаны следующие функции:

* MaterialRecord\* createRecord() — Функция для создания новой записи;
* void addRecord() — Функция для добавления записи в таблицу;
* void deleteRecord() — Функция для удаления записи из таблицы по ключевому полю;
* void editRecord() — Функция для корректировки записи в таблице по ключевому полю;
* void displayRecords() — Просмотр таблицы с возможностью скроллинга;
* void sortRecords() — Функция для сортировки таблицы по нескольким полям;
* void searchByLastName() — Функция для поиска записи в таблице по не ключевому полю (Фамилия);
* void saveToTextFile() — Функция для сохранения таблицы в файле;
* void loadFromTextFile() — Создание таблицы из файла;
* void generateReport() — Обработка таблицы и вывод результатов.

На рисунках ниже представлены структурные схемы некоторых алгоритмов (функций) программы.

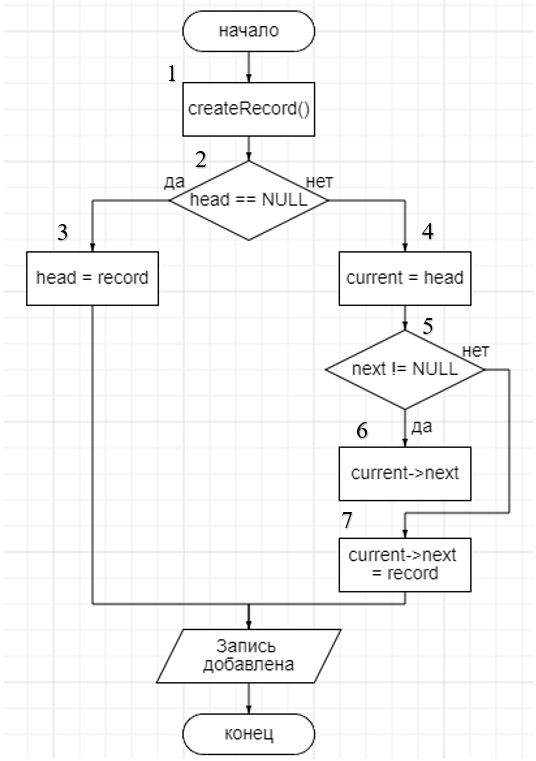


Рисунок 2.1 – Функция addRecord

На рисунке 2.1 представлена функция добавление записи в таблицу.

Поблочное описание функции:

Блок 1 – указатель record на структуру Record и присваивает ему значение, возвращаемое функцией createRecord().

Блок 2 – проверяет, если переменная head, являющаяся указателем на первую запись в таблице, равна NULL.

Блок 3 – Если head равен NULL, то эта строка присваивает ему значение указателя record.

Блок 4 – создает указатель current на структуру Record и присваивает ему значение head.

Блок 5 – запускает цикл while, который выполняется, пока у текущей записи temp есть следующая запись.

Блок 6 – внутри цикла while эта строка переходит к следующей записи в связном списке, присваивая current значение temp->next.

Блок 7 – После выхода из цикла while, эта строка устанавливает указатель next последней записи в связном списке (temp) на новую запись record. Таким образом, новая запись добавляется в конец списка.



Рисунок 2.2 – Функция createRecord

На рисунке 2.2 представлена функция создания записи.

Поблочное описание функции:

Блок 1 – выделяет память под новую запись (структуру MaterialRecord).

Блок 2 – устанавливает поле next структуры record в значение NULL. Это гарантирует, что поле next указывает на конец списка или связанной структуры данных.

Блок 3 – возвращает указатель record на созданную запись (структуру MaterialRecord).

* 1. Обоснование состава технических и программных средств

Технические средства:

Для запуска программы необходим персональный компьютер с операционной системой Windows.

Минимальные требования к аппаратному обеспечению включают процессор с тактовой частотой не менее 1 ГГц, 1 ГБ оперативной памяти, и несколько мегабайт свободного места на жестком диске.

Программные средства:

Операционная система Windows.

Исходный код программы: Весь исходный код программы, включая заголовочные файлы и текст программы на языке C, предоставляется в виде исходных файлов.

Библиотеки: для работы с файлами и вводом-выводом используются стандартные библиотеки языка C, такие как stdio.h, stdlib.h, и другие.

Инструменты разработки: для разработки программы использовалась среда Embarcadero Dev-C++. Она предоставляет удобную платформу для разработки, отладки и сборки программы.

Тестовые данные: для тестирования программы могут использоваться тестовые данные, созданные пользователем.

Все указанные программные и технические средства обоснованы на основе анализа требований программы, а также удовлетворяют минимальным системным требованиям, необходимым для нормального функционирования программы "Учет и управление данными о материальных ценностях" на платформе Windows.

Размер exe-файла программы – 93 266 373 байта

Размер cpp-файла программы – 23 678 байт

Размер одной сохраненной записи в текстовом файле – 221 байт

Размер одной сохраненной записи в типизированном файле – 624 байта

1. ВЫПОЛНЕНИЕ ПРОГРАММЫ
   1. Условия выполнения программы

Минимальный состав аппаратных средств:

* Персональный компьютер (ПК) под управлением операционной системы Windows.
* Процессор с тактовой частотой не менее 1 ГГц.
* Минимум 1 ГБ оперативной памяти (RAM).
* Свободное место на жестком диске для хранения программы и данных.
* Клавиатура и мышь (или другие устройства ввода).

Минимальное программное обеспечение:

* Операционная система Windows (XP, 7, 8, 10 и др.).
* Компилятор языка C, такой как GCC, для сборки программы.
  1. Загрузка и запуск программы

Инсталляция программы: Установка программы не требуется, поскольку она поставляется в виде исходного кода и может быть скомпилирована с использованием компилятора языка C. Пользователь должен убедиться в наличии необходимого компилятора и инструментов разработки, таких как Microsoft Visual Studio или Embarcadero Dev-C++, для компиляции программы из исходного кода.

Загрузка и компиляция:

* Скачайте исходный код программы из репозитория или хранилища.
* Откройте исходный код программы в выбранной среде разработки (например, Microsoft Visual Studio).
* Скомпилируйте программу, следуя инструкциям вашей среды разработки.

Запуск программы:

* После успешной компиляции запустите программу, следуя инструкциям вашей среды разработки.
* Программа будет выполняться в командной строке, предоставляя пользователю меню с доступными функциями.