АННОТАЦИЯ

Данная пояснительная записка содержит в себе описание, характеристика, область применения и текст программы приложения, работающего с данными автостанций, такими как: Ф.И.О. директора, номер автостанции, расход топлива (у.е.), количество транспорта. Приложение способно сформировать список из записей, выполнить удаление отдельной записи или всего списка, сортировать записи, выполнить поиск по одному из полей. Возможно применение в сфере транспортной инфраструктура для учета транспортных ресурсов и формирования представления о расходах топлива.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ…………………………………………………………………

1 НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ………………………...

2 ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ………………………………..

2.1 Постановка задачи на разработку программы……………………..

2.2 Описание и обоснование выбора метода организации входных, выходных и промежуточных данных…………………………………….

2.3 Описание алгоритмов функционирования программы…….........

2.3.1 Схема алгоритма основной программы…………………...

2.3.2 Схема алгоритма добавления элемента…………………...

2.3.3 Схема алгоритма загрузки списка из файла……………….

2.3.4 Схема алгоритма сохранения в файл………………………

2.3.5 Схема алгоритма удаления списка………………………...

2.3.6 Схема алгоритма ввода данных с консоли………………...

2.3.7 Схема алгоритма поиска элементов ………………………

2.3.8 Схема алгоритма рисования пункта меню………………...

2.3.9 Схема алгоритма вывода меню на экран…………………

2.3.10 Схема алгоритма вывода данных…………………………

2.4 Обоснование состава технических и программных средств.........

3 ВЫПОЛНЕНИЕ ПРОГРАММЫ………………………………………

3.1 Условия выполнения программы…………………………………

3.2 Загрузка и запуск программы……………………………………..

3.3 Проверка работоспособности программы………………………..

ВЫВОДЫ………………………………………………………………….

ВВЕДЕНИЕ

Данный документ содержит описание программы “Программа обработки данных”, разработанной на основании технического задания.

На сегодняшний день, данный курсовой проект является не актуальным, так как в мире существует множество альтернативных приложений с более внушительным функционалом, которые перекрывают в плане функционала и удобства данный курсовой проект.

Целью данного курсового проекта является улучшение и закрепление навыков программирования, в результате чего будет создано, позволяющего вести учет авиарейсов на основе данных, таких как:

- номер рейса

- имя рейса

- марка самолета

- расходы на рейс

- количество пассажиров

Для достижения поставленной цели, было написано приложение с графическим интерфейсом, организующее двусвязный список записей с полями из задания, а также выполняющее с этим списком все необходимые действия.

Для создания и оформления проекта были пройдены такие этапы:

- проектирование структуры будущей программы;

- написание простейших необходимых функций для работы со списком;

- реализация функции согласно индивидуальному заданию;

- реализация более продвинутого интерфейса;

- отладка и тесты программы, исправление ошибок;

- разработка документации;

1 НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ ПРОГРАММЫ

Назначение программы – ведение учета авиарейсов. Выполняет простейшие действия со списком: создание, добавление записи, удаление элементов, сортировка, поиск.

Приложение предназначено для использования в сфере транспортной инфраструктура для учета транспортных ресурсов и формирования представления о расходах топлива.

2 ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ ПРОГРАММЫ

2.1 Постановка задачи на разработку программы

Задача: Даны сведения о расходовании на автобазах города топлива по следующему макету: номер автобазы, Ф.И.О. директора (15 символов), израсходовано топлива (в условных единицах), количество автомашин на базе. Подсчитать средний расход топлива на одну машину по каждой базе и в целом по городу.

2.2 Описание и обоснование выбора метода организации входных, выходных и промежуточных данных

Ввод данных был реализован с помощью двух способов: с клавиатуры и с файла. С клавиатуры ввод осуществлялся в случае организации списка и в случае редактирования.

- номер базы – int;

- фамилия, имя, отчество – char[15];

- расход топлива – float;

- количество транспорта – int;

Все данные организованы в структуру *ab*,объединяющую в себе все вводимые пользователем поля. Это сделано для корректной организации ввода из файлов и с клавиатуры. Эта структура входит в структуру *list*, содержащую в себе информационное поле элемента и указатель на следующий и предыдущий элемент в двунаправленном списке. Выходные данные организуются в двух режимах: в файл или на экран. На дисплей список выводится в виде таблицы (табл. 2.1) с подсветкой активного элемента с возможностью перемещения по записям с помощью стрелок на клавиатуре, а также перемещения между страницами с помощью клавиш Page Up и Page Down. При нажатии клавиши Enter возможно редактирование элемента по полю, которое выберет пользователь, При нажатии клавиши Escape происходит выход к главному меню.

Таблица 2.1 – Формат выходных данных

|  |  |  |  |
| --- | --- | --- | --- |
| Номер автобазы | Ф.И.О. директора | Расход топлива | Количество траспорта |

Промежуточные величины в приложении были представлены главным меню, содержащей пункты «Организация списка», «Просмотр списка», «Работа с файлами», «Удаление», «Добавление в конец списка», «Сортировки», «Поиск».

Для полноценной реализации проекта были объявлены следующие функции:

*- ab\* Create(const ab& z)* – функция организации списка с клавиатуры;

- *ab\* Add(ab \*beg, ab \*end, const ab& z)* – добавления элемента в список;

- *ab\* Delete(ab\* beg)* – функция удаления всего списка;

- *ab\* DeleteOne(ab \*beg, int n)* – функция удаления одного элемента списка;

- *ab\* Search(ab \*beg)* – функция поиска элемента по заданному полю;

- *ab Input()* – функция ввода элемента;

- *int menu()* – функция создания меню;

- *void Print(const ab& z)* – функция печати списка;

- *void View(ab\* beg)* – функция просмотра списка;

- *void Sort(ab\* beg)* – сортировки списка по заданному полю;

- *int ReadFile(char\* file, ab\*\* beg, ab\*\* end)* – функция чтения списка из файла. Выполняется автоматически при запуске программы;

- *int WriteFile(char\* file, ab\* elem)* – функция записи списка в файл. Выполняется автоматически при выходе из программы;

- *void Green(char \*text)* – функция окрашивания выбранного текста в зеленый цвет;

- *void Red(char \*text)* – функция окрашивания выбранного текста в красный цвет;

- *void Blue(char \*text)* – функция окрашивания выбранного текста в синий цвет;

2.4 Описание алгоритмов функционирования программы

3 ВЫПОЛНЕНИЕ ПРОГРАММЫ

3.1 Условия выполнения программы

В результате компиляции получается файл с расширением .exe размером 78,5 Кб. Для современных компьютеров размер данного файла является не существенным. Программа потребляет 5,5 Мб оперативной памяти. Поэтому она доступна для любого компьютера на базе операционной системой Windows.

3.2 Загрузка и запуск программы

Запуск программы производится с любого носителя информации путем открытия файла KURS.exe. При ее запуске окно, пользователю доступно главное меню, необходимое для взаимодействия с приложением. Изображение данного меню приводится в рисунке 3.1.

Рисунок 3.1 – Главное меню приложения

Пользователь может использовать стрелки для перемещения по пунктам меню, активный пункт меню подсвечивается ярко желтым цветом. Если пользователь нажмет стрелку «Вверх», находясь на первом пункте, то активным станет последний элемент. На рисунке 3.2 демонстрируется, как при нажатии стрелки «Вверх» активным элементом становится нижний элемент меню. Если находясь на последнем элементе меню, пользователь нажмет стрелку «Вниз», активным станет первый элемент.

Рисунок 3.2 – До нажатия и после нажатия на стрелку «Вверх»

Для выбора нужного пункта пользователю требуется нажать клавишу Enter, а для выхода – клавишу.

Для работы со списком необходимо либо загрузить из файла, либо организовать вручную. Организация вручную представляет собой последовательный ввод необходимых значений для каждого элемента списка. Демонстрация ввода с консоли представлена на рисунке 3.3.

Рисунок 3.3 – Ввод с консоли

В функции ввода реализован механизм проверки на правильный ввод, если пользователь введет неверное значение, он получит сообщение об ошибке, и ему потребуется ввести значение заново. Демонстрация вывода сообщения об ошибке представлена на рисунке 3.4

Рисунок 3.4 – Вывод сообщения об ошибке

Когда элемент введен пользователю выводится сообщение, о том не хочет ли он добавить еще один элемент, если выбирает добавить еще один элемент происходит ввод, иначе вход в главное меню. Демонстрация выбора представлена на рисунке 3.5.

Рисунок 3.5 – Вывод сообщения о добавлении еще одного элемента

Если пользователь захочет загрузить файл, ему следует выбрать пункт меню работа с файлами. При выборе этого пункта появится контекстное меню, в котором пользователю следует выбрать необходимое действие. Работа этого пункта представлена на рисунке 3.6.

Рисунок 3.6 – Вывод меню о работе с файлами

При загрузке списка из файла необходимо ввести его имя с расширением либо путь к нему. Если загрузка из файла успешна, будет выведено соответствующее сообщение. Иначе – сообщение об ошибке. Демонстрация работы этого пункта меню изображено на рисунке 3.7

Рисунок 3.7 – Вывод сообщения о том, что файл не найден

Процесс сохранения списка в файл полностью аналогичен процессу загрузки.

Пользователь может просмотреть, что произошло со списком после его действий. Для этого есть пункт меню «Просмотр списка». При его активации на экран выводится до 5-х записей списка в табличном виде. На рисунке 3.8 продемонстрирован просмотр списка, активный элемент подсвечивается ярко-желтым цветом.

Рисунок 3.8 – Просмотр списка

Для просмотра остальных записей предусмотрено переключение между страницами. Чтобы перейти на следующую страницу необходимо нажать клавишу Page Up, а для того чтобы перейти на предыдущую нужно нажать Page Down. Данная функция показана на рисунке 3.9

Рисунок 3.8 – Переключение между страницами списка

В таблице предусмотрено редактирование элемента с помощью нажатия клавиши Enter. При редактировании элемента загружается меню, позволяющее выбрать поле для корректировки. Вывод меню продемонстрирован на рисунке 3.9.

Рисунок 3.9 – Меню выбора поля редактирования

После выбора поля требуется ввести новое значение. Ввод также представлен с обработкой ошибок, как в случае организации списка. Данная функция представлена на рисунок 3.10.

Рисунок 3.10 – Ввод нового значения

Пользователь может удалить весь список либо же удалить первый элемент. Для этого существует пункт меню «Удаление» в главном меню. При выборе пункта на экране появляется меню, продемонстрированное на рисунке 3.11.

Рисунок 3.11 – Вывод меню для удаления

Пользователь может удалить весь список. Для этого используется пункт «Удаление списка». Функция удалит весь список и выведет на экран соответствующее сообщение об успешной очистке. Пример удаления вы можете увидеть на рисунке 3.12.

Рисунок 3.12 – Удаление всего списка

В главном меню реализована функция добавления в конец списка нового элемента, при этом ввод также представлен с обработкой ошибок, как в случае организации списка. Работа данной функции продемонстрирована на рисунке 3.13 – 3.14

Рисунок 3.13 – Добавление в конец списка

Рисунок 3.14 – Проверка добавление в конец списка

Пользователю доступен поиск элементов по определенному значению поля списка. Это реализовано в выборе пункта «Поиск». При открытии отображается меню поиска, позволяющее выбрать поле, по которому будет производиться поиск (рисунок 3.15).

Рисунок 3.15 – Вывод меню для поиска

После выбора поля вводится значение, по которому будет производиться поиск (рисунок 3.16).

Рисунок 3.16 – Ввод значения для поиска

Если такой элемент существует, он выводится на экран.

Рисунок 3.17 – Вывод значения найденного элемента

Пользователь может отсортировать список по одному из трех полей по возрастанию. Для этого существует пункт меню «Сортировки» в главном меню. При выборе пункта на экране появляется меню, продемонстрированное на рисунке 3.18.

Рисунок 3.18 – Вывод меню для сортировки списка

После выбора нужного пункта список сортируется.

Рисунок 3.19 – Вывод отсортированного списка