МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

**Кафедра защиты информации

**КУРСОВАЯ РАБОТА**

**по дисциплине: «Программирование» на тему: «** *Разработка программного обеспечения для создания и ведения журнала "Багаж пассажира"* **»**

Выполнил: Проверил:

Студент гр. « *АБ-121* », « *АВТФ* » *доцент кафедры ЗИ*

*Новиков Ян Валерьевич Архипова А. Б.*

« » 20 г. « » 20 г.

(подпись) (подпись)

Новосибирск 2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра *Защита информации*

(полное название кафедры)

УТВЕРЖДАЮ

Зав. кафедрой Иванов А. В.

(подпись, дата)

**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

студенту *Новикову Яну Валерьевичу*

(фамилия, имя, отчество)

Направление подготовки *10.03.01 – Информационная безопасность*

(код и наименование направления подготовки бакалавра)

*Факультет автоматики и вычислительной техники*

(полное название факультета)

Тема

*Разработка программного обеспечения для создания и ведения журнала "Багаж пассажира"*

Календарный план

|  |  |  |
| --- | --- | --- |
| Наименование задач (мероприятий), составляющих задание | Дата выполнения задачи (мероприятия) | Подпись руководителя |
| 1 | 2 | 3 |
| Развернутая постановка задачи, изучение необходимой научно- технической литературы | 17.09.2022 г. |  |
| Разработка структуры данных и  алгоритма решения задачи | 04.10.2022 г. |  |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Написание текста задачи | 21.10.2022 г. |  |
| Тестирование и отладка  программного продукта | 23.11.2022 г. |  |
| Оформление отчета о проделанной  работе | 13.12.2022 г. |  |
| Сдача работы руководителю и ее  защита | 15.12.2022 г. |  |

Задание согласовано и принято к исполнению.

Руководитель от НГТУ

Архипова Анастасия Борисовна

(фамилия, имя, отчество)

Студент

(фамилия, имя, отчество)

к.т.н.

(ученая степень, ученое звание)

(подпись, дата)

(факультет, группа)

(подпись, дата)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Ведение журналов является неотъемлемой частью работы тех, кто имеет дело с перевозкой багажа. В современном мире появилось множество возможностей упростить данный процесс. Я решил воспользоваться современными возможностями для создания программного обеспечения, дабы упростить ведение журнала.

\*Теоретическое обоснование!\*

Целью курсовой работы является разработка программного обеспечения для создания и ведения журнала «Багаж пассажира».

Для реализации поставленной цели необходимо решить ряд задач:

* проанализировать программные средства, выполняющие аналогичную задачу;
* проработать структуру программы и проанализировать, какие данные и в каком виде будут храниться в журнале;
* изучить работу с файлами и данными, располагающимся в журнале;
* разработать функционал для работы с данными, а именно:
  + добавление рейса, пассажира и веса его вещей;
  + изменение рейса определённого пассажира;
  + внесение изменений в файл, содержащий информацию о рейсах;
  + удаление всех ранее записанных данных;
  + вывод информации лишь о тех пассажирах, у которых вес хотя бы одной вещи превышает контрольный.
* изучить и проанализировать необходимую информацию о разработке графического приложения;
* разработать и реализовать внешнюю оболочку программы, понятную и удобную для рядового пользователя;
* реализовать функционал и связать его с внешней оболочкой программы.

Ожидаемый результат: полностью функционирующая программа, позволяющая осуществлять удобное и понятное создание и ведение журнала «Багаж пассажира».

1. Постановка задачи
   1. Характеристика задачи
      1. Программа «Passenger\_baggage» предназначена для облегчения создания и ведения журнала для учёта багажей пассажиров – добавление в журнал рейса с определённым номером, пассажира, обозначенного ФИО, количество вещей определённого пассажира и вес каждой из них, а также ведения, удаления, изменения данных.
      2. Программа используется работниками транспортных компаний для учёта поступления и хранения багажа.
      3. Прекращение работы с журналом происходит путём закрытия окна программы.
      4. Специальных ограничений на временные характеристики решения задачи не налагается.
      5. Специальных требований на уровень подготовки пользователя не налагаются. Но лицо, работающее с программой, должно иметь минимальное представление о компьютере (знание необходимых операций).
   2. Выходная информация

Перечень и описание выходных данных содержится в таблице 1.1.

Таблица 1.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Иденти-  фикатор | Форма представления | Наименование | Периодичность | Получатель |
| 1 | 2 | 3 | 4 | 5 |
| V001 | Видеограмма | LoginForm | По запросу | Пользователь |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| V002 | Видеограмма | Passenger Baggage Magazine | По запросу | Пользователь |
| F001 | Файл | Whole magazine.txt | По запросу | Пользователь |
| F002 | Файл | ChosenFlight.txt | По запросу | Пользователь |
| F003 | Файл | SummOfWeightInChosenFlight.txt | По запросу | Пользователь |
| F004 | Файл | OldList.xml | По запросу | Пользователь |

Продолжение таблицы 1.1.

Описание структурных единиц видеограммы V001 содержится в Таблице 1.2.

Таблица 1.2 – Описание структурных единиц видеограммы V001

|  |  |  |  |
| --- | --- | --- | --- |
| Выходная информация | | Идентификатор | |
| Данные для входа | | V001 | |
| Наименование структурных единиц | Обозначение | Требование к точности | |
| Длина в знаках | Диапазон |
| Логин | LoginText | X(30) | ‘А’-‘Я’,’а’-’я’,  ‘A’-‘Z’, ‘a’-‘z’, ‘ ’,’.’, ‘0’-‘9’. |
| Пароль | PasswordText | X(30) | ‘А’-‘Я’, ’а’-’я’,  ‘A’-‘Z’, ‘a’-‘z’, ‘ ’,’.’, ‘0’-‘9’. |

Форма видеограммы V001 представлена на рисунке 1.1.

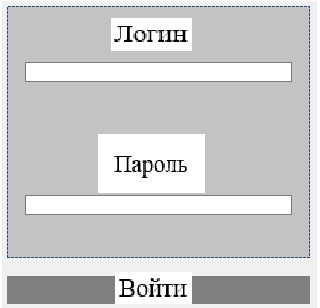


Рисунок 1.1 – Форма видеограммы V001

Описание структурных единиц видеограммы V002 содержится в Таблице 1.3.

Таблица 1.3

|  |  |  |  |
| --- | --- | --- | --- |
| Выходная информация | | Идентификатор | |
| Данные для входа | | V001 | |
| Наименование структурных единиц | Обозначение | Требование к точности | |
| Длина в знаках | Диапазон |
| ФИО | FIObox | A(50) | ‘А’-‘Я’,’а’-’я’,  ‘A’-‘Z’, ‘a’-‘z’, ‘ ’. |
| Номер рейса | NumRace | 9(5) | ‘0’-‘9’. |
| Вес добавляемой вещи | WeightOfItem | 9(2) | ‘0’-‘9’. |
| ФИО | FIOChange | A(50) | ‘А’-‘Я’,’а’-’я’,  ‘A’-‘Z’, ‘a’-‘z’, ‘ ’. |
| Текущий номер рейса | OldNumFlight | 9(5) | ‘0’-‘9’. |
| Новый номер рейса | NewNumFlight | 9(5) | ‘0’-‘9’. |
| Контрольный вес | ControlWeightNumber | 9(2) | ‘0’-‘9’. |
| Список пассажиров | ListOfThePasengers | - | - |
| Содержимое элемента списка | TextOfItem | X(32767) | ‘А’-‘Я’,’а’-’я’,  ‘A’-‘Z’, ‘a’-‘z’, ‘ ’,’.’, ‘0’-‘9’. |

Форма видеограммы V002 представлена на рисунке 1.2.

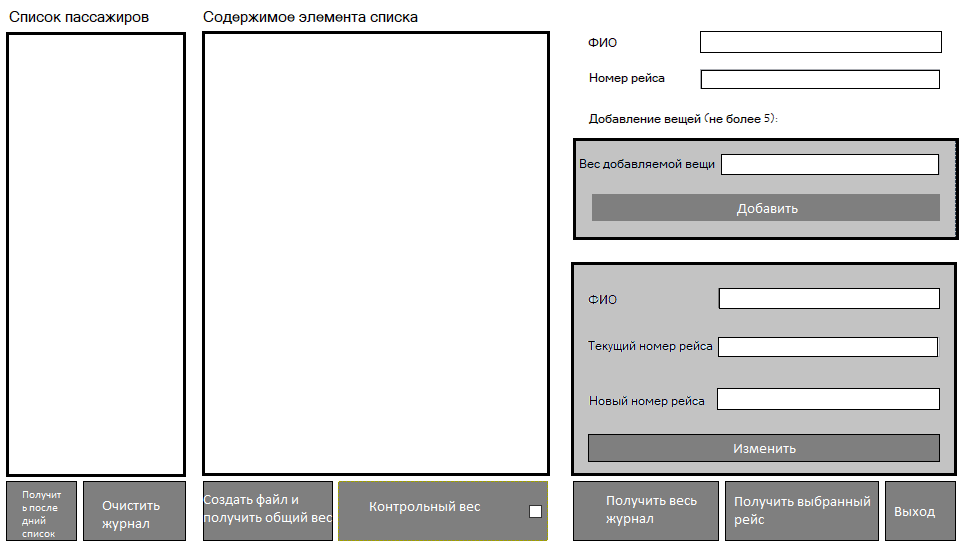


Рисунок 1.2 – Форма видеограммы V002

***Примерное содержимое файла Whole magazine.txt:***

Number of Flight: \*номер первого рейса\*

Name of the pasenger: \*ФИО первого пассажира\*

Number of items: \*кол-во вещей\*

Weight of items, that have pasenger: \*вес каждой вещи\*

Name of the pasenger: \*ФИО второго пассажира\*

Number of items: \*кол-во вещей\*

Weight of items, that have pasenger: \*вес каждой вещи\*

…

Number of Flight: \*номер второго рейса\*

Name of the pasenger: \*ФИО первого пассажира\*

Number of items: \*кол-во вещей\*

Weight of items, that have pasenger: \*вес каждой вещи\*

Name of the pasenger: \*ФИО второго пассажира\*

Number of items: \*кол-во вещей\*

Weight of items, that have pasenger: \*вес каждой вещи\*

…

***Примерное содержимое файла ChosenFlight.txt:***

Number of Flight: \*номер выбранного рейса\*

Name of the pasenger: \*ФИО первого пассажира\*

Number of items: \*кол-во вещей\*

Weight of items, that have pasenger: \*вес каждой вещи\*

Name of the pasenger: \*ФИО второго пассажира\*

Number of items: \*кол-во вещей\*

Weight of items, that have pasenger: \*вес каждой вещи\*

…

***Содержимое файла OldList.xml*** будет состоять из сериализованного объекта класса. Будет использована сериализация xml.

* 1. Входная информация

Перечень и содержание входных сообщений содержится в таблице 1.4.

Таблица 1.4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентифи- катор | Форма представле- ния | Наименование | Периодичность | Получатель |
| V001 | Видеограмма | LoginForm | По запросу | Пользователь |
| V002 | Видеограмма | Passenger Baggage Magazine | По запросу | Пользователь |
| F004 | Файл | OldList.xml | По запросу | Пользователь |

Описание структурных единиц видеограммы V001 содержится в Таблице 1.2. Форма видеограммы V001 представлена на рисунке 1.1.

Описание структурных единиц видеограммы V002 содержится в Таблице 1.3. Форма видеограммы V002 представлена на рисунке 1.2.

Содержимое файла OldList.xmlбудет сериализовано и преобразовано в список рейсов на видеограмме V002. Содержимое файла уточняется в п. 1.2.

1. Алгоритм решения
   1. Назначение и характеристика задачи.

Программа «Passenger\_baggage» предназначена для облегчения создания и ведения журнала для учёта багажей пассажиров – добавление в журнал рейса с определённым номером, пассажира, обозначенного ФИО, количество вещей определённого пассажира и вес каждой из них, а также ведения, удаления, изменения данных.

* 1. Используемая информация.

Описание входной информации содержится в пункте 1.3.

* 1. Результаты решения.

Описание выходной информации содержится в пункте 1.2.

* 1. Алгоритм решения задачи.
     1. Вывод формы для входа в приложение:

«Логин»  
«Пароль»;

* + 1. Ввод данных для входа;
    2. Нажатие кнопки «Вход»;
       1. Анализ введённых данных;
          1. Если неверно введён логин – делаем делаем видимым красный текст с оповещением, располагающимся чуть ниже поля для записи;
          2. Если неверно введён пароль – делаем делаем видимым красный текст с оповещением, располагающимся чуть ниже поля для записи;
          3. Если данные для входа введены верно – выводим основную форму со всем функционалом и закрываем старую.
    3. Ввод данных в поля:

«ФИО»  
«Номер рейса»  
«Вес добавляемой вещи»;

* + 1. Нажатие кнопки «Добавить»;
       1. Анализ введённых данных;
          1. Если неверно введено ФИО:

Выдать уведомление с текстом «Неверно введено ФИО».

* + - 1. Проверка кол-ва вещей, которые есть у пассажира;
         1. Если количество до добавления уже равно пяти:  
            Выдать уведомление с текстом «Вещей должно быть не больше пяти».
      2. Проверка на наличие введённого рейса в списке.
         1. Если уже есть такой рейс:

Проверка на наличия в рейсе этого человека.

Если такой человек есть:

Добавить ещё одну вещь к нему;

Если такого человека нет:

Добавить нового пассажира в рейс.

* + - * 1. Если такого рейса ещё нет:

Добавление в список нового элемента с именем, соответствующим номеру рейса, и добавление в новый элемент пассажира.

* + 1. Ввод данных в поля:

«ФИО»  
«Текущий номер рейса»  
«Новый номер рейса»;

* + 1. Нажатие кнопки «Изменить»;
       1. Анализ введённых данных.
          1. Если неверно введено ФИО:

Выдать на экран уведомление с текстом «Неверно введено ФИО»;

* + - * 1. Если нет рейса, указанного в поле «Текущий номер рейса»:

Выдать на экран уведомление с текстом «Рейса с таким номером не было найдено»;

* + - * 1. Если нет пассажира в указанном рейсе:

Выдать на экран уведомление с текстом: «Человека с таким ФИО в выбранном рейсе нет»;

* + - * 1. Если рейс, на который мы хотим изменить, уже есть в списке, удаляем пассажира из старого рейса и добавляем его в новый;
        2. Если рейс, на который мы хотим изменить, отсутствует в списке, добавляем новый рейс с нашим пассажиром и удаляем его из старого.
    1. Внесение данных в поле с контрольным весом;
    2. Проставление галочки в поле, рядом с контрольным весом;
       1. Проверка веса каждой вещи у всех пассажиров в выбранном рейсе.
          1. Если у пассажира вес хотя бы одной вещи больше контрольного веса, он выводится в поле с текстом;
          2. Если у пассажира нет вещей, вес которых превышает контрольный, убираем его из поля с текстом.
    3. Отмена галочки в поле, рядом с контрольным весом;
       1. Возвращение всех пассажиров элемента списка в поле с текстом.
    4. Нажатие кнопки «Очистить журнал»;
       1. Очистка списка пассажиров.
    5. Нажатие кнопки «Создать файл и получить общий вес»;
       1. Если не выбран один элемент из списка:

Выдать на экран уведомление с текстом «Выберите один элемент из списка»;

* + - 1. Подсчёт суммы всех вещей всех пассажиров выбранного рейса;
      2. Выдать на экран уведомление с общим весом вещей в выбранном рейсе;
      3. Создать файл txt, содержащий общий вес выбранного рейса.
    1. Нажатие кнопки «Получить весь журнал»;
       1. Вывод информации о пассажирах и их багаже во всех рейсах в файл txt.
    2. Нажатие кнопки «Получить выбранный рейс»;
       1. Если рейс не выбран/выбрано несколько рейсов:  
          Выдать на экран уведомление с текстом «Выберите один элемент из списка»;
       2. Если выбран один рейс:  
          Вывод информации о всех пассажирах и их багаже в созданный файл txt.
    3. Нажатие кнопки «Получить последний список»;
       1. Если есть необходимый файл xml:

Совершить десериализацию и вставить полученный список на место старого;

* + - 1. Если необходимый файл отсутствует:

Выдать на экран уведомление с текстом «Не было найдено сохранённого журнала».

* + 1. Нажатие кнопки «Выход».
       1. Проверка на наличие хотя бы одного элемента списка;
          1. Если хотя бы один элемент есть:

Совершить xml сериализацию списка пассажиров;

* + - 1. Совершить выход из программы.

1. Руководство пользователя  
   1. Введение  
      1. Программа «Passenger\_baggage» предназначена для облегчения создания и ведения журнала для учёта багажей пассажиров – добавление в журнал рейса с определённым номером, пассажира, обозначенного ФИО, количество вещей определённого пассажира и вес каждой из них, а также ведения, удаления, изменения данных.
      2. Программа предоставляет пользователю следующие возможности:

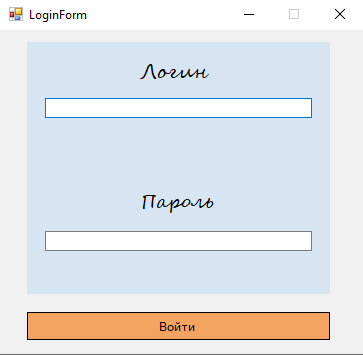
* добавление элементов в список;
* изменение элементов в списке;
* очистку списка;
* отбор подходящих элементов списка;
* получение общего веса всех вещей рейса;
* получение структурированной записи данных;
* получение данных о всех рейсах и пассажирах, включая их багаж, в виде файла txt;
* получение данных о выбранном рейсе в виде файла txt.  
  1. Назначение и условия применения  
     1. Программа “ Passenger\_baggage ” предназначена для автоматизации рабочего места и избавления от ручного труда.  
        Решение данной задачи призвано обеспечить:
* быстрый и удобный доступ к данным;
* исключение ошибок при составлении списка;
* удобный интерфейс, позволяющий работать с данными.  
  + 1. Программа реализована на языке C#. Работает в любой среде совместимой с Windows. Дисковой памяти для запуска программы требуется не менее 5 Mb. Оперативной памяти для нормальной работы программы требуется не менее 10 Mb.
  1. Подготовка к работе
     1. Система состоит из исполнительного файла Passenger\_baggage.exe.
     2. Запуск программы

Для установки программы нужно скачать файл Passenger\_baggage.exe. После установки файла его можно запустить и начать работу.

* 1. Описание операций  
     1. Введение

Основные функции системы соответствуют созданию журнала с багажами пассажиров и созданию отчёта на основе данных, содержащихся в нём. Далее описаны все функции системы, а также формы и диалоги для ввода информации. Для каждой формы приведены основные компоненты и их назначение.

* + 1. Окно для входа в систему

После запуска программы на экран выводится окно, где необходимо ввести правильные данные и выполнить вход. Вид окна приведен на рисунке 3.1.

*Рисунок 3.1 – Окно для входа*

В данном окне видим два поля для ввода:

«Логин» и «Пароль».

Вводим данные, установленные по умолчанию:

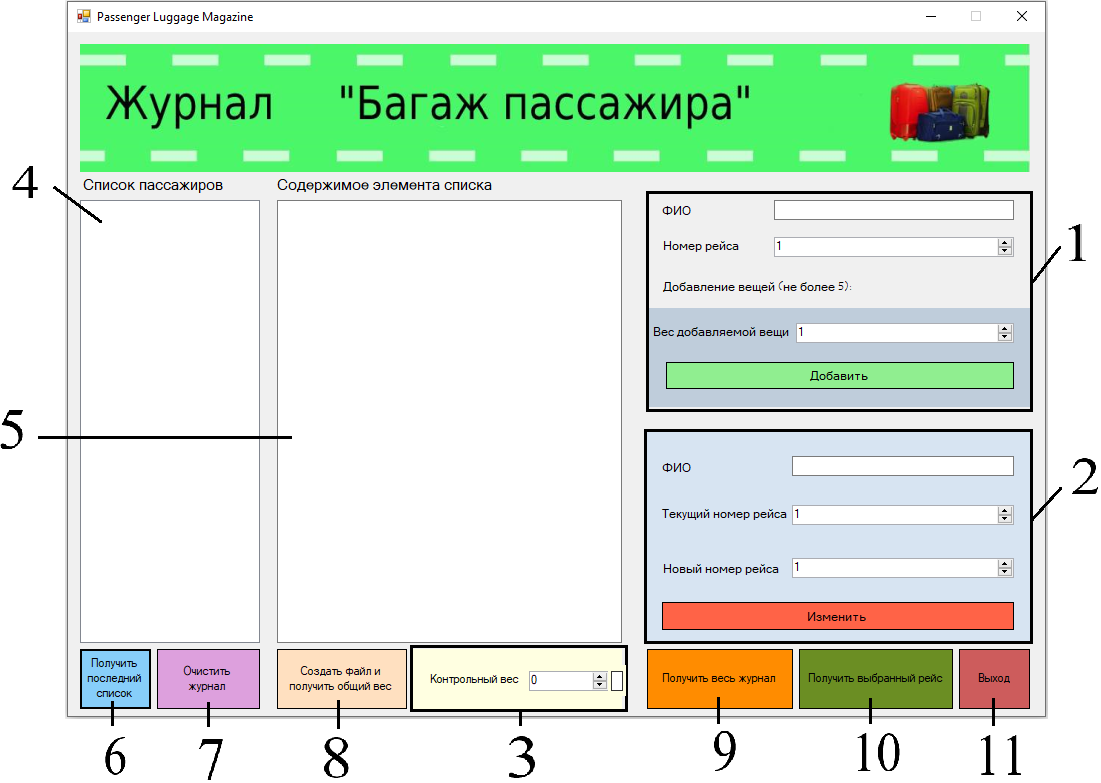
Логин: Noname

Пароль: simple

После ввода данных для входа нажимаем на кнопку «Войти».

3.4.3. Главное окно приложения

После успешного входа вы попадаете на основную «рабочую область» приложения, с которым вам предстоит работать. Вид окна приведён на рисунке 3.2.



*Рисунок 3.2 – Основная рабочая область*

Элементы основного окна:

1 – Модуль приложения для добавления пассажиров и их вещей в список.

2 – Модуль программы для изменения рейса пассажира.

3 – Модуль программы, отвечающий за отбор пассажиров, у которых вес хотя бы одной вещи превышает контрольный.

4 – Список пассажиров.

5 – Содержимое списка пассажиров.

6 – Кнопка для получения списка, с которым работал пользователь в своей прошлой сессии.

7 – Кнопка, очищающая журнал.

8 – Кнопка, создающая файл с общим весом и выводящая его на экран.

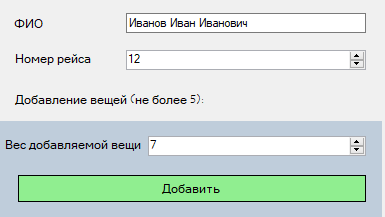
9 – Кнопка, создающая структурированную запись, основанную на списке пассажиров.

10 – Кнопка, действующая аналогично девятой, однако сохраняющая в файл информацию лишь по выбранному рейсу.

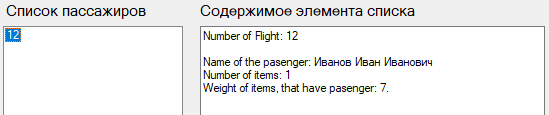
11 – Кнопка, предназначенная для выхода из программы.

Рассмотрим использование каждого из элементов подробнее.

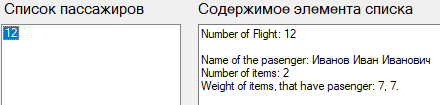
***Первый элемент*** - модуль приложения для добавления пассажиров и их вещей в список. С помощью него происходит добавление элементов в список. Заполненные поля будут выглядеть следующим образом:



После нажатия кнопки «Добавить», происходит добавление элемента в список. Мы можем выделить данный элемент и посмотреть на его содержимое:

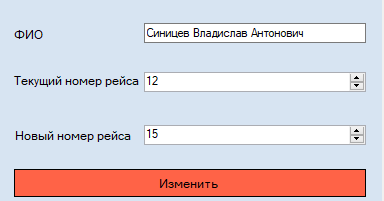


В случае, если вы хотите добавить ещё одну вещь пассажиру, вы должны, не меняя ФИО и номер рейса, повторно нажать кнопку добавить (при желании вес добавляемой вещи можно изменить):



***Второй элемент*** – модуль программы, перемещающий указанного пассажира в другой рейс.

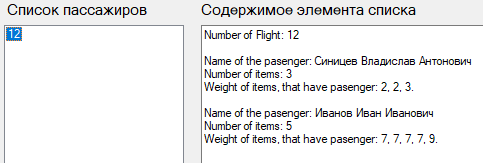
Пример заполнения:



***Третий элемент*** - модуль программы, отвечающий за отбор пассажиров, у которых вес хотя бы одной вещи превышает контрольный.

Пример использования:

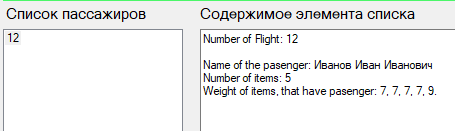
Было:



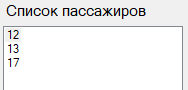
Вписываем контрольный вес и ставим галочку:



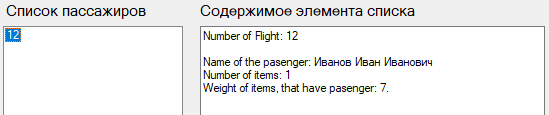
Получаем:



***Четвёртый элемент*** -список пассажиров. Содержит в себе рейсы с пассажирами. Название элемента списка – номер рейса.

Как выглядит:

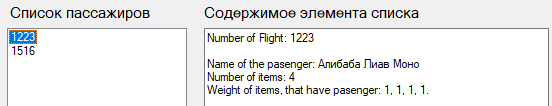
***Пятый элемент*** - содержимое списка пассажиров. При выделении соответствующего элемента списка, в данном поле будет содержаться его содержимое. Пример:



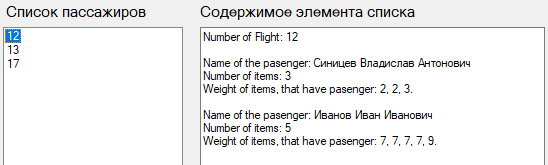
***Шестой элемент*** - кнопка для получения списка, с которым работал пользователь в своей прошлой сессии. При нажатии на данную кнопку нынешний список будет очищен и заменён на тот, с которым вы работали в своей прошлой сессии.

Пример:

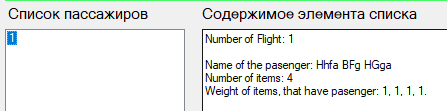
До нажатия:



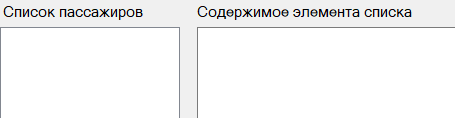
После:



***Седьмой элемент*** - кнопка, очищающая журнал.

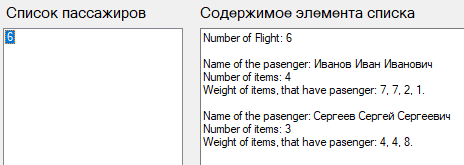
До:

После:

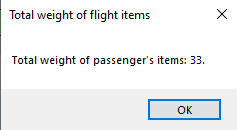
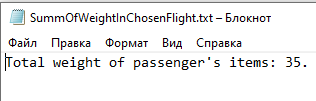


***Восьмой элемент*** - кнопка, создающая файл с общим весом и выводящая его на экран.

Имеем:

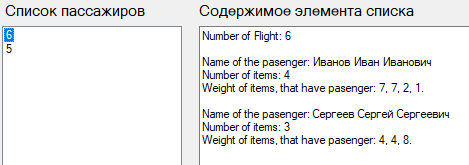


После нажатия получаем:

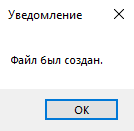
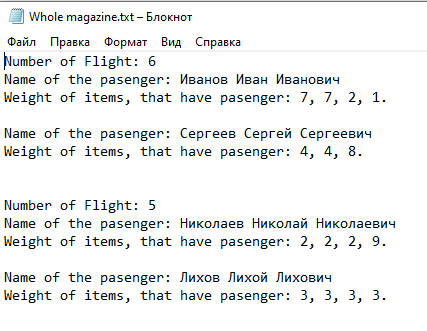
1. Уведомление  
   
2. Файл txt  
   

***Девятый элемент*** - кнопка, создающая структурированную запись, основанную на списке пассажиров.

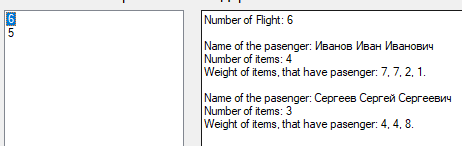
Имеет список:



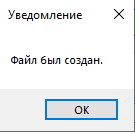
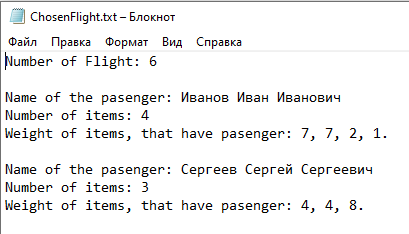
По нажатию кнопки получаем:

1. Уведомление:  
   
2. Файл txt:  
   

***Десятый элемент*** - кнопка, действующая аналогично девятой, однако сохраняющая в файл информацию лишь по выбранному рейсу.

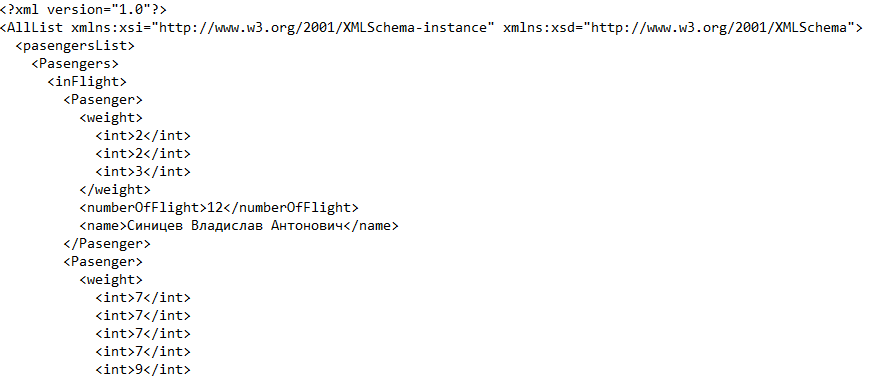
Имеем рейс:

После нажатия кнопки получаем:

1. Уведомление:  
   
2. Файл txt:  
   

***Одиннадцатый элемент*** - кнопка, предназначенная для выхода из программы. При нажатии данной кнопки происходит сохранение текущего списка в файл xml и выход из программы.

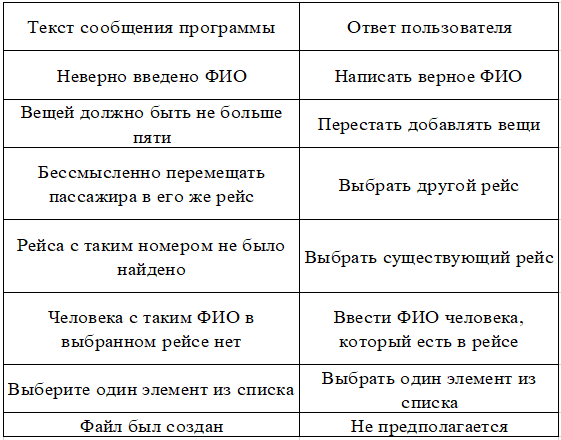
Пример файла xml:



* 1. Сообщения пользователю

При работе с программой могут появиться следующие сообщения, представленные в таблице 3.2.

Таблица 3.2 – сообщения пользователю



* 1. Аварийные ситуации  
     -

1. Руководство системного программиста
   1. Общие сведения

Программа «Passenger\_baggage» предназначена для облегчения создания и ведения журнала для учёта багажей пассажиров – добавление в журнал рейса с определённым номером, пассажира, обозначенного ФИО, количество вещей определённого пассажира и вес каждой из них, а также ведения, удаления, изменения данных.

Входные и выходные данные подробно описаны в п.1 “Постановка задачи”.

Результатом работы программы являются видеограммы и файлы.

Программа реализована на языке С# в среде Visual Studio. Работает в любой среде совместимой с Windows. Дисковой памяти для запуска программы требуется не менее 5 Mb. Оперативной памяти для нормальной работы программы требуется не менее 10 Mb.

Перед эксплуатацией системы необходимости в установки дополнительных пакетов программного обеспечения отсутствует.

* 1. Структура программы

Программа реализована на языке C# в среде Visual Studio, основанном на визуальном построении приложений (помещение компонентов на формы и изменение их свойств и методов).

Модули программы:

Form2 – модуль, предназначенный для входа в главный модуль;

Form1 – главный модуль программа, где и происходит.

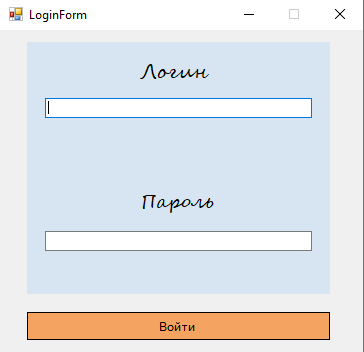
1. Контрольный пример
   1. Запуск программы
      1. После запуска на экран выводится окно авторизации, вид которого представлен на рисунке 5.1.

Рисунок 5.1 – Окно авторизации

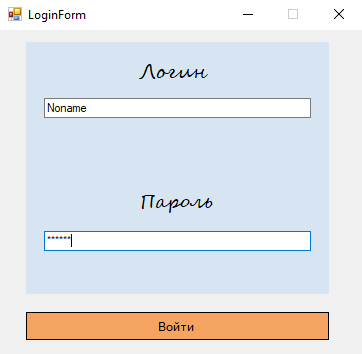
* + 1. Для перехода на главное окно программы, необходимо ввести логин и пароль (см. руководство пользователя), после чего нажать на кнопку «Войти». Выглядеть это будет следующим образом:

Рисунок 5.2 – Заполненное окно авторизации

* + 1. После того, как мы совершим успешный вход, перед нами откроется главное окно программы, с которым мы и будем работать в дальнейшем (см. рис. 5.3).

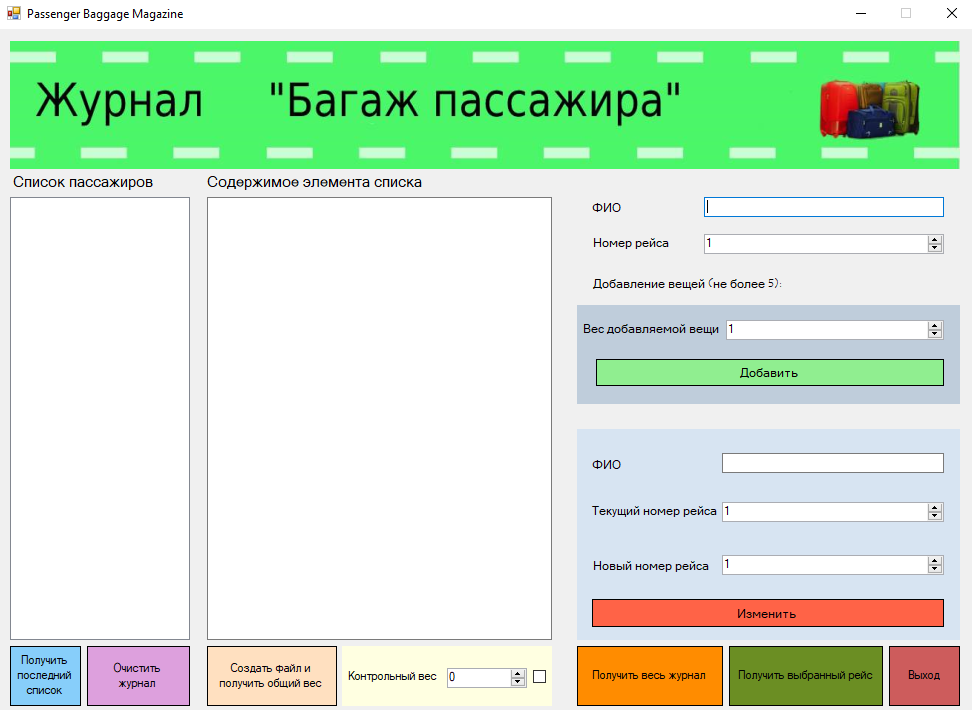


Рисунок 5.3 – Главное окно программы

* + 1. Заполним поля «ФИО», «Номер рейса» и «Вес добавляемой вещи», после чего нажмём на кнопку «Добавить». В результате мы получим первый элемент списка пассажиров, нажав на который мы сможем ознакомиться с его содержимым (см. рис. 5.4).

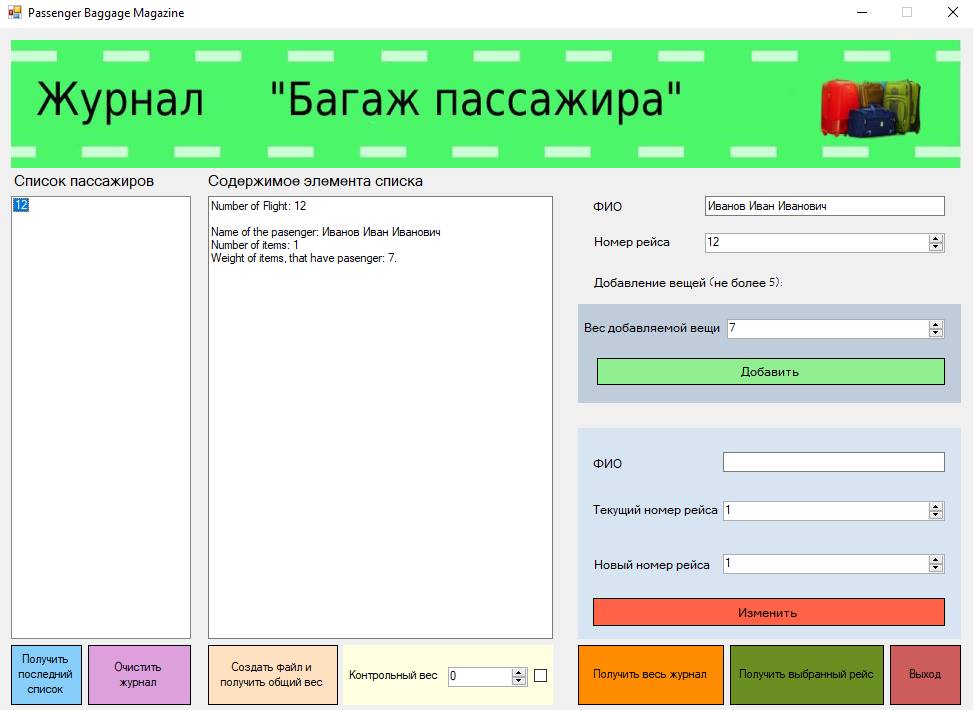


Рисунок 5.4 – Добавление первого элемента списка

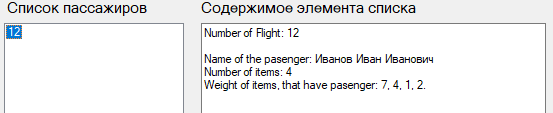
* + 1. Для добавления новых вещей к уже имеющемуся пассажиру, нужно, не меняя ФИО и номер рейса, изменить (по желанию) вес добавляемой вещи и снова нажать на кнопку «Добавить» (см. рис. 5.5).

Рисунок 5.5 – Добавление вещей для пассажира

* + 1. Для добавления нового пассажира в уже имеющийся рейс, нужно, не меняя номер рейса, изменить ФИО и нажать на кнопку «Добавить» (по желанию вес добавляемой вещи можно изменить). Если же вы хотите добавить новый рейс в список, вы должны изменить номер рейса и нажать кнопку «Добавить». Пример списка из нескольких рейсов вы можете увидеть на рисунке 5.6.

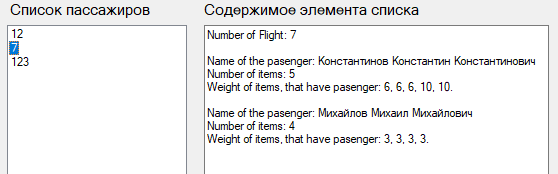
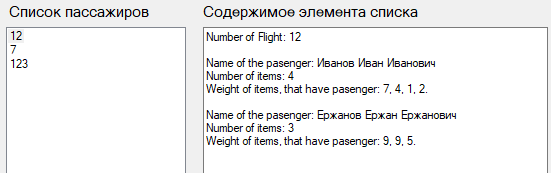


Рисунок 5.6 – Список пассажиров, состоящий из нескольких элементов

* + 1. Изменим рейс одного из пассажиров. Для этого заполним поля «ФИО», «Текущий номер рейса», «Новый номер рейса» и нажмём на кнопку «Изменить». Пример такого перемещения можно наблюдать на рисунке 5.7.



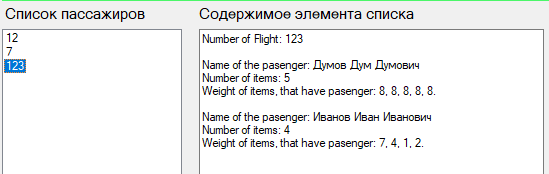
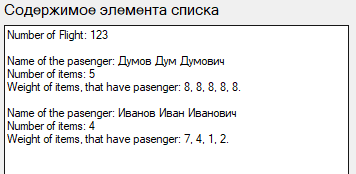


Рисунок 5.7 – Изменение рейса у пассажира Иванова И.И.

* + 1. Рассмотрим опцию «Контрольный вес» (см. рис. 5.8). При проставлении галочки в квадратик, в поле «Содержимое элемента списка» остаются только те пассажиры, у которых вес хотя бы одно вещи превышает контрольный. На примере рисунка 5.9 можно увидеть, как изменился список при контрольном весе, равном 7.

Рисунок 5.8 – Опция «Контрольный вес»



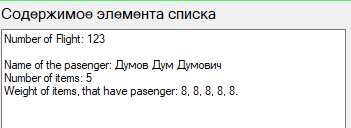


Рисунок 5.9 – Изменение содержимого после проставления галочки

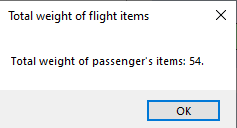
* + 1. При нажатии кнопки «Создать файл и получить общий вес», на экран будет выдана информация, содержащая общий вес всех вещей в выбранном рейса (см. рис. 5.10), и будет создан текстовый файл с аналогичной информацией (см. рис. 5.11).

Рисунок 5.10 – Уведомление с информацией об общем весе

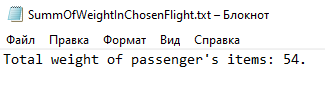
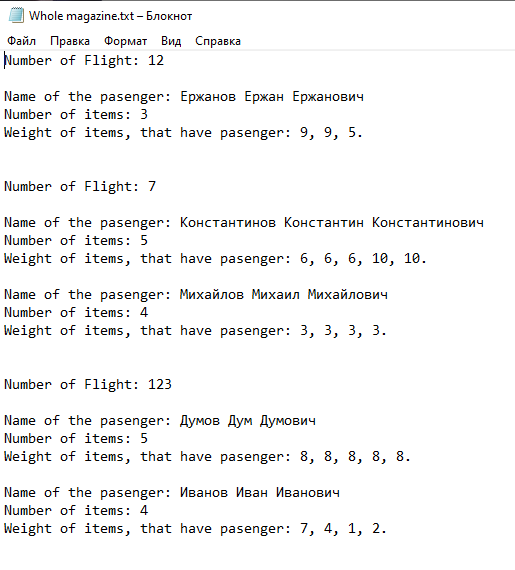


Рисунок 5.11 – Файл с информацией об общем весе

* + 1. При нажатии на кнопку «Получить весь журнал», будет получен весь список в виде файла txt (см. рис. 5.12) и получено уведомление о создании файла (см. рис. 5.13).

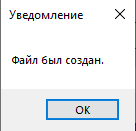
Рисунок 5.12 – Полученный полный список в виде файла txt

Рисунок 5.13 – Уведомление о создании файла

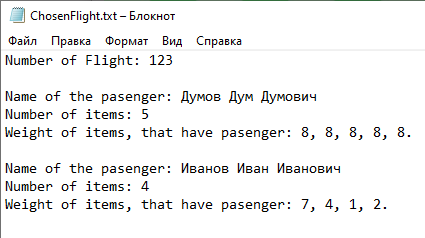
* + 1. При нажатии на кнопку «Получить выбранный рейс», будет получено уведомление о создании файла (см. рис. 5.13) и создан файл, в котором будет храниться содержимое выбранного элемента из списка в виде файла txt (см. рис. 5.14).

Рисунок 5.14 – Файл txt с содержимым выбранного элемента

* + 1. При нажатии на кнопку «Выход», будет произведено сохранение нынешнего списка в виде файла xml (см. рис. 5.15) и выход из программы.

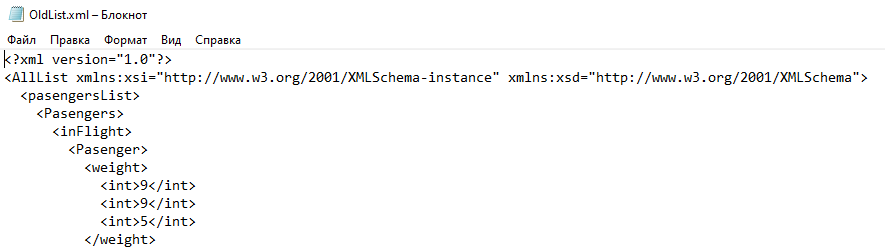


Рисунок 5.15 – Фрагмент файла xml

* + 1. После выполнения п. 5.1.12 появится возможность восстановить последний список при повторном запуске программы. Для этого необходимо нажать на кнопку «Получить последний список». Пример работы данной кнопки вы можете увидеть на рисунке 5.16.

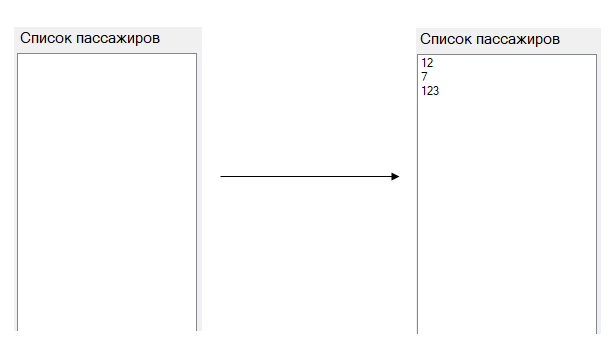


Рисунок 5.16 – Восстановление списка пассажиров

ЗАКЛЮЧЕНИЕ

Во время выполнения курсовой работы были изучены: работа с Windows Forms на C# (работа с элементами формы, их свойствами, создание дополнительных форм, добавление к элементам формы какого-либо функционала и т.п.), сериализация XML (более углубленно), всевозможная работа с файлами и т.д.

В результате выполнения курсовой работы было разработано приложение с простым интерфейсом, позволяющее вести журнал и сохранять его. К достоинствам разработанного приложения можно отнести понятный и красочный интерфейс, удобную работу с журналом, возможность восстановить журнал с последней рабочей сессии и др. К недостаткам же возможно отнести единственный верный логин и пароль, отсутствие базы данных.

В дальнейшем приложение может быть дополнено базой данных, благодаря которой приложение станет ещё удобнее и избавится от ряда недостатков.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ