Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему:

«Программное средство “Электронный кошелек”

(Программа учета доходов и расходов)»

БГУИР КП 1-40 01 01  07  10

Студент: гр. 751007 Плескач К.С.

Руководитель: асс. Болтак С.В.

Минск 2018

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ПОИТ

––––––––––––––––––––––––

(подпись)

Лапицкая Н.В. 2018 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту    –––––––––––––

1. Тема работы    ––––

2. Срок сдачи студентом законченной работы 23.05.2018

3. Исходные данные к работе   .

4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)

*Введение.*

*1.Аналитический обзор литературы и существующих аналогов;*

*2.Разработка алгоритма;*

*3. Разработка программного средства;*

*4. Обоснование технических приемов программирования;*

*5. Тестирование, экспериментальные исследования и анализ полученных результатов;*

*6. Руководство пользователя программы;*

*Заключение, список литературы, ведомость, приложения.*

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. Схема программы

6. Консультант по курсовому проекту Болтак С.В.

7. Дата выдачи задания 28.02.2018 г.––   –

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

раздел 1, введение к 02.03.2018 – 10 % готовности работы;

разделы 2 к 15.03.2018 – 30 % готовности работы;

разделы 3,4 к 15.04.2018 – 60 % готовности работы;

раздел 5, 6 к 05.05.2018 – 90 % готовности работы;

оформление пояснительной записки и графического материала к 20.05.2018 – 100 % готовности работы.

Защита курсового проекта с 23.05 по 12.06 2018 г.

РУКОВОДИТЕЛЬ С.В. Болтак

(подпись)

Задание принял к исполнению –––\_\_\_\_––     28.02.2018 г.

(дата и подпись студента)

**СОДЕРЖАНИЕ**

Введение ........................................................................................................ 5

1 Анализ прототипов, литературных источников и формирование требований и проектируемому ПС .................................................................................... 6

1.1 Анализ существующих аналогов....................................................... 6

1.2 Формирование требований к проектируемому ПС.......................... 8

2 Разработка программного средства…...................................................... 9

2.1 Разработка функциональных требований ........................................ 9

2.2 Необходимые ресурсы для курсовой работы ................................... 9

2.2.1 Выбор языка программирования и среды разработки…….... 9

2.2.2 Описание используемых типов данных................................... 9

3 Проектирование программного средства................................................. 10

3.1 Разработка основного алгоритма........................................................ 10

4 Тестирование ............................................................................................. 14

5 Руководство пользователя ........................................................................ 18

Заключение..............................................................................…................... 19

Список использованных источников .......................................................... 20

Приложение А................................................................................................. 21

**ВВЕДЕНИЕ**

На сегодняшний день самой популярной техникой для дома и офиса является персональный компьютер. Его можно использовать и для отдыха, и для работы. С его помощью можно просматривать видео, работать с документами, обрабатывать фото и видеофайлы, играть в игры и использовать в качестве развлекательного центра.

С чем бы мы ни сталкивались в повседневной жизни, всё работает на базе компьютеров. Банковские системы, торговые сети, сфера образования, медицина, да и любая другая сфера, так или иначе касающаяся нашей жизни, требует использование ПК для обеспечения своей работоспособности. Но самый главный плюс техники подобного рода даже не в этом.

Персональный компьютер существенно экономии наше время на такие банальные операции, как расчет своих доходов и расходов. Больше нет необходимости тратить бумагу и исписывать последнюю ручку. Все может быть сделано за вас.

В настоящее время одним из важнейшим аспектом жизни является возможность держать свои финансы под контролем. С помощью специальных программ можно следить за тратами, формировать бюджет и всегда быть в курсе того, сколько у вас имеется сбережений, фиксировать ваши долговые обязательства и постоянные выплаты, а также предоставляет исчерпывающую статистику за любой интересующий вас период времени.

Следовательно целью работы является создать программный продукт, способный, опираясь на внесенные траты и доходы:

1) Редактировать список приходных и расходных статей, добавлять и удалять статьи, выводить в зависимости от выбранного периода;

2) Редактировать, добавлять и удалять долги и займы, выводить в зависимости от выбранного периода;

3) Создавать, редактировать, удалять категории доходов/расходов и валюту;

4) Вести расчет количества наличности «на руках», потраченной, остаток;

5) Возможность использовать калькулятор при внесении расходов/доходов;

6) Отображать графически катерогии и суммы расходов, доходов; имена и суммы займов и долгов.

**1 АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ**

**И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПС**

## Анализ существующих аналогов

В настоящее время на рынке наблюдается подлинное многообразие автоматизированных систем учета доходов, некоторые из них более удобны, некоторые менее.

Например, программа “Cемейный бюджет Lite” призвана разгрузить ваши муки в отношении подсчета личных трат. От вас требуется лишь ввести собственные доходы и расходы в соответствующие колонки. Все остальные операции программа сделает самостоятельно. Преимущества клиента следующие: учитывается доходность по нескольким основным категориям и счетам; можно заниматься учетом собственных долгов, кредитов, инвестиций, вкладов и прочих расчетов; вы можете воспользоваться функцией авто-категорий, т.е. при вводе названий товара программа будет автоматически подбирать требуемую категорию из таблицы;

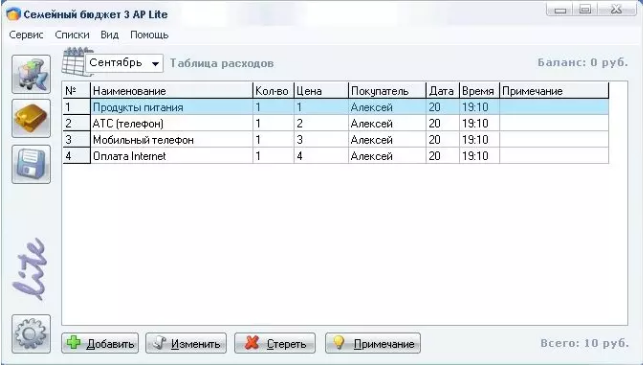


рис1.1.1 Приложение Семейный бюджет

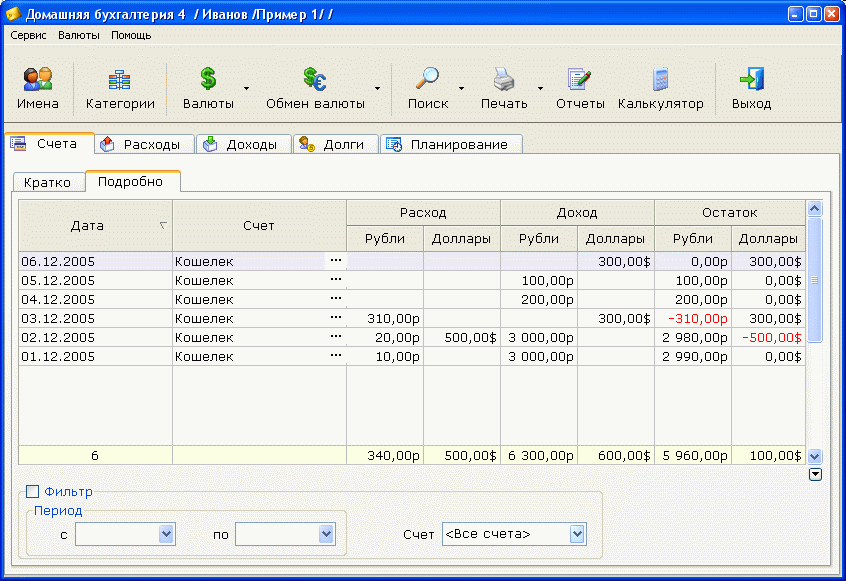
Так же среди плюсов можно заметить и явные недостатки, такие как использование только трех видов валюты, нет разбиения по подкатегориям, интуитивно не самый удобный интерфейс.

Среди подобных программ можно также выделить программу «Домашняя бухгалтерия», которая является одной из лидирующей на рынке.

Как достоинством, так и недостатком данной программы можно выделить огромнейшее количество функций как учета, так и планирования своих финансов.

Клиентом может пользоваться одновременно несколько человек. При этом каждый будет обладать собственной учетной записью и паролем. Последний можно установить при запуске приложения. Поиск доходов и расходов очень удобен, поскольку есть возможность настроить выдачу сразу по нескольким фильтрам: товар, дата, категория и т.д.

рис 1.1.2 Приложение Домашняя бухгалтерия



Просмотрев функционал данных программ, было решено создать программу для операционной системы Windows, которая будет выполнять основные функции, иметь простой и понятный каждому пользователю интерфейс, будут доступны дополнительные возможности и которая будет лишена данных недостатков.

Последним примером можно привести приложение AbilityCash.

Основными достоинствами является неограниченное количество счетов, быстрый обзор текущего периода на экране сводки, защита доступа к данным через пароль.

Минусами является однопользовательский режим и полный контроль над созданием категорий и подкатегорий, а так же, как и у большинства программ данного типа, большая нагроможденность, что создаст непривычному человеку большое количество неудобств и дискомфорта.

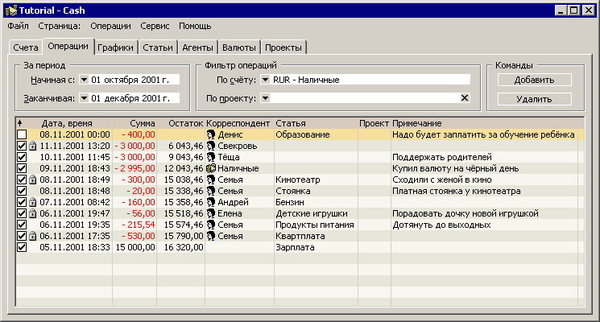


рис 1.1.3 Приложение AbilityCash

* 1. **Формирование требований к проектируемому программному средству.**

Создать в среде Delphi приложение, способное вести калькуляцию расходов и доходов, добавлять и редактировать расходы и доходы, долги и займы с вводом текущей даты, выводить на экран доходы и расходы, долги и займы за определенный период времени, введенный пользователем, защищать хранящиеся данные паролем, имеющее многопользовательский режим, способное добавлять и редактировать как имя пользователя, так и пароль в процессе работы, имеющее основной набор функций для приложения учетов расходов и доходов, имеющее простой и понятный пользовательский интерфейс.

Так же у программы должна иметься функция графического отображения информации. Для удобства работы также планируется реализовать процедуру, удаляющая все данные из всех используемых файлов.

Программа должна обладать удобным интерфейсом, эффективно использовать память и обладать по возможности большей скоростью работы.

**2 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА**

**2.1 Разработка функциональных требований.**

Итоговая программа должна обладать следующими функциями:

1. Ввод пользователем своего имени и пароля;
2. Добавление, редактирование категорий и валюты;
3. Добавление, редактирование расходов и доходов, долгов и займов;
4. Вывод на главной странице количества наличности на руках, протраченной и оставщейся;
5. Возможность пользоваться калькулятором в процессе работы;
6. Отображать расходы и доходы, долги и займы за выбранный период времени;
7. Графическое отображение расходов, доходов, долгов и займов;
8. Очищение всех используемых типизированных файлов;
9. Сохранение данных о каждом пользователе в типизированный файл для долговременной работы с приложением.
   1. **Необходимые ресурсы для курсовой работы**

**2.2.1** Выбор языка программирования и среды разработки

В качестве языка программирования был выбран язык Object Pascal, а средой разработки – Borland Delphi, изучаемые на занятиях по дисциплине «Основы алгоритмизации и программирования».

**2.2.2. Описание используемых типов данных.**

Приложение создается на формах, с использованием в основном таких компонентов как StringGrid, Labels для отображения данных и комфортного использования программного средства. Основным требованием к разрабатываемому ПС является отображение информации, хранящихся в файлах на компьютере пользователя. Для этой цели активно применялись таблицы StringGrid.

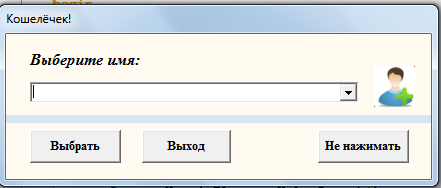
Для управления программным средством пользователем использовались такие компоненты как Button, BitButton, Edit, ComboBox и другие. При нажатии на интуитивно понятные компоненты управления происходит взаимодействие человека с программным средством. При каждом нажатии компоненты отображения изменяются и дают пользователю представление о своих расходах и доходов.

Для отображения диаграмм использовался встроенный компонент Delphi TChart. При добавлении в него информации, она отображается в виде круговой схемы.

**3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА.**

**3.1. Разработка основного алгоритма.**

При запуске программного средства должно показывать окно, которое ждет заполнения данных. Список возможных действий:



Кнопки на экране:

1. Выбор пользователя и переход к работе с приложением
2. Добавление, редактирование и удаление пользователей
3. Кнопка, которая убирает все окна со страницы
4. Кнопка, которая очищает файлы

Рассмотрим подробнее основные операции:

1. Редактирование пользователей.

При запуске ПС из файла с информацией считываются ранее введенные пользователи и заносятся в однонаправленный список пользователей, состоящий из имени пользователя и пароля. Изначально в компоненте ComboBox отображен список доступных пользователей. При нажатии на BitButtonVibor открывается новая форма для редактирования. Рассмотрим ее основные процедуры:

Таблица 3.1.1 – описание процедур и функций формы «Администратор пользователей».

|  |  |  |
| --- | --- | --- |
| №  п.п | Название подпрограммы | Назначение подпрограммы |
| 1 | procedure TAdminNames.FormShow(Sender: TObject); | Заполняет таблицу исходными данными о пользователях и их количестве при каждом показе формы |
| 2 | procedure TAdminNames.btnAddClick(Sender: TObject); | Открывает форму для добавления нового пользователя, заносит его в список |
| 3 | procedure TAdminNames.btnChangeClick(Sender: TObject); | Открывает форму для изменения информации о выбранном пользователе, заносит изменения в список |
| 4 | procedure TAdminNames.btnDeleteClick(Sender: TObject); | Удаляет выбранного пользователя и убирает его из списка |
| 5 | procedure TAdminNames.btnCloseClick(Sender: TObject); | Закрывает форму для редактирования пользователей и переходит на начальную форму для выбора пользователя |

2) Выбор пользователя и переход к работе с приложением

При выборе пользователя программа изначально ждет получения пароля, далее открывает главное окно программы, на которой располагается таблица с данными о текущем состоянии наличности на руках и большое количество кнопок для внесения новых доходов, расходов, долгов и займов с выбранной категорией, валютой и датой.

Таблица 3.2 – основные процедуры и функции главной формы программного средства

|  |  |  |
| --- | --- | --- |
| №  п.п | Название подпрограммы | Назначение подпрограммы |
| 6 | procedure TMainForm.btnExitClick(Sender: TObject); | Выходит из программного средства |
| 7 | procedure TMainForm.btnNamesClick(Sender: TObject); | Возвращается к первой форме для выбора другого пользователя |
| 8 | procedure TMainForm.btnMainClick(Sender: TObject); | Показывает главную таблицу с количеством наличных на руках |
| 9 | procedure TMainForm.btnValutClick(Sender: TObject); | Открывает форму для редактирования валют |
| 10 | procedure TMainForm.btnCategoryClick(Sender: TObject); | Открывает форму для редактирования категорий |
| 11 | procedure TMainForm.btnCalculateClick(Sender: TObject); | Открывает калькулятор |
| 12 | procedure TMainForm.btnRashodClick(Sender: TObject); | Показывает таблицу расходов |
| 13 | procedure TMainForm.btnDohodClick(Sender: TObject); | Показывает таблицу доходов |
| 14 | procedure TMainForm.btnZaemClick(Sender: TObject); | Показывает таблицу займов |
| 15 | procedure TMainForm.btnDolgClick(Sender: TObject); | Показывает таблицу долгов |
| 16 | procedure TMainForm.tablRashodMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Вычисляет координаты нажатого расхода |
| 17 | procedure TMainForm.btnAddClick(Sender: TObject); | Добавляет расход в таблицу |
| 18 | procedure TMainForm.btnChangeClick(Sender: TObject); | Изменяет выбранный расход |
| 19 | procedure TMainForm.btnDeleteClick(Sender: TObject); | Удаляет выбранный расход |
| 20 | procedure TMainForm.btnDavaiClick(Sender: TObject); | Показывает расходы в текущем временном интервале |
| 21 | procedure TMainForm.tablDohodMouseDown(Sender: TObject;  Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Вычисляет координаты нажатого дохода |
| 22 | procedure TMainForm.btnDeleteDohodClick(Sender: TObject); | Удаляет выбранный доход |
| 23 | procedure TMainForm.btnDavaiDohodClick(Sender: TObject); | Показывает доходы в текущем временном интервале |
| 24 | procedure TMainForm.btnAddDohodClick(Sender: TObject); | Добавляет доход |
| 25 | procedure TMainForm.btnChangeDohodClick(Sender: TObject); | Изменяет выбранный доход |
| 26 | procedure TMainForm.tablDolgiMouseDown(Sender: TObject;  Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Вычисляет координаты нажатого долга |
| 27 | procedure TMainForm.btnDeleteDolgiClick(Sender: TObject); | Удаляет выбранный доход |
| 28 | procedure TMainForm.btnDavaiDolgiClick(Sender: TObject); | Показывает долги в текущем временном интервале |
| 29 | procedure TMainForm.btnAddDolgiClick(Sender: TObject); | Добавляет долг |
| 30 | procedure TMainForm.btnChangeDolgiClick(Sender: TObject); | Изменяет нажатый долг |
| 31 | procedure TMainForm.tablZaemiMouseDown(Sender: TObject;  Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Вычисляет координаты нажатого займа |
| 32 | procedure TMainForm.btnAddZaemiClick(Sender: TObject); | Добавляет займ |
| 33 | procedure TMainForm.btnChangeZaemiClick(Sender: TObject); | Изменяет нажатый займ |
| 34 | procedure TMainForm.btnDeleteZaemiClick(Sender: TObject); | Удаляет нажатый займ |
| 35 | procedure TMainForm.btnDavaiZaemiClick(Sender: TObject); | Показывает займы в текущем временном интервале |
| 36 | procedure TPassword.btnNoooClick(Sender: TObject); | Очищает используемые файлы |
| 37 | procedure btnOtchetClick(Sender: TObject); | Открывает форму, в которой идет отображение расходов, доходов, долгов и займов в зависимости от выбранной валюты |

Большинство процедур открывает новые формы, для реализации того или иного действия, а именно: список категорий (удаление, редактирование удаление), список валют (добавление, удаление), карточки для ввода новых или редактирования старых доходов, расходов, долгов и займов.

В таблицах можно увидеть информацию как об отдельных доходах, расходах, о итоговой сумме доходов, расходов и их отображение в зависимости от временного интервала.

Все компоненты понятно подписаны, указывая пользователю, какие функции она выполняет.

**4** **ТЕСТИРОВАНИЕ**

Таблица 4.1 – тестирование программы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тесткейсы и шаги | Ожидаемые результаты | Полученные результаты |
| 1 | Нажать на кнопку с человеком для редактирования пользователей | Откроется форма с заполненной таблицей |  |
| 2 | Нажать на кнопку добавить | Откроется форма для добавления пользователя |  |
| 3 | Заполняем поля, нажимаем готово | Видим, что набранный пользователь отобразился в таблице |  |
| 4 | С помощью кнопки изменить переименуем мама в наташа и удалим карина | Отображение полученных данных в таблице |  |
| 5 | Возвращаемся на первую форму | Отображение данных в ComboBox |  |

Продолжение таблицы 4.1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | | Тесткейсы и шаги | | Ожидаемые результаты | | Полученные результаты | |
| 6 | | Выбираем имя Наташа | | Программа ждет подтверждения пароля | |  | |
| 7 | | Вводим пароль | | Видим главную таблицу с заполненными данными о пользователе | |  | |
| 8 | | Нажмем категории | | Форма со списками категорий доходов и расходов | |  | |
| 9 | | Добавим категорию доходов халтура, удалим остальное, добавим категорию расхода телефон, изменим остальное на телефон | | Видим изменившиеся данные | |  | |
| 10 | | Нажмем валюты | | Форма со списком используемых валют | |  | |
| 11 | | Удалим динар и добавим египетский фунт, вернемся обратно | | Видим изменившуюся главную таблицу | |  | |
| 12 | | Нажмем калькулятор | | Откроется калькулятор | |  | |
| 13 | | Нажмем расходы | | Появится таблица со списком расходов | |  | |
| 14 | | Добавим пару расходов | | Видим изменившиеся данные в таблице | |  | |
| 15 | | Поставим фильтр с 08.05.2018 по 05.06.2018 | | Данные в таблице отсортировались по выбранным датам | |  | |
| 16 | | Попробуем удалить валюту Египетский фунт | | Изначально увидим предупреждение, что можно потерять данные | |  | |
| 17 | | Переходим на таблицу расходом | | Видим, что категория праздники вместе с остальными данными удалилась | |  | |
| 18 | | Перейдем к отчетам, нажав на соответствующую кнопку и выберем доходы, белорусский рубль | | Видим нарисованную круговую диаграмму | |  | |

**5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

Для успешной работы с программой изначально внесите свое имя в список пользователей и запомните пароль от входа в свой аккаунт.

Выберите свое имя на первой форме, введите пароль. Перед вами откроется главная форма, на которой расположены все основные действия программы. Если рассматривать кнопки слева направо, то первая кнопка позволит вам вернуться на первую форму и выбрать другого пользователя. Следующая позволит изменить категории расходов и доходов. Для этого в появившемся окне в нужной колонне выберите одно из действий: добавить, изменить, удалить. Данные изменения сразу отобразятся в соответствующих таблицах. Следующая кнопка действует схоже, но только со списком валют. Следующая откроет новую форму, в которой на диаграмме будет показана статистика того, на какую кнопку внизу вы нажмете, при этом также надо выбрать валюту, по которой надо строить диаграмму. Следующая – откроет калькулятор. Последняя – завершит работу программы.

Следующий ряд кнопок отвечает за главные таблицы, отображающие информацию о приходных и расходных статьях. При нажатии на вторую и последующие кнопки вам откроется свойство добавления, изменения и удаления расходов, доходов, долгов и займов. Так же внизу появятся два поля ввода дат, установление которых приведет к отсортировке статей, которые были сделаны в промежутке между этими датами.

Таким образом, с помощью данного программного средства вы сможете наглядно оценить свои финансовые возможности и контролировать свои расходы и доходы.**ЗАКЛЮЧЕНИЕ**

В результате работы над курсовым проектом было создано рабочее приложение, которое является результатом процесса, включающего анализ поставленной задачи, работу с литературой, проектирование структур данных, разработку алгоритмов, тестирование созданного приложения, составление руководство пользователя.

Приложение предоставляет пользователю все требуемые согласно поставленной задаче гарантии и возможности.

Разработанное приложение включало в себе решения множества задач и проблем. Чтобы преодолеть их было изучено большое количество функций и процедур, проведен их анализ, и сформированы требования для приложений подобного рода. Были изучены методы работы с типизированными файлами, различными компонентами delphi. Далее были детализированы все функции.

Приложение было подвергнуто всевозможным тестам, в ходе которых неисправности обнаружены не были. Количество тестов было достаточным для того, чтобы сделать вывод об эффективной работе программного средства.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Интернет ресурс <http://www.delphisources.ru>

[2] Фаронов В.В. — Delphi. Программирование на языке высокого уровня (2004)

[3] Основы алгоритмизации и программирования Часть 2

[4] Приложение «Домашняя бухгалтерия»

[5] Интернет ресурс Уроки Delphi начинающим с нуля <http://www.delphi-manual.ru>

**ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Текст программного модуля**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Buttons, ExtCtrls, mmsystem,Grids;

type

TPassword = class(TForm)

lbVibor: TLabel;

cbVibor: TComboBox;

btnVibor: TBitBtn;

spl1: TSplitter;

btnOk: TButton;

btnExit: TButton;

btnNooo: TButton;

procedure btnExitClick(Sender: TObject);

procedure btnViborClick(Sender: TObject);

procedure btnOkClick(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure btnNoooClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Password: TPassword;

type

Elements = record

Name, Password: string[30];

KolvoCategoryDohodov, KolvoCategoryRashodov, KolvoDolgnikov, KolvoZaemchikov, KolvoDohodov, KolvoRashodov, KolvoValut: integer;

MoneyYouHave, AreYouNewOne: Integer;

end;

AdrUsers = ^TUsers;

TUsers = record

Element: Elements;

Next: AdrUsers;

end;

var

FileWithUsers: file of Elements;

FirstUser, CurrentUser: AdrUsers;

DataUser: Elements;

i, j: Integer;

s: string;

implementation

uses Unit2, Unit4, Unit6, Unit8, Unit3, Unit5, Unit7, Unit9, Unit10, Unit11,Unit12,Unit13,Unit14,Unit15, Unit16, Unit17;

{$R \*.dfm}

procedure TPassword.btnViborClick(Sender: TObject);

begin

PlaySound('D:\ó÷åáà\ÎÀÈÏ\ëàáû\4\_ïðîöåäóðíûé òèï\button.wav', 0, SND\_ASYNC);

AdminNames.Show;

Password.Hide;

end;

procedure TPassword.btnOkClick(Sender: TObject);

begin

PlaySound('D:\ó÷åáà\ÎÀÈÏ\ëàáû\4\_ïðîöåäóðíûé òèï\button.wav', 0, SND\_ASYNC);

s := cbVibor.Items[cbVibor.ItemIndex];

if Length(s) <> 0

then

begin

ViFormMain.Show;

Password.Hide;

end

else

begin

ShowMessage('Âûáåðèòå ïîëüçîâàòåëÿ.');

cbVibor.Text := '';

end;

end;

procedure TPassword.btnExitClick(Sender: TObject);

var

i: Integer;

begin

PlaySound('D:\ó÷åáà\ÎÀÈÏ\ëàáû\4\_ïðîöåäóðíûé òèï\1.wav', 0, SND\_ASYNC);

i := 0;

while i<10 do

begin

sleep(45);

Application.ProcessMessages;

inc(i);

end;

Password.Close;

AdminNames.Close;

FormName.Close;

Change.Close;

Vi.Close;

ViForm.Close;

ViFormMain.Close;

MainForm.Close;

NastroikaValut.Close;

AddValut.Close;

Attantion.Close;

NastroikaCategory.Close;

AddDohod.Close;

AddRashod.Close;

ChangeDohod.Close;

ChangeRashod.Close;

end;

procedure TPassword.FormShow(Sender: TObject);

begin

cbVibor.Text := '';

cbVibor.Items.Clear;

AssignFile(FileWithCategoryDohodov, 'CategoryDohodov.txt');

try

Reset(FileWithCategoryDohodov);

New(FirstCategoryDohodov);

CurrentCategoryDohodov := FirstCategoryDohodov;

CurrentCategoryDohodov^.Next := nil;

while not EOF(FileWithCategoryDohodov) do

begin

Read(FileWithCategoryDohodov, DataCategoryDohodov);

New(CurrentCategoryDohodov^.Next);

CurrentCategoryDohodov := CurrentCategoryDohodov^.Next;

CurrentCategoryDohodov^.Element.Name := DataCategoryDohodov.Name;

CurrentCategoryDohodov^.Element.User := DataCategoryDohodov.User;

CurrentCategoryDohodov^.Next := nil;

end;

except

Rewrite(FileWithCategoryDohodov);

end;

CloseFile(FileWithCategoryDohodov);

AssignFile(FileWithCategoryRachodov, 'CategoryRashodov.txt');

try

Reset(FileWithCategoryRachodov);

New(FirstCategoryRachodov);

CurrentCategoryRachodov := FirstCategoryRachodov;

CurrentCategoryRachodov^.Next := nil;

while not EOF(FileWithCategoryRachodov) do

begin

Read(FileWithCategoryRachodov, DataCategoryRachodov);

New(CurrentCategoryRachodov^.Next);

CurrentCategoryRachodov := CurrentCategoryRachodov^.Next;

CurrentCategoryRachodov^.Element.Name := DataCategoryRachodov.Name;

CurrentCategoryRachodov^.Element.User := DataCategoryRachodov.User;

CurrentCategoryRachodov^.Next := nil;

end;

except

Rewrite(FileWithCategoryRachodov);

end;

CloseFile(FileWithCategoryRachodov);

AssignFile(FileWithValuta,'Valuta.txt');

try

Reset(FileWithValuta);

New(FirstValuta);

CurrentValuta := FirstValuta;

CurrentValuta^.Next := nil;

while not EOF(FileWithValuta) do

begin

Read(FileWithValuta, DataValuta);

New(CurrentValuta^.Next);

CurrentValuta := CurrentValuta^.Next;

CurrentValuta^.Element.Name := DataValuta.Name;

CurrentValuta^.Element.User := DataValuta.User;

CurrentValuta^.Next := nil;

end;

except

Rewrite(FileWithValuta);

end;

CloseFile(FileWithValuta);

AssignFile(FileWithRashod,'Rashod.txt');

try

Reset(FileWithRashod);

New(FirstRashod);

CurrentRashod := FirstRashod;

CurrentRashod^.Next := nil;

while not EOF(FileWithRashod) do

begin

Read(FileWithRashod, DataRashod);

New(CurrentRashod^.Next);

CurrentRashod := CurrentRashod^.Next;

CurrentRashod^.Element.Money := DataRashod.Money;

CurrentRashod^.Element.Valuta := DataRashod.Valuta;

CurrentRashod^.Element.Category := DataRashod.Category;

CurrentRashod^.Element.Day := DataRashod.Day;

CurrentRashod^.Element.User := DataRashod.User;

CurrentRashod^.Next := nil;

end;

except

Rewrite(FileWithRashod);

end;

CloseFile(FileWithRashod);

AssignFile(FileWithDohod,'Dohod.txt');

try

Reset(FileWithDohod);

New(FirstDohod);

CurrentDohod := FirstDohod;

CurrentDohod^.Next := nil;

while not EOF(FileWithDohod) do

begin

Read(FileWithDohod, DataDohod);

New(CurrentDohod^.Next);

CurrentDohod := CurrentDohod^.Next;

CurrentDohod^.Element.Money := DataDohod.money;

CurrentDohod^.Element.Valuta := DataDohod.Valuta;

CurrentDohod^.Element.Category := DataDohod.Category;

CurrentDohod^.Element.Day := DataDohod.Day;

CurrentDohod^.Element.User := DataDohod.User;

CurrentDohod^.Next := nil;

end;

except

Rewrite(FileWithDohod);

end;

CloseFile(FileWithDohod);

AssignFile(FileWithDolgi,'Dolgi.txt');

try

Reset(FileWithDolgi);

New(FirstDolgi);

CurrentDolgi := FirstDolgi;

CurrentDolgi^.Next := nil;

while not EOF(FileWithDolgi) do

begin

Read(FileWithDolgi, DataDolgi);

New(CurrentDolgi^.Next);

CurrentDolgi := CurrentDolgi^.Next;

CurrentDolgi^.Element.Name := DataDolgi.Name;

CurrentDolgi^.Element.Money := DataDolgi.Money;

CurrentDolgi^.Element.Valuta := DataDolgi.Valuta;

CurrentDolgi^.Element.Day := DataDolgi.Day;

CurrentDolgi^.Element.User := DataDolgi.User;

CurrentDolgi^.Next := nil;

end;

except

Rewrite(FileWithDolgi);

end;

CloseFile(FileWithDolgi);

AssignFile(FileWithZaemi,'Zaemi.txt');

try

Reset(FileWithZaemi);

New(FirstZaemi);

CurrentZaemi := FirstZaemi;

CurrentZaemi^.Next := nil;

while not EOF(FileWithZaemi) do

begin

Read(FileWithZaemi, DataZaemi);

New(CurrentZaemi^.Next);

CurrentZaemi := CurrentZaemi^.Next;

CurrentZaemi^.Element.Name := DataZaemi.Name;

CurrentZaemi^.Element.Money := DataZaemi.MOney;

CurrentZaemi^.Element.Valuta := DataZaemi.Valuta;

CurrentZaemi^.Element.Day := DataZaemi.Day;

CurrentZaemi^.Element.User := DataZaemi.User;

CurrentZaemi^.Next := nil;

end;

except

Rewrite(FileWithZaemi);

end;

CloseFile(FileWithZaemi);

AssignFile(FileWithUsers, 'MyUsers.txt');

try

Reset(FileWithUsers);

New(FirstUser);

CurrentUser := FirstUser;

CurrentUser^.Next := nil;

while not EOF(FileWithUsers) do

begin

Read(FileWithUsers, DataUser);

New(CurrentUser^.Next);

CurrentUser := CurrentUser^.Next;

CurrentUser^.Element.Name := DataUser.Name;

CurrentUser^.Element.Password := DataUser.Password;

CurrentUser^.Element.KolvoCategoryDohodov := DataUser.KolvoCategoryDohodov;

CurrentUser^.Element.KolvoCategoryRashodov := DataUser.KolvoCategoryRashodov;

CurrentUser^.Element.KolvoDolgnikov := DataUser.KolvoDolgnikov;

CurrentUser^.Element.KolvoZaemchikov := DataUser.KolvoZaemchikov;

CurrentUser^.Element.KolvoDohodov := DataUser.KolvoDohodov;

CurrentUser^.Element.KolvoRashodov := DataUser.KolvoRashodov;

CurrentUser^.Element.KolvoValut := DataUser.KolvoValut;

CurrentUser^.Element.MoneyYouHave := DataUser.MoneyYouHave;

CurrentUser^.Element.AreYouNewOne := DataUser.AreYouNewOne;

CurrentUser^.Next := nil;

cbVibor.Items.Add(DataUser.Name);

end;

except

Rewrite(FileWithUsers);

end;

CloseFile(FileWithUsers);

end;

procedure TPassword.btnNoooClick(Sender: TObject);

begin

PlaySound('D:\ó÷åáà\ÎÀÈÏ\ëàáû\4\_ïðîöåäóðíûé òèï\1.wav', 0, SND\_ASYNC);

Rewrite(FileWithUsers);

CloseFile(FileWithUsers);

Rewrite(FileWithCategoryDohodov);

CloseFile(FileWithCategoryDohodov);

Rewrite(FileWithCategoryRachodov);

CloseFile(FileWithCategoryRachodov);

Rewrite(FileWithValuta);

CloseFile(FileWithValuta);

Rewrite(FileWithRashod);

CloseFile(FileWithRashod);

Rewrite(FileWithDohod);

CloseFile(FileWithDohod);

Rewrite(FileWithDolgi);

CloseFile(FileWithDolgi);

Rewrite(FileWithZaemi);

CloseFile(FileWithZaemi);

cbVibor.Text := '';

cbVibor.Items.Clear;

AssignFile(FileWithCategoryDohodov, 'CategoryDohodov.txt');

try

Reset(FileWithCategoryDohodov);

New(FirstCategoryDohodov);

CurrentCategoryDohodov := FirstCategoryDohodov;

CurrentCategoryDohodov^.Next := nil;

while not EOF(FileWithCategoryDohodov) do

begin

Read(FileWithCategoryDohodov, DataCategoryDohodov);

New(CurrentCategoryDohodov^.Next);

CurrentCategoryDohodov := CurrentCategoryDohodov^.Next;

CurrentCategoryDohodov^.Element.Name := DataCategoryDohodov.Name;

CurrentCategoryDohodov^.Element.User := DataCategoryDohodov.User;

CurrentCategoryDohodov^.Next := nil;

end;

except

Rewrite(FileWithCategoryDohodov);

end;

CloseFile(FileWithCategoryDohodov);

AssignFile(FileWithCategoryRachodov, 'CategoryRashodov.txt');

try

Reset(FileWithCategoryRachodov);

New(FirstCategoryRachodov);

CurrentCategoryRachodov := FirstCategoryRachodov;

CurrentCategoryRachodov^.Next := nil;

while not EOF(FileWithCategoryRachodov) do

begin

Read(FileWithCategoryRachodov, DataCategoryRachodov);

New(CurrentCategoryRachodov^.Next);

CurrentCategoryRachodov := CurrentCategoryRachodov^.Next;

CurrentCategoryRachodov^.Element.Name := DataCategoryRachodov.Name;

CurrentCategoryRachodov^.Element.User := DataCategoryRachodov.User;

CurrentCategoryRachodov^.Next := nil;

end;

except

Rewrite(FileWithCategoryRachodov);

end;

CloseFile(FileWithCategoryRachodov);

AssignFile(FileWithValuta,'Valuta.txt');

try

Reset(FileWithValuta);

New(FirstValuta);

CurrentValuta := FirstValuta;

CurrentValuta^.Next := nil;

while not EOF(FileWithValuta) do

begin

Read(FileWithValuta, DataValuta);

New(CurrentValuta^.Next);

CurrentValuta := CurrentValuta^.Next;

CurrentValuta^.Element.Name := DataValuta.Name;

CurrentValuta^.Element.User := DataValuta.User;

CurrentValuta^.Next := nil;

end;

except

Rewrite(FileWithValuta);

end;

CloseFile(FileWithValuta);

AssignFile(FileWithRashod,'Rashod.txt');

try

Reset(FileWithRashod);

New(FirstRashod);

CurrentRashod := FirstRashod;

CurrentRashod^.Next := nil;

while not EOF(FileWithRashod) do

begin

Read(FileWithRashod, DataRashod);

New(CurrentRashod^.Next);

CurrentRashod := CurrentRashod^.Next;

CurrentRashod^.Element.Money := DataRashod.Money;

CurrentRashod^.Element.Valuta := DataRashod.Valuta;

CurrentRashod^.Element.Category := DataRashod.Category;

CurrentRashod^.Element.Day := DataRashod.Day;

CurrentRashod^.Element.User := DataRashod.User;

CurrentRashod^.Next := nil;

end;

except

Rewrite(FileWithRashod);

end;

CloseFile(FileWithRashod);

AssignFile(FileWithDohod,'Dohod.txt');

try

Reset(FileWithDohod);

New(FirstDohod);

CurrentDohod := FirstDohod;

CurrentDohod^.Next := nil;

while not EOF(FileWithDohod) do

begin

Read(FileWithDohod, DataDohod);

New(CurrentDohod^.Next);

CurrentDohod := CurrentDohod^.Next;

CurrentDohod^.Element.Money := DataDohod.money;

CurrentDohod^.Element.Valuta := DataDohod.Valuta;

CurrentDohod^.Element.Category := DataDohod.Category;

CurrentDohod^.Element.Day := DataDohod.Day;

CurrentDohod^.Element.User := DataDohod.User;

CurrentDohod^.Next := nil;

end;

except

Rewrite(FileWithDohod);

end;

CloseFile(FileWithDohod);

AssignFile(FileWithDolgi,'Dolgi.txt');

try

Reset(FileWithDolgi);

New(FirstDolgi);

CurrentDolgi := FirstDolgi;

CurrentDolgi^.Next := nil;

while not EOF(FileWithDolgi) do

begin

Read(FileWithDolgi, DataDolgi);

New(CurrentDolgi^.Next);

CurrentDolgi := CurrentDolgi^.Next;

CurrentDolgi^.Element.Name := DataDolgi.Name;

CurrentDolgi^.Element.Money := DataDolgi.Money;

CurrentDolgi^.Element.Valuta := DataDolgi.Valuta;

CurrentDolgi^.Element.Day := DataDolgi.Day;

CurrentDolgi^.Element.User := DataDolgi.User;

CurrentDolgi^.Next := nil;

end;

except

Rewrite(FileWithDolgi);

end;

CloseFile(FileWithDolgi);

AssignFile(FileWithZaemi,'Zaemi.txt');

try

Reset(FileWithZaemi);

New(FirstZaemi);

CurrentZaemi := FirstZaemi;

CurrentZaemi^.Next := nil;

while not EOF(FileWithZaemi) do

begin

Read(FileWithZaemi, DataZaemi);

New(CurrentZaemi^.Next);

CurrentZaemi := CurrentZaemi^.Next;

CurrentZaemi^.Element.Name := DataZaemi.Name;

CurrentZaemi^.Element.Money := DataZaemi.MOney;

CurrentZaemi^.Element.Valuta := DataZaemi.Valuta;

CurrentZaemi^.Element.Day := DataZaemi.Day;

CurrentZaemi^.Element.User := DataZaemi.User;

CurrentZaemi^.Next := nil;

end;

except

Rewrite(FileWithZaemi);

end;

CloseFile(FileWithZaemi);

AssignFile(FileWithUsers, 'MyUsers.txt');

try

Reset(FileWithUsers);

New(FirstUser);

CurrentUser := FirstUser;

CurrentUser^.Next := nil;

while not EOF(FileWithUsers) do

begin

Read(FileWithUsers, DataUser);

New(CurrentUser^.Next);

CurrentUser := CurrentUser^.Next;

CurrentUser^.Element.Name := DataUser.Name;

CurrentUser^.Element.Password := DataUser.Password;

CurrentUser^.Element.KolvoCategoryDohodov := DataUser.KolvoCategoryDohodov;

CurrentUser^.Element.KolvoCategoryRashodov := DataUser.KolvoCategoryRashodov;

CurrentUser^.Element.KolvoDolgnikov := DataUser.KolvoDolgnikov;

CurrentUser^.Element.KolvoZaemchikov := DataUser.KolvoZaemchikov;

CurrentUser^.Element.KolvoDohodov := DataUser.KolvoDohodov;

CurrentUser^.Element.KolvoRashodov := DataUser.KolvoRashodov;

CurrentUser^.Element.KolvoValut := DataUser.KolvoValut;

CurrentUser^.Element.MoneyYouHave := DataUser.MoneyYouHave;

CurrentUser^.Element.AreYouNewOne := DataUser.AreYouNewOne;

CurrentUser^.Next := nil;

cbVibor.Items.Add(DataUser.Name);

end;

except

Rewrite(FileWithUsers);

end;

CloseFile(FileWithUsers);

end;

end.