МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ЧУО «Минский КОЛЛЕДЖ предпринимательства»

**ЗАЩИТА ДАННЫХ ОТ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА**

Пояснительная записка к курсовому проекту по учебной дисциплине

«Основы алгоритмизации и программирование»

КПП22.018.104.081ПЗ

**Автор проекта**

Учащийся 2 курса группы П-22 А. А. Юзефович

**Руководитель**

Преподаватель О. А. Салтыкова

Минск, 2018МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ЧУО «МИНСКИЙ КОЛЛЕДЖ ПРЕДПРИНИМАТЕЛЬСТВА»

Специальность

«Программное обеспечение информационных технологий» 2-40 01 01

УТВЕРЖДАЮ

Председатель цикловой комиссии

\_\_\_\_\_\_\_\_\_\_/ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

**ЗАДАНИЕ**

**на курсовой проект**

Учащемуся Юзефовичу Артуру Артёмовичу \_

курса 2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ группы П22 \_

по дисциплине «Основы алгоритмизации и программирование» \_

Тема курсового проекта Защита данных от несанкционированного доступа \_

Исходные данные файл \_

**Состав проекта**

**Пояснительная записка**

Содержание разделов Срок выполнения

Введение\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 Постановка задачи\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2 Вычислительная система\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3 Проектирование\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4 Описание программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5 Отладка и испытание программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6 Описание применения программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Заключение\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Графическая часть проекта**

Схема работы программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи «01» апреля\_\_\_\_\_\_2018г.

Срок завершения «06» июня\_\_\_2018г.

**Преподаватель-руководитель курсового проекта** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись учащегося**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **СОДЕРЖАНИЕ**  [Введение 4.](#_Toc422206439)  [1 Постановки задачи 7.](#_Toc422206440)  [1.1 Формулировка задания 6.](#_Toc422206441)  [1.2 Определение данных и их представление 6.](#_Toc422206442)  [2 Вычислительная система 7.](#_Toc422206443)  [2.1 Обоснование выбора языка программирования 7.](#_Toc422206444)  [2.2 Обоснование выбора среды разработки](#_Toc422206445) 7[. Требования к конфигурации программного и аппаратного обеспечения Ошибка! Закладка не определена.](#_Toc422206446)  [2.3 Требования к конфигурации программного и аппаратного обеспечения](#_Toc422206445) [8. Требования к конфигурации программного и аппаратного обеспечения Ошибка! Закладка не определена.](#_Toc422206446)  [3 Проектирование 9.](#_Toc422206447)  [3.1 Проектирование интерфейса 9.](#_Toc422206448)  [3.2 Физическая модель базы данных 11.](#_Toc422206449)  [4 Описание программы 12.](#_Toc422206451)  [4.1 Логическая структура 12.](#_Toc422206452)  [4.2 Физическая структура 18.](#_Toc422206453)  [4.3 Особенности интерфейса 20.](#_Toc422206454)  [5 Отладка и испытание программы 24.](#_Toc422206455)  [5.1 Тестовые примеры 24.](#_Toc422206456)  [5.2 Анализ полученных результатов 27.](#_Toc422206457)  [6 Описание применения программы 29.](#_Toc422206458)  [6.1 Назначение программы и условия применения 29.](#_Toc422206459)  [Заключение 30.](#_Toc422206460)  [Список использованных источников 32.](#_Toc422206461)  [Приложение 1 33.](#_Toc422206462) | | | | | | | | | | |  |
|  |  |  |  |  | *КПП22.018.104.081ПЗ* | | | | | | |
|  |  |  |  |  |
| *Изм* | *Лист* | *№ докум.* | *Подпись* | *Дата* |
| *Разраб.* | | *Юзефович* |  |  | *Защита данных от НСД* | *Лит.* | | | *Лист* | *Листов* | |
| *Пров.* | | *Салтыкова* |  |  |  | *у* |  | 3 | 68 | |
| *Реценз.* | |  |  |  | *ЧУО МКП* | | | | | |
| *Н.Контр.* | |  |  |  |
| *Утв.* | |  |  |  |

ВВЕДЕНИЕ

Появившиеся в начале 80-ых персональные ЭВМ (ПЭВМ или ПК) прочно вошли во все сферы человеческой деятельности. Вместе с ними у эксплуатирующих ПЭВМ организаций и ведомств возникли и многочисленные проблемы. Одна из них — защита информации. Согласно статистическим данным более 80% компаний и агентств несут финансовые убытки из-за нарушения безопасности данных. Проблема защиты информации представляет собой совокупность тесно связанных проблем в областях права, организации управления, разработки технических средств, программирования и математики. Одна из центральных задач проектирования систем защиты состоит в рациональном распределении имеющихся ресурсов.

Характерная особенность использования ПЭВМ в нашей стране заключатся в том, что доступ к ним имеют многие пользователи. В связи с таким "многопользовательским" режимом работы возникает целый набор взаимосвязанных вопросов по защите информации, хранящейся в ПЭВМ. При создании и использовании ПЭВМ возникает целый ряд взаимосвязанных

теоретических и практических проблем. В коммерческих и военных областях одной из основных является проблема защиты информации. Так можно выделить следующие объективные причины, определяющие важность проблемы защиты информации:

· высокие темпы роста парка ПЭВМ, находящихся в эксплуатации;

· широкое применение ПЭВМ в самых различных сферах человеческой деятельности;

· высокая степень концентрации информации в ПЭВМ;

· совершенствование способов доступа пользователей к ресурсам ПЭВМ;

· усложнение вычислительного процесса в ПЭВМ.

Усложнение методов и средств организации машинной обработки информации приводят к тому, что информация становится все более уязвимой. Этому способствуют такие факторы, как постоянно возрастающие объемы обрабатываемых данных, накопление и хранение данных в ограниченных местах, постоянное расширение круга пользователей, имеющих доступ как к ресурсам ПЭВМ, так и к программам и данным, хранящихся в них, усложнение режимов эксплуатации вычислительных систем и т. п.

Основной целью выполнения курсового проекта является расширение, углубление знаний по дисциплине «Основы алгоритмизации и программирование», формирование навыков научно-исследовательской деятельности.

Задачи выполнения курсового проекта:

* систематизация знаний в области программирования на языке Pascal.
* углубление уровня и расширение объема профессионально значимых знаний, умений и навыков разработки программ в среде Delphi;
* формирование умений и навыков самостоятельной организации научно-исследовательской работы;

- овладение современными методами поиска, обработки и использования информации.

Данная пояснительная записка содержит следующие разделы:

1. Раздел «Постановка задачи» содержит описание всех функций проекта, входных и выходных данных.
2. В разделе «Вычислительная система» приводится описание операционной системы, языка программирования и среды разработки выбранных для реализации проекта.
3. Раздел «Проектирование» содержит описание внешнего пользовательского интерфейса, описание алгоритма программы и структуры справочной системы.
4. В разделе «Описание программы» описываются все библиотечные модули, входящие в состав программы, а также назначение всех функций, входящих в приложение, здесь же приводится интерфейс программы с рисунками пользовательских форм.
5. Раздел «Отладка и испытание программы» иллюстрирует проверку работы программы на различных тестах и содержит анализ полученных результатов.
6. В разделе «Описание применения программы» содержатся сведения о назначении программного средства, классе решаемых задач и ограничениях, накладываемых на область применения.
7. В «Заключении» описываются использованные методы и средства, полнота раскрытия проблемы и возможность дальнейшей модификации проекта.
8. ПОСТАНОВКА ЗАДАЧИ
   1. Формулировка задания

Необходимо разработать аппаратно-программный комплекс «Защита данных от НСД». Реализовать следующие операции:

1. Аутентификация пользователя по ключевому USB-носителю
2. Шифрование выбранных файлов по паролю или по USB-носителю
3. Пример работы шифрования

Предусмотреть возможность выбора USB-носителя для шифрования файлов пользователем.

Реализовать контроль за вводом пароля, чтоб пароль не пыл пустым, при вводе пароля пустым будет выдаваться соответствующее сообщение.

Разработать справочную систему для приложения.

* 1. Определение данных и их представление

Входные данные:

Т. к. программа работает с файлами (шифрование) то входными данными являются файлы различного типа для шифрования.

Выходные данные:

Зашифрованные файлы и защищенные программы.Вычислительная система

**2**

* 1. Обоснование выбора языка программирования

Pascal – один из наиболее известных языков программирования, который очень эффективен для обучения структурному программированию, благодаря небольшому количеству синтаксических конструкций.

Особенностями языка является строгая типизация и наличие средств структурного программирования. Pascal способствует дисциплинированному программированию, так как на ряду со строгой типизацией там сведены к минимуму возможные синтаксические неоднозначности, при этом синтаксис языка интуитивно понятен.

Диалект Object Pascal разработанный в 1985 году поддерживает объектно-ориентированное программирование.

* 1. Обоснование выбора среды разработки

Delphi – интегрированная среда разработки программного обеспечения для операционной системы Windows.

Высоко производительный инструмент визуального построения приложений включает в себя компилятор кода и предоставляет средства визуального программирования. В Delphi так же входят локальный SQL-сервер, генераторы отчетов, библиотеки компонентов.

Преимущества Delphi:

* Быстрая разработка приложений
* Высокая производительность разработанных приложений
* Низкие требование разработанного приложения к ресурсам компьютера
* Наращиваемость за счет встраивания новых компонент в среду Delphi
* Возможность разработки новых компонентов собственными средствами Delphi
* Удачная проработка иерархии объектов
  1. Требования к конфигурации программного и аппаратного обеспечения

Windows 7 – пользовательская операционная система семейства Windows NT. В этой операционной системе реализована поддержка Unicode 5.1, она обладает поддержкой мультитач-управления.

Дополнительным преимуществом Windows 7 является более тесная интеграция с производителями драйверов, 50 новых шрифтов, улучшенная совместимость со старыми приложениями.

Сетевая технология Branch Cache позволяет кэшировать содержимое интернет трафика.

Минимальные требования:

- процессор Intel Pentium III 420 Мгц

- ОЗУ 256 Мбайт

- Место на HDD 1024 Мбайт

1. Проектирование
   1. Проектирование интерфейса

Главная форма приложения содержит 4 компонента типа TButton:

* Шифрование файлов
* Пример работы
* О программе
* Выход

При нажатии на кнопку «Шифрование файлов», открывается новая форма «Шифрование файлов», на которой расположены 1 компонент TMemo для вывода информации о шифровании пользователю. 4 копонента TButton для выбора метода шифрования или расшифровки файлов. 2 компонента TComboBox для выбора USB-носителя для шифрования. 1 компонент TImage для выхода в главное меню.

При нажатии на кнопку «Пример работы» открывается новая форма «Пример работы», на которой расположены 2 компонента TMemo для ввода текста для шифрования и вывода шифрованного текста. Ещё компонент TButton для шифрования введённого текста. 1 компонент TImage для выхода в главное меню.

При нажатии на кнопку «О программе» открывается справочная система программы. Где пользователь может посмотреть как пользоваться программой, различные правила шифрования или расшифровки файлов.

Так же есть форма, которую мы будем видеть самой первой, но она не является главной. На этой форме содержится 2 компонента TButton для входа в приложение или же создания ключевого USB-носителя. 1 компонент TComboBox для выбора USB-носителя­.

* 1. Описание алгоритма

Перед описанием алгоритма следует ввести некоторые термины.

Зашифрованием данных называется процесс преобразования открытых данных в зашифрованные с помощью шифра, а расшифрованием данных — процесс преобразования закрытых данных в открытые с помощью шифра.

Шифрованием называется процесс зашифрования или расшифрования данных.

Дешифрованием будем называть процесс преобразования закрытых данных в открытые при неизвестном ключе и, возможно, неизвестном алгоритме.

Криптографическая защита — это защита данных с помощью криптографического преобразования, под которым понимается преобразование данных шифрованием.

Уравнение зашифрования — соотношение, описывающее процесс образования зашифрованных данных из открытых данных в результате преобразований, заданных алгоритмом криптографического преобразования.

Уравнение расшифрования — соотношение, описывающее процесс образования открытых данных из зашифрованных данных в результате преобразований, заданных алгоритмом криптографического преобразования.

Под шифром понимается совокупность обратимых преобразований множества открытых данных на множество зашифрованных данных, заданных алгоритмом криптографического преобразования.

Криптостойкостью называется характеристика шифра, определяющая его стойкость к дешифрованию. Обычно эта характеристика определяется периодом времени, необходимым для дешифрования.

Принцип зашифрования заключается в генерации гаммы шифра с помощью датчика псевдослучайных чисел (ПСЧ) и наложением полученной гаммы на открытые данные с помощью логической операции “исключающее ИЛИ” (т. е. обратимым образом). Процесс расшифрования данных сводится к повторной генерации гаммы шифра при известном ключе и наложению такой гаммы на зашифрованные данные. Полученный зашифрованный текст является достаточно трудным для раскрытия в том случае, когда гамма шифра не содержит повторяющихся битовых последовательностей. По сути дела гамма шифра должна изменяться случайным образом для каждого шифруемого слова. Фактически если период гаммы превышает длину всего зашифрованного текста и неизвестна никакая часть исходного текста, то шифр можно раскрыть только прямым перебором (подбором ключа). В этом случае криптостойкость определяется размером ключа. Чтобы получить линейные последовательности элементов гаммы, используются датчики ПСЧ. К настоящему времени на основе теории групп разработано несколько типов таких датчиков.

В своей программе я использовал так называемый конгруэнтный генератор ПСЧ — наиболее доступный и эффективный. Для этого класса генераторов ПСЧ можно сделать математически строгое заключение о том, какими свойствами обладают выходные сигналы этих генераторов с точки зрения периодичности и случайности. Данный линейный конгруэнтный датчик ПСЧ вырабатывает последовательности псевдослучайных чисел T(i), описываемые соотношением:

, (1)

T(0) — исходная величина, выбранная в качестве порождающего числа. Этот датчик ПСЧ генерирует псевдослучайные числа с определенным периодом повторения, зависящим от выбранных значений A и C. Значение М обычно устанавливается равным 2b, где b — длина слова ЭВМ в битах. Датчик имеет максимальный период М до того, как генерируемая последовательность чисел начнет повторяться. Линейный конгруэнтный датчик ПСЧ имеет максимальную длину М тогда и только тогда, когда A mod 4 = 1 и С — нечетное. В своей программе я положил А = 5, С = 27, Т(0) — пароль, вводимый пользователем.

С полученной последовательностью Т(i) поступают следующим образом:

, (2)

где D(i) — последовательность открытых данных, F(i) — последовательность зашифрованных данных.

Также при разработке алгоритма шифрования использовался алгоритм американского федерального стандарта на шифрование данных — Data Encryption Standard (DES).

При зашифровании входные данные шифруются по формуле (2), далее обрабатываются блоками по 64 слова (word). Эта обработка заключается в следующем: 4 слова переставляются в соответствии с таблицей, изображенной на рис. 1:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **40** | **8** | **48** | **16** | **56** | **24** | **64** | **32** | |  |
| **39** | **7** | **47** | **15** | **55** | **23** | **63** | **31** | |  |
| **38** | **6** | **46** | **14** | **54** | **22** | **62** | **30** | |  |
| **37** | **5** | **45** | **13** | **53** | **21** | **61** | **29** | |  |
| **36** | **4** | **44** | **12** | **52** | **20** | **60** | **28** | |  |
| **35** | **3** | **43** | **11** | **51** | **19** | **59** | **27** | |  |
| **34** | **2** | **42** | **10** | **50** | **18** | **58** | **26** | |  |
| **33** | **1** | **41** | **9** | **49** | **17** | **57** | | **25** | |
|  |  |  |  |  |  |  | |  | |

*Рис. 1. Перестановка после зашифрования.*

Как видно из данной таблицы, слово 40 входной последовательности становится 1-ым, слово 8 — 2-ым и т. д. Процесс расшифрования данных является инверсным относительно процесса зашифрования. Т. е. данные сначала переставляются в соответствии с таблицей, изображенной на рис 2., а затем преобразуются по формуле (2). Как легко видеть, данная перестановка является обратной по отношению к начальной.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **58** | **50** | **42** | **34** | **26** | **18** | **10** | **2** | |  |
| **60** | **52** | **44** | **36** | **28** | **20** | **12** | **4** | |  |
| **62** | **54** | **46** | **38** | **30** | **22** | **14** | **6** | |  |
| **64** | **56** | **48** | **40** | **32** | **24** | **16** | **8** | |  |
| **57** | **49** | **41** | **33** | **25** | **17** | **9** | **1** | |  |
| **59** | **51** | **43** | **35** | **27** | **19** | **11** | **3** | |  |
| **61** | **53** | **45** | **37** | **29** | **21** | **13** | **5** | |  |
| **63** | **55** | **47** | **39** | **31** | **23** | **15** | | **7** | |
|  |  |  |  |  |  |  | |  | |
|  |  |  |  |  |  |  |  |  |  |

*Рис. 2. Перестановка перед расшифрованием.*

**Описание программы**

* 1. Логическая структура

Основная форма (Form1):

- procedure Button1Click(Sender: TObject); - открытие формы «Шифрование файлов».

- procedure Button2Click(Sender: TObject); - открытие формы «Пример работы».

- procedure Button3Click(Sender: TObject); - открывает справку.

- procedure FormClose(Sender: TObject; var Action: TCloseAction); - закрывает форму и выходит из программы.

Форма «Шифрование файлов»(Form2):

- procedure Button1Click(Sender: TObject); - шифрование выбранного файла по паролю.

- procedure Button2Click(Sender: TObject); - расшифрование выбранного файла по паролю.

- procedure FormCreate(Sender: TObject); - записывает определённые инструкции в TMemo1.

- procedure Image1Click(Sender: TObject); - переход на главную форму.

- procedure FormClose(Sender: TObject; var Action: TCloseAction); - закрывает форму и выходит из программы.

- procedure Button3Click(Sender: TObject); - выбор USB-носителя для шифрования.

- procedure Button4Click(Sender: TObject); - выбор USB-носителя для расшифрования.

- procedure ComboBox1Change(Sender: TObject); - шифрование файла по выбранному USB-носителю.

- procedure ComboBox2Change(Sender: TObject); - расшифрование файла по выбранному USB-носителю.

- procedure Image1MouseEnter(Sender: TObject); - изменение курсора при наведении на картинку.

- procedure Image2Click(Sender: TObject); - переход назад к выбору метода шифрования.

- procedure Image2MouseEnter(Sender: TObject); - изменение курсора при наведении на картинку.

- procedure pass; - получение пароля.

Форма «Пример работы» (Form3):

- procedure Button1Click(Sender: TObject); -шифрование введённого текста.

- procedure FormCreate(Sender: TObject); -очистка полей TMemo1 и TMemo2.

- procedure FormClose(Sender: TObject; var Action: TCloseAction); - закрывает форму и выходит из программы.

- procedure Image1Click(Sender: TObject); - переход на главную форму.

- procedure Memo1Change(Sender: TObject); - активирует кнопку если в TMemo1 был введён какой-то текст.

- procedure Image1MouseEnter(Sender: TObject); - изменение курсора при наведении на картинку.

Форма «Вход в приложение» (Form4):

- procedure Button1Click(Sender: TObject); - считывание всех подключённых USB-носителей к ПК.

- procedure ComboBox1Change(Sender: TObject); - создание ключевого USB-носителя.

- procedure FormCreate(Sender: TObject); - проверка на то, был ли создан ключевой USB-носитель ранее.

- procedure Button2Click(Sender: TObject); - вход в приложение по ключевому USB-носителю.

* 1. Физическая структура

Unit1 – основная форма программы, служит для выбора выполняемой операции.

Unit2 – форма «Сложение и вычитание матриц», служит для сложения или вычитания 2-х матриц.

Unit3 – форма «Умножение матрицы на число», служит для умножения матрицы на число.

Unit4 – форма «Умножение матриц», служит для умножения 2-х матриц.

Unit5 – форма «Возведение матрицы в степень», служит для возведения матрицы в степень.

Unit6 – форма «Транспонирование матрицы», служит для нахождения транспонированной матрицы.

Unit7 – форма «Детерминант матрицы», служит для нахождения детерминанта матрицы.

* 1. Особенности интерфейса

Интерфейс главной формы приложения показан на рисунке 1

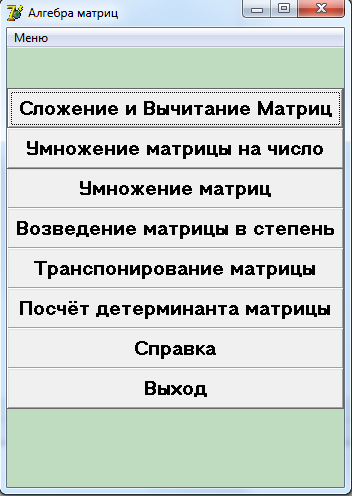


Рисунок 1 – Главная форма приложения

Интерфейс формы «Сложение и вычитание матриц» показан на рисунке 2.

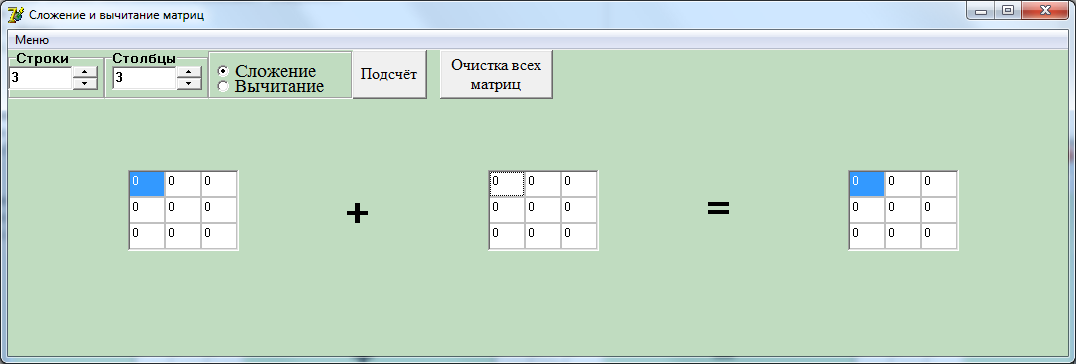


Рисунок 2 – Форма «Сложение и вычитание матриц»

Интерфейс формы «Умножение матрицы на число» показан на рисунке 3.

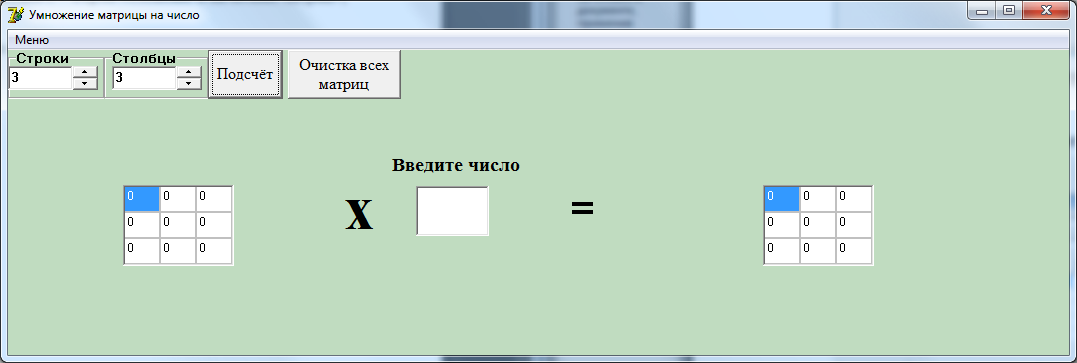


Рисунок 3 – Форма «Умножение матрицы на число»

Интерфейс формы «Умножение матриц» показан на рисунке 4.

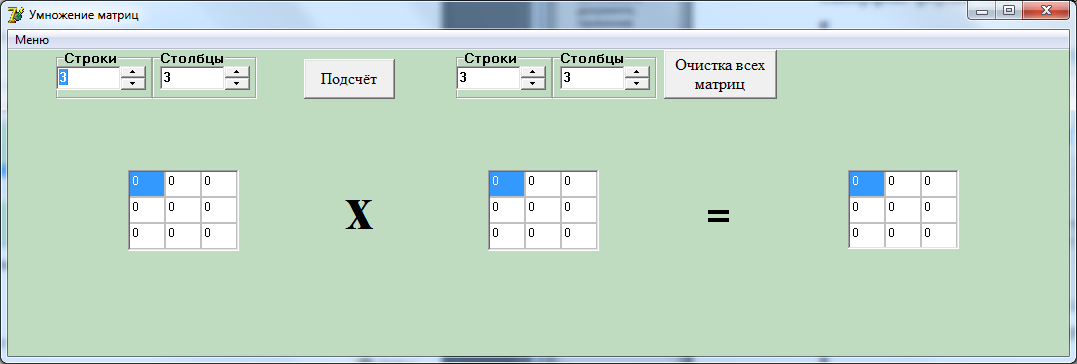


Рисунок 4 – Форма «Умножение матриц»

Интерфейс формы «Возведение матрицы в степень» показан на рисунке 5.

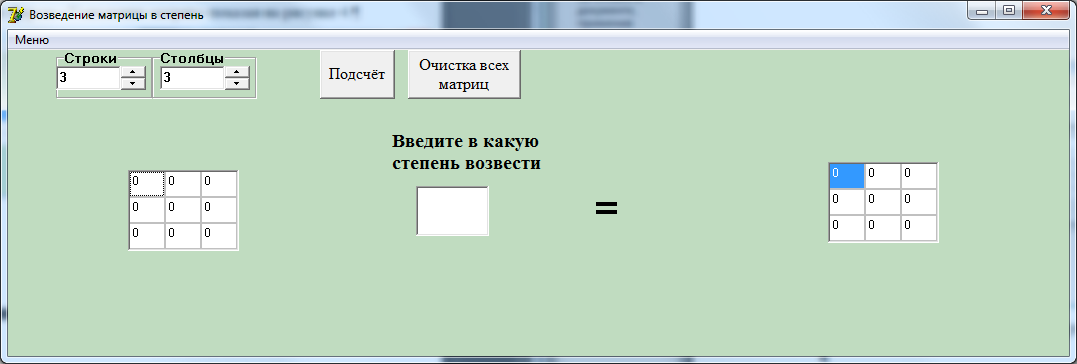


Рисунок 5 – Форма «Возведение матрицы в степень»

Интерфейс формы «Транспонирование матрицы» показан на рисунке 6.

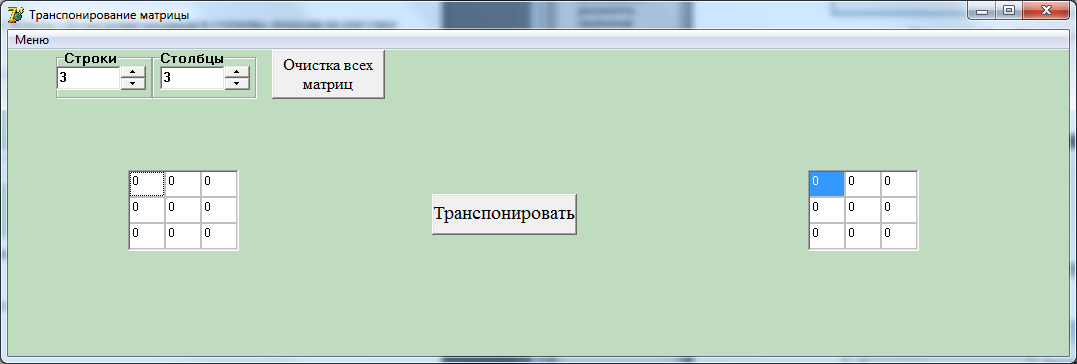


Рисунок 6 – Форма «Транспонирование матрицы»

Интерфейс формы «Детерминант матрицы» показан на рисунке 7.

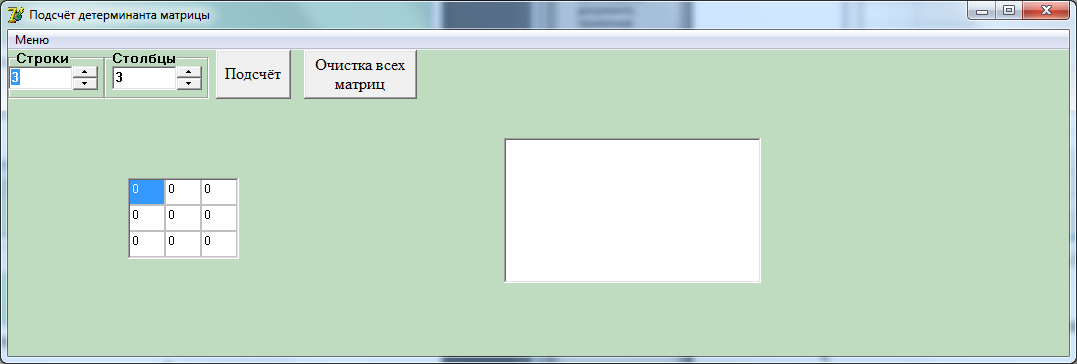


Рисунок 7 – Форма «Детерминант матрицы

1. ОТЛАДКА И ИСПЫТАНИЕ ПРОГРАММЫ
   1. Тестовые примеры

Тест №1: Сложение матриц с одинаковой размерностью и корректно заполненными значениями. Результат приведён на рисунке 8.

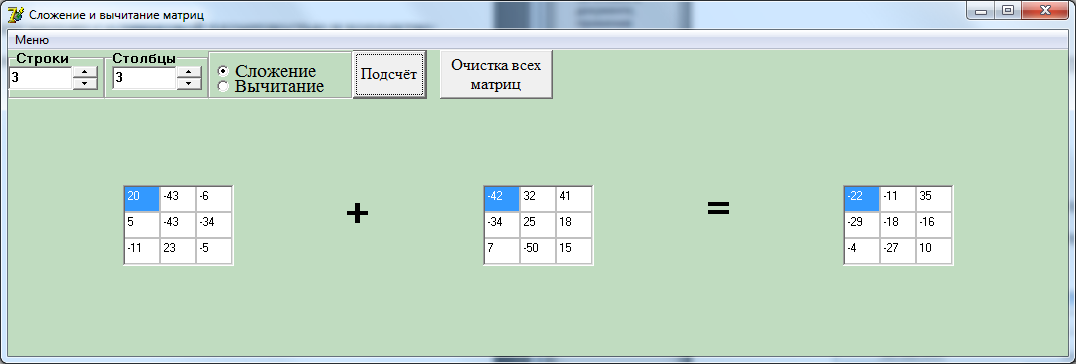


Рисунок 8 – Тест №1

Тест №2: Вычитание матриц с одинаковой размерностью и корректно заполненными значениями. Результат приведён на рисунке 9.

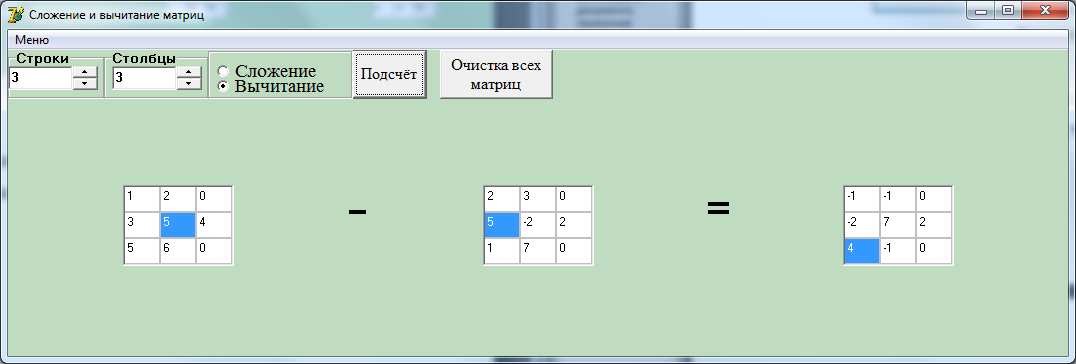


Рисунок 9 – Тест №2

Тест №3: Умножение матриц с одинаковой размерностью и корректно заполненными значениями. Результат приведён на рисунке 10.

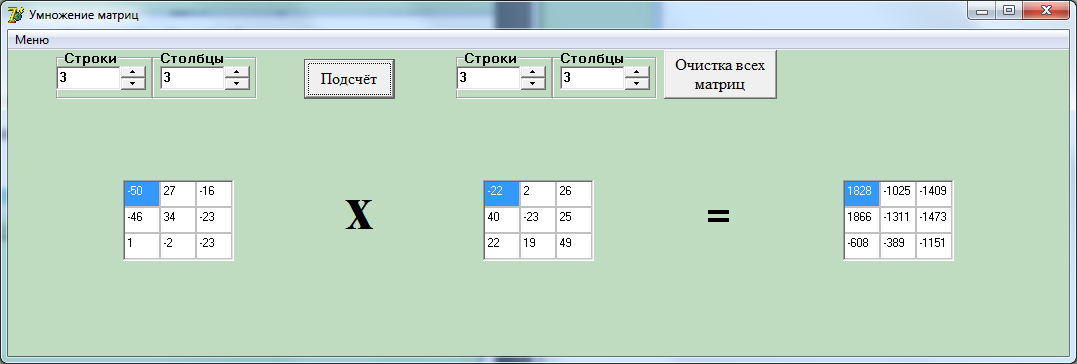


Рисунок 10 – Тест №3

Тест №4: Умножение матриц с разной размерностью и корректно заполненными значениями. Результат приведён на рисунке 11.

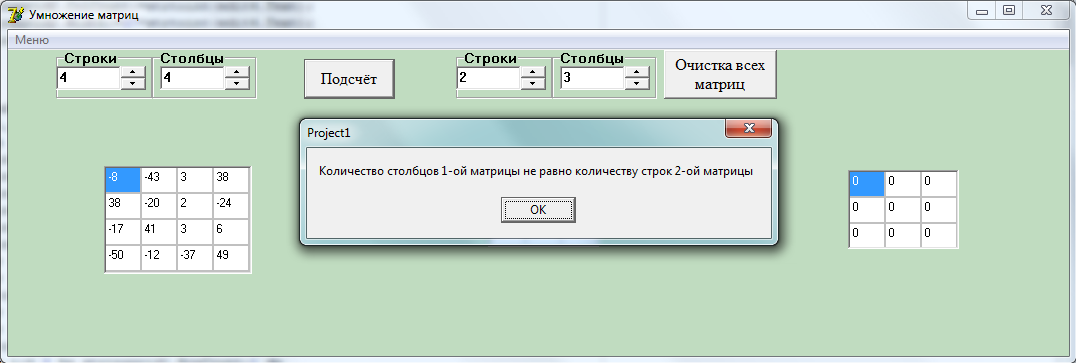


Рисунок 11 – Тест №4

Тест №5: Возведение матрицы в степень с корректно заполненными значениями. Результат приведён на рисунке 12.

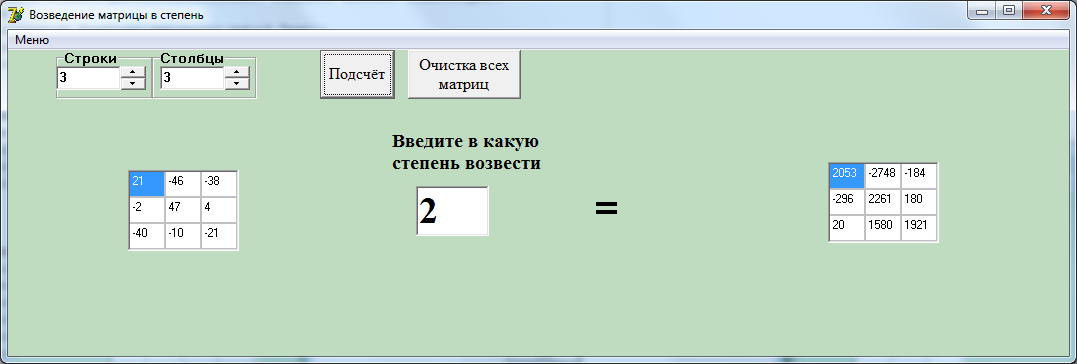


Рисунок 12 – Тест №5

Тест №6: Умножение матрицы на число с некорректно заполненными значениями. Результат приведён на рисунке 12.

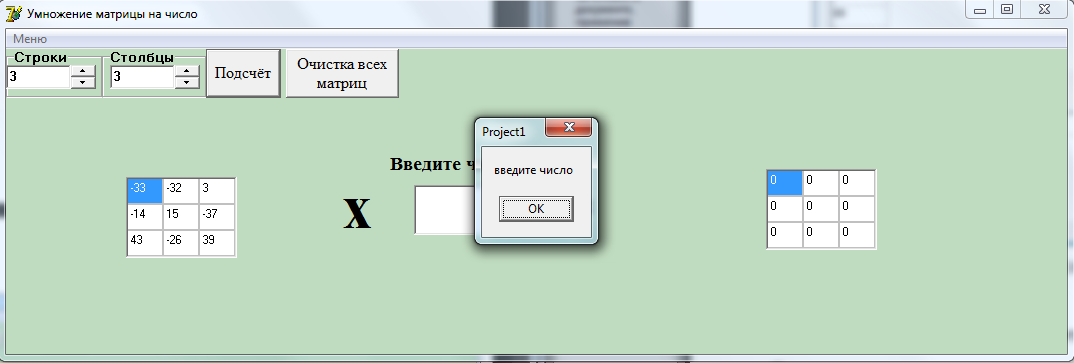


Рисунок 12 – Тест №6

Тест №7: Подсчёт детерминанта матрицы с корректно заполненными значениями. Результат приведён на рисунке 13.

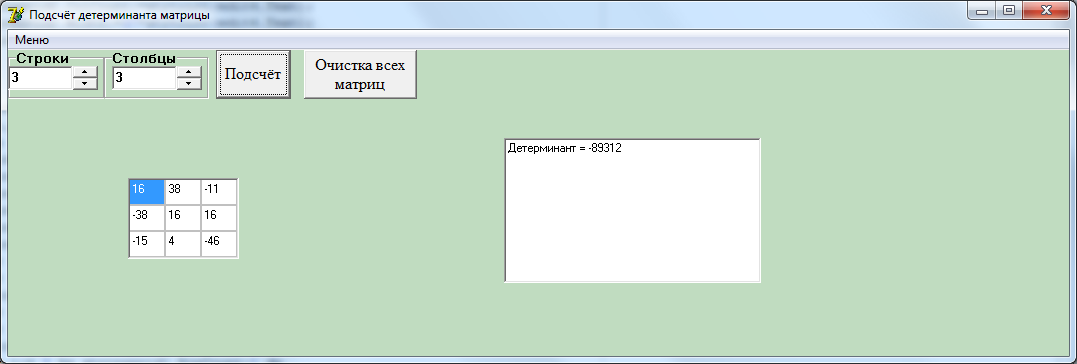


Рисунок 13 – Тест №7

* 1. Анализ полученных результатов

Тест №1 показал, что при корректно введённых данных и заполненных значений, всё считается правильно, без каких либо ошибок.

Тест №2 показал, что при корректно введённых данных и заполненных значений, всё считается правильно, без каких либо ошибок.

Тест №3 показал, что при корректно введённых данных и заполненных значений, всё считается правильно, без каких либо ошибок.

Тест №4 показал, что при неправильном выборе размерности матрицы, что не соответствует правилу ,умножения матрицы, то появляется сообщение, информирующие нас о том, что размерность выбрана не правильно.

Тест №5 показал, что при корректно введённых данных и заполненных значений, всё считается правильно, без каких либо ошибок.

Тест №6 показал, что при некорректно введённых данных, появляется сообщение, информирующие нас о том, что пользователь не заполнил число и программа просит ввести число.

Тест №7 показал, что при корректно введённых данных и заполненных значений, всё считается правильно, без каких либо ошибок.

1. ОПИСАНИЕ ПРИМЕНЕНИЯ ПРОГРАММЫ
   1. Назначение программы и условия применения

Данная программа используется в качестве быстрого подсчёта матрицы любых размерностей. Пользователь может как сам считать всё в данной программе так и сверяться со своим решением.

Минимальные требования для корректной работы приложения:

- процессор Intel Pentium III 420 Мгц

- ОЗУ 256 Мбайт

- Место на HDD 1024 Мбайт

- VGA или монитор с более высоким разрешением;

- мышь, клавиатура.

Для запуска программы нужно зайти в папку Курсовая работа и запустить файл «Алгебра матриц». После запуска откроется окно. Пользователь должен выбрать ему нужную функцию калькулятора и затем осуществлять подсчёт данных.

ЗАКЛЮЧЕНИЕ

В данном курсовом проекте была разработана программа «Алгебра матриц».

Было произведено тестирование разработанного приложения. Программа работает корректно при любых действиях пользователя.

Проект может быть доработан следующим образом:

* Добавить действие нахождения обратной матрицы.
* Добавить действие нахождения обратной матрицы методом алгебраических дополнений.
* Добавить действие проверки матрицы на единичность.
* Добавить справочную систему.

Список использованных источников

1. Архангельский А.Я.Программирование в Delphi для Windows. М.: Бином-Пресс,2007.
2. Галисеев Г.В.Компоненты в Delphi 7. Профессиональная работа. М.: Издательский дом «Вильямс», 2004.
3. Долинский М.С.Алгоритмизация и программирование на TurboPascal: от простых до олимпиадных задач. СПб: Питер, 2004.
4. Хомоненко А.Д. и др. Delphi 7. СПб.: БХВ-Петербург, 2008.
5. Руссинович М. Внутреннее устройство Microsoft Windows, издательство Питер, 2013

Приложение 1

**(обязательное)**

**Листинг программы**

unit Unit1;

interface

uses

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

Dialogs, StdCtrls, Menus;

type

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Button3: TButton;

Button4: TButton;

Button5: TButton;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

Button6: TButton;

Button7: TButton;

Button8: TButton;

procedure Button2Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Button4Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure Button8Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

uses Unit2, Unit3, Unit4, Unit5, Unit6, Unit7;

{$R \*.dfm}

procedure TForm1.Button2Click(Sender: TObject);

begin

Form3.show;

Form1.Hide;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

Form2.show;

Form1.Hide;

end;

procedure TForm1.Button3Click(Sender: TObject);

begin

Form4.show;

Form1.Hide;

end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TForm1.Button4Click(Sender: TObject);

begin

Form5.show;

Form1.Hide;

end;

procedure TForm1.Button5Click(Sender: TObject);

begin

Form6.show;

Form1.Hide;

end;

procedure TForm1.Button6Click(Sender: TObject);

begin

Form7.show;

Form1.Hide;

end;

procedure TForm1.N3Click(Sender: TObject);

begin

application.Terminate;

end;

procedure TForm1.Button8Click(Sender: TObject);

begin

application.Terminate;

end;

end.

unit Unit2;

interface

uses

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

Dialogs, StdCtrls, Grids, Menus, ComCtrls, ExtCtrls;

type

TForm2 = class(TForm)

Label2: TLabel;

Edit1: TEdit;

Edit2: TEdit;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

StringGrid2: TStringGrid;

StringGrid3: TStringGrid;

N4: TMenuItem;

UpDown1: TUpDown;

UpDown2: TUpDown;

GroupBox1: TGroupBox;

Label1: TLabel;

GroupBox2: TGroupBox;

Label3: TLabel;

Button1: TButton;

RadioGroup1: TRadioGroup;

StringGrid1: TStringGrid;

Button2: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure RadioGroup1Click(Sender: TObject);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure StringGrid2KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2; i,j:integer;

implementation

uses Unit1, Unit3;

{$R \*.dfm}

procedure TForm2.Button1Click(Sender: TObject);

begin

if radiogroup1.ItemIndex=0 then

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

stringgrid3.Cells[j,i]:=inttostr(strtoint(stringgrid1.Cells[j,i])+strtoint(stringgrid2.Cells[j,i]));

end;

if radiogroup1.ItemIndex=1 then

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

stringgrid3.Cells[j,i]:=inttostr(strtoint(stringgrid1.Cells[j,i])-strtoint(stringgrid2.Cells[j,i]));

end;

end;

procedure TForm2.Button2Click(Sender: TObject);

begin

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

end;

begin

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

end;

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

end;

end;

procedure TForm2.N3Click(Sender: TObject);

begin

form1.close;

form3.close;

end;

procedure TForm2.N2Click(Sender: TObject);

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

edit1.Text:='3';

edit2.Text:='3';

stringgrid1.Height:=27\*strtoint(edit1.Text); //при закрытии формы позиция матрицы

stringgrid1.Top:=175-stringgrid1.Height div 2; // становилась по центру

stringgrid2.Height:=27\*strtoint(edit1.Text);

stringgrid2.Top:=175-stringgrid1.Height div 2;

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=175-stringgrid1.Height div 2;

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid2.Width:=37\*strtoint(edit2.Text);

stringgrid2.left:=530-stringgrid1.width div 2;

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=890-stringgrid1.width div 2; //

form2.Visible:=false;

form1.Show;

end;

procedure TForm2.N4Click(Sender: TObject);

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

stringgrid1.Cells[j,i]:=inttostr(random(100)-50);

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do

stringgrid2.Cells[j,i]:=inttostr(random(100)-50);

end;

procedure TForm2.FormCreate(Sender: TObject);

begin

stringgrid1.RowCount:=9;

stringgrid1.ColCount:=9;

stringgrid2.RowCount:=9;

stringgrid2.ColCount:=9;

stringgrid3.RowCount:=9;

stringgrid3.ColCount:=9;

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[i,j]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[i,j]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0';

stringgrid1.RowCount:=3;

stringgrid1.ColCount:=3;

stringgrid2.RowCount:=3;

stringgrid2.ColCount:=3;

stringgrid3.RowCount:=3;

stringgrid3.ColCount:=3;

UpDown2.Associate:=Edit2;

UpDown1.Associate:=Edit1;

end;

procedure TForm2.UpDown1Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid2.RowCount:=strtoint(edit1.Text);

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=175-stringgrid1.Height div 2;

stringgrid2.Height:=27\*strtoint(edit1.Text);

stringgrid2.Top:=175-stringgrid1.Height div 2;

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=175-stringgrid1.Height div 2;

end;

procedure TForm2.UpDown2Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid2.ColCount:=strtoint(edit2.Text);

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid2.Width:=37\*strtoint(edit2.Text);

stringgrid2.left:=530-stringgrid1.width div 2;

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=890-stringgrid1.width div 2;

end;

procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TForm2.RadioGroup1Click(Sender: TObject);

begin

case radiogroup1.ItemIndex of

1: begin

label1.Visible:=false;

label3.Visible:=true;

end;

0:

begin

label1.Visible:=true;

label3.Visible:=false;

end;

end;

end;

procedure TForm2.StringGrid1KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm2.StringGrid2KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm2.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm2.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit2.text)=1) then key:=char(0);

end;

end.

unit Unit3;

interface

uses

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

Dialogs, StdCtrls, ComCtrls, Grids, Menus;

type

TForm3 = class(TForm)

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N4: TMenuItem;

N3: TMenuItem;

StringGrid1: TStringGrid;

GroupBox1: TGroupBox;

UpDown1: TUpDown;

Edit1: TEdit;

GroupBox2: TGroupBox;

UpDown2: TUpDown;

Edit2: TEdit;

Button1: TButton;

Edit3: TEdit;

Label3: TLabel;

Label1: TLabel;

Label2: TLabel;

StringGrid3: TStringGrid;

Button2: TButton;

procedure N2Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3; i,j,n:integer;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm3.N2Click(Sender: TObject);

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[i,j]:='0'; // обнуляем все матрицы при закрытии формы

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0';

edit3.Text:='';

Stringgrid1.Cells[i,j]:='';

edit1.Text:='3';

edit2.Text:='3';

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=175-stringgrid1.Height div 2;

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=175-stringgrid1.Height div 2;

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=810-stringgrid1.width div 2;

form3.Visible:=false;

form1.Show;

end;

procedure TForm3.N3Click(Sender: TObject);

begin

form1.close;

form3.close;

end;

procedure TForm3.N4Click(Sender: TObject);

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

stringgrid1.Cells[j,i]:=inttostr(random(100)-50);

end;

procedure TForm3.UpDown1Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=175-stringgrid1.Height div 2;

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=175-stringgrid1.Height div 2;

end;

procedure TForm3.UpDown2Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=810-stringgrid1.width div 2;

end;

procedure TForm3.Button1Click(Sender: TObject);

begin

if edit3.text<>'' then

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

stringgrid3.Cells[j,i]:=inttostr(strtoint(stringgrid1.Cells[j,i])\*strtoint(edit3.Text));

end else showmessage('введите число');

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

stringgrid1.RowCount:=9;

stringgrid1.ColCount:=9;

stringgrid3.RowCount:=9;

stringgrid3.ColCount:=9;

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do // обнуляем все матрицы при открытии формы

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

stringgrid1.RowCount:=3;

stringgrid1.ColCount:=3;

stringgrid3.RowCount:=3;

stringgrid3.ColCount:=3;

UpDown2.Associate:=Edit2;

UpDown1.Associate:=Edit1;

end;

procedure TForm3.FormClose(Sender: TObject; var Action: TCloseAction);

begin

application.Terminate;

end;

procedure TForm3.StringGrid1KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm3.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm3.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit2.text)=1) then key:=char(0);

end;

procedure TForm3.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8 :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm3.Button2Click(Sender: TObject);

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do // обнуляем все матрицы при открытии формы

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

end;

end.

unit Unit4;

interface

uses

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

Dialogs, StdCtrls, Grids, ComCtrls, Menus;

type

TForm4 = class(TForm)

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N4: TMenuItem;

N3: TMenuItem;

GroupBox1: TGroupBox;

UpDown1: TUpDown;

Edit1: TEdit;

GroupBox2: TGroupBox;

UpDown2: TUpDown;

Edit2: TEdit;

Button1: TButton;

StringGrid1: TStringGrid;

StringGrid2: TStringGrid;

StringGrid3: TStringGrid;

Label3: TLabel;

GroupBox3: TGroupBox;

UpDown3: TUpDown;

Edit3: TEdit;

GroupBox4: TGroupBox;

UpDown4: TUpDown;

Edit4: TEdit;

Label1: TLabel;

Button2: TButton;

procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);

procedure FormCreate(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Button1Click(Sender: TObject);

procedure UpDown3Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown4Click(Sender: TObject; Button: TUDBtnType);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure StringGrid2KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure Edit4KeyPress(Sender: TObject; var Key: Char);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm4.UpDown1Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=170-stringgrid1.Height div 2;

end;

procedure TForm4.UpDown2Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

end;

procedure TForm4.UpDown3Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid2.RowCount:=strtoint(edit3.Text);

stringgrid2.Height:=27\*strtoint(edit3.Text);

stringgrid2.Top:=170-stringgrid2.Height div 2;

end;

procedure TForm4.UpDown4Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid2.ColCount:=strtoint(edit4.Text);

stringgrid2.Width:=37\*strtoint(edit4.Text);

stringgrid2.left:=530-stringgrid2.width div 2;

end;

procedure TForm4.FormCreate(Sender: TObject);

var i,j:integer;

begin

stringgrid1.RowCount:=9;

stringgrid1.ColCount:=9;

stringgrid2.RowCount:=9;

stringgrid2.ColCount:=9;

stringgrid3.RowCount:=9;

stringgrid3.ColCount:=9;

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

stringgrid1.RowCount:=3;

stringgrid1.ColCount:=3;

stringgrid2.RowCount:=3;

stringgrid2.ColCount:=3;

stringgrid3.RowCount:=3;

stringgrid3.ColCount:=3;

UpDown2.Associate:=Edit2;

UpDown1.Associate:=Edit1;

UpDown3.Associate:=Edit3;

UpDown4.Associate:=Edit4;

end;

procedure TForm4.N2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[i,j]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[i,j]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0';

edit1.Text:='3';

edit2.Text:='3';

edit3.Text:='3';

edit4.Text:='3';

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=170-stringgrid1.Height div 2;

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2; // //при закрытии формы позиция матрицы

stringgrid2.RowCount:=strtoint(edit3.Text); // становилась по центру

stringgrid2.Height:=27\*strtoint(edit3.Text);

stringgrid2.Top:=170-stringgrid1.Height div 2;

stringgrid2.ColCount:=strtoint(edit4.Text);

stringgrid2.Width:=37\*strtoint(edit4.Text);

stringgrid2.left:=530-stringgrid1.width div 2;

Stringgrid3.RowCount:=Stringgrid1.RowCount;

Stringgrid3.ColCount:= Stringgrid2.ColCount;

Stringgrid3.Height:=27\*strtoint(edit1.Text);

Stringgrid3.Top:=170-stringgrid1.Height div 2;

Stringgrid3.Width:=37\*strtoint(edit4.Text);

Stringgrid3.left:=890-stringgrid2.width div 2;

form4.Visible:=false;

form1.show;

end;

procedure TForm4.N4Click(Sender: TObject);

var i,j:integer;

begin

randomize;

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

stringgrid1.Cells[j,i]:=inttostr(random(100)-50);

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do

stringgrid2.Cells[j,i]:=inttostr(random(100)-50);

end;

procedure TForm4.N3Click(Sender: TObject);

begin

form1.close;

form4.close;

end;

procedure TForm4.FormClose(Sender: TObject; var Action: TCloseAction);

begin

application.Terminate;

end;

procedure TForm4.Button1Click(Sender: TObject);

var i,j,k,Sum:integer;

begin

if (StringGrid1.ColCount=StringGrid2.RowCount) then

begin

Stringgrid3.RowCount:=Stringgrid1.RowCount;

Stringgrid3.ColCount:= Stringgrid2.ColCount;

Stringgrid3.Height:=27\*strtoint(edit1.Text);

Stringgrid3.Top:=170-stringgrid1.Height div 2;

Stringgrid3.Width:=37\*strtoint(edit4.Text);

Stringgrid3.left:=890-stringgrid2.width div 2;

for i:=0 to StringGrid3.RowCount-1 do

for j:=0 to StringGrid3.ColCount-1 do

begin

Sum:=0;

for k:=0 to StringGrid1.ColCount-1 do

Sum:=Sum+StrToInt(StringGrid1.Cells[k,i])\*StrToInt(StringGrid2.Cells[j,k]);

StringGrid3.Cells[j,i]:=IntToStr(Sum);

end;

end else ShowMessage('Количество столбцов 1-ой матрицы не равно количеству строк 2-ой матрицы');

end;

procedure TForm4.StringGrid1KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm4.StringGrid2KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm4.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm4.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit2.text)=1) then key:=char(0);

end;

procedure TForm4.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit3.text)=1) then key:=char(0);

end;

procedure TForm4.Edit4KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm4.Button2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

end;

end.

unit Unit5;

interface

uses

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

Dialogs, Menus, StdCtrls, Grids, ComCtrls;

type

TForm5 = class(TForm)

StringGrid1: TStringGrid;

StringGrid2: TStringGrid;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N4: TMenuItem;

N3: TMenuItem;

GroupBox1: TGroupBox;

UpDown1: TUpDown;

Edit1: TEdit;

GroupBox2: TGroupBox;

UpDown2: TUpDown;

Edit2: TEdit;

Button1: TButton;

Label2: TLabel;

Edit3: TEdit;

Label1: TLabel;

StringGrid3: TStringGrid;

Button2: TButton;

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCreate(Sender: TObject);

procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);

procedure Button1Click(Sender: TObject);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure Edit3KeyPress(Sender: TObject; var Key: Char);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm5.N2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[i,j]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0';

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[i,j]:='0';

edit1.Text:='3';

edit2.Text:='3';

edit3.Text:='';

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=170-stringgrid1.Height div 2;

stringgrid2.RowCount:=strtoint(edit1.Text);

stringgrid2.Height:=27\*strtoint(edit1.Text);

stringgrid2.Top:=170-stringgrid1.Height div 2;

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=418-stringgrid1.Height div 2 ;

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid2.ColCount:=strtoint(edit2.Text);

stringgrid2.Width:=37\*strtoint(edit2.Text);

stringgrid2.left:=850-stringgrid1.width div 2;

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=642-stringgrid1.width div 2;

form5.Visible:=false;

form1.show;

end;

procedure TForm5.N4Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

stringgrid1.Cells[j,i]:=inttostr(random(100)-50);

end;

procedure TForm5.N3Click(Sender: TObject);

begin

form1.close;

form5.close;

end;

procedure TForm5.FormClose(Sender: TObject; var Action: TCloseAction);

begin

application.Terminate;

end;

procedure TForm5.FormCreate(Sender: TObject);

var i,j:integer;

begin

stringgrid1.RowCount:=9;

stringgrid1.ColCount:=9;

stringgrid2.RowCount:=9;

stringgrid2.ColCount:=9;

stringgrid3.RowCount:=9;

stringgrid3.ColCount:=9;

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

stringgrid1.RowCount:=3;

stringgrid1.ColCount:=3;

stringgrid2.RowCount:=3;

stringgrid2.ColCount:=3;

stringgrid3.RowCount:=3;

stringgrid3.ColCount:=3;

UpDown2.Associate:=Edit2;

UpDown1.Associate:=Edit1;

end;

procedure TForm5.UpDown1Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=170-stringgrid1.Height div 2;

stringgrid2.RowCount:=strtoint(edit1.Text);

stringgrid2.Height:=27\*strtoint(edit1.Text);

stringgrid2.Top:=170-stringgrid1.Height div 2;

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=418-stringgrid1.Height div 2;

end;

procedure TForm5.UpDown2Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid2.ColCount:=strtoint(edit2.Text);

stringgrid2.Width:=37\*strtoint(edit2.Text);

stringgrid2.left:=850-stringgrid1.width div 2;

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=642-stringgrid1.width div 2;

end;

procedure TForm5.Button1Click(Sender: TObject);

var i,j,k,Sum,q:integer;

begin

if edit3.text<>'' then

begin

for i:=0 to StringGrid1.RowCount-1 do

for j:=0 to StringGrid1.ColCount-1 do

if StringGrid1.Cells[i,j]='' then StringGrid1.Cells[i,j]:='0';

for i:=0 to StringGrid3.RowCount-1 do

for j:=0 to StringGrid3.ColCount-1 do

stringgrid3.Cells[i,j]:=StringGrid1.Cells[i,j];

for q:=1 to strtoint(Edit3.text)-1 do

if (StringGrid1.ColCount=StringGrid1.RowCount) then

begin

Stringgrid2.RowCount:=Stringgrid1.RowCount;

Stringgrid2.ColCount:= Stringgrid2.ColCount;

Stringgrid3.RowCount:=Stringgrid1.RowCount;

Stringgrid3.ColCount:= Stringgrid2.ColCount;

for i:=0 to StringGrid3.RowCount-1 do

for j:=0 to StringGrid3.ColCount-1 do

begin

Sum:=0;

for k:=0 to StringGrid1.ColCount-1 do

Sum:=Sum+StrToInt(StringGrid1.Cells[k,i])\*StrToInt(StringGrid3.Cells[j,k]);

StringGrid2.Cells[j,i]:=IntToStr(Sum);

end;

for i:=0 to StringGrid3.RowCount-1 do

for j:=0 to StringGrid3.ColCount-1 do

stringgrid3.Cells[i,j]:=StringGrid2.Cells[i,j];

end else ShowMessage('Количество столбцов 1-ой матрицы не равно количеству строк 1-ой матрицы');

end else showmessage('введите сптепень');

end;

procedure TForm5.StringGrid1KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm5.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm5.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit2.text)=1) then key:=char(0);

end;

procedure TForm5.Edit3KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm5.Button2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

end;

end.

unit Unit6;

interface

uses

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

Dialogs, Menus, StdCtrls, ComCtrls, Grids;

type

TForm6 = class(TForm)

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N4: TMenuItem;

N3: TMenuItem;

StringGrid1: TStringGrid;

StringGrid2: TStringGrid;

GroupBox1: TGroupBox;

UpDown1: TUpDown;

Edit1: TEdit;

GroupBox2: TGroupBox;

UpDown2: TUpDown;

Edit2: TEdit;

Button1: TButton;

StringGrid3: TStringGrid;

Button2: TButton;

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);

procedure FormCreate(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Button1Click(Sender: TObject);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form6: TForm6;

implementation

uses Unit1, Unit4;

{$R \*.dfm}

procedure TForm6.N2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[i,j]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0';

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[i,j]:='0';

edit1.Text:='3';

edit2.Text:='3';

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=175-stringgrid1.Height div 2;

stringgrid2.RowCount:=strtoint(edit1.Text);

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=175-stringgrid1.Height div 2;

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid2.ColCount:=strtoint(edit2.Text);

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=850-stringgrid1.width div 2;

edit1.Text:='3';

edit2.Text:='3';

form6.Visible:=false;

form1.show;

end;

procedure TForm6.N4Click(Sender: TObject);

var i,j:integer;

begin

randomize;

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

stringgrid1.Cells[j,i]:=inttostr(random(100)-50);

end;

procedure TForm6.N3Click(Sender: TObject);

begin

form1.close;

form6.close;

end;

procedure TForm6.UpDown1Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=175-stringgrid1.Height div 2;

stringgrid2.RowCount:=strtoint(edit1.Text);

stringgrid2.Height:=37\*strtoint(edit1.Text);

stringgrid2.Top:=175-stringgrid1.width div 2;

stringgrid3.RowCount:=strtoint(edit1.Text);

stringgrid3.Height:=27\*strtoint(edit1.Text);

stringgrid3.Top:=175-stringgrid1.Height div 2;

end;

procedure TForm6.UpDown2Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

stringgrid2.ColCount:=strtoint(edit2.Text);

stringgrid2.Width:=37\*strtoint(edit2.Text);

stringgrid2.left:=466-stringgrid1.width div 2;

stringgrid3.ColCount:=strtoint(edit2.Text);

stringgrid3.Width:=37\*strtoint(edit2.Text);

stringgrid3.left:=850-stringgrid1.width div 2;

end;

procedure TForm6.FormCreate(Sender: TObject);

var i,j:integer;

begin

stringgrid1.RowCount:=9;

stringgrid1.ColCount:=9;

stringgrid2.RowCount:=9;

stringgrid2.ColCount:=9;

stringgrid3.RowCount:=9;

stringgrid3.ColCount:=9;

for i:= 0 to stringgrid3.RowCount-1 do

for j:= 0 to stringgrid3.ColCount-1 do

Stringgrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

stringgrid1.RowCount:=3;

stringgrid1.ColCount:=3;

stringgrid2.RowCount:=3;

stringgrid2.ColCount:=3;

stringgrid3.RowCount:=3;

stringgrid3.ColCount:=3;

UpDown2.Associate:=Edit2;

UpDown1.Associate:=Edit1;

end;

procedure TForm6.FormClose(Sender: TObject; var Action: TCloseAction);

begin

application.Terminate;

end;

procedure TForm6.Button1Click(Sender: TObject);

var i,j,q,w:integer;

begin

for i:=0 to StringGrid2.RowCount-1 do

for j:=0 to StringGrid2.ColCount-1 do

stringgrid2.Cells[j,i]:=StringGrid1.Cells[j,i];

for i:=0 to StringGrid3.RowCount-1 do

for j:=0 to StringGrid3.ColCount-1 do

stringgrid3.Cells[j,i]:=StringGrid2.Cells[i,j];

end;

procedure TForm6.StringGrid1KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm6.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm6.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit2.text)=1) then key:=char(0);

end;

procedure TForm6.Button2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to StringGrid3.RowCount-1 do

for j:= 0 to StringGrid3.ColCount-1 do

StringGrid3.Cells[j,i]:='0';

for i:= 0 to stringgrid2.RowCount-1 do

for j:= 0 to stringgrid2.ColCount-1 do // обнуляем все матрицы при закрытии формы

Stringgrid2.Cells[j,i]:='0';

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

end;

end.

unit Unit7;

interface

uses

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

Dialogs, StdCtrls, Grids, ComCtrls, Menus;

type

TForm7 = class(TForm)

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N4: TMenuItem;

N3: TMenuItem;

GroupBox1: TGroupBox;

UpDown1: TUpDown;

Edit1: TEdit;

GroupBox2: TGroupBox;

UpDown2: TUpDown;

Edit2: TEdit;

Button1: TButton;

StringGrid1: TStringGrid;

Memo1: TMemo;

Button2: TButton;

procedure Edit1KeyPress(Sender: TObject; var Key: Char);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);

procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);

procedure N4Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure StringGrid1KeyPress(Sender: TObject; var Key: Char);

procedure N2Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

type

Tmass = array of Real;

Tmatrix = array of Tmass;

var

Form7: TForm7;

k, j, i, n : integer;

a : Tmatrix;

det : real;

implementation

uses Unit1;

{$R \*.dfm}

procedure TForm7.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm7.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (length(edit1.text)=1) then key:=char(0);

end;

procedure TForm7.UpDown1Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.RowCount:=strtoint(edit1.Text);

stringgrid1.Height:=27\*strtoint(edit1.Text);

stringgrid1.Top:=175-stringgrid1.Height div 2;

end;

procedure TForm7.UpDown2Click(Sender: TObject; Button: TUDBtnType);

begin

stringgrid1.ColCount:=strtoint(edit2.Text);

stringgrid1.Width:=37\*strtoint(edit2.Text);

stringgrid1.left:=170-stringgrid1.width div 2;

end;

procedure TForm7.N4Click(Sender: TObject);

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

stringgrid1.Cells[j,i]:=inttostr(random(100)-50);

end;

procedure TForm7.N3Click(Sender: TObject);

begin

form1.close;

form7.close;

end;

procedure TForm7.FormClose(Sender: TObject; var Action: TCloseAction);

begin

application.Terminate;

end;

procedure TForm7.FormCreate(Sender: TObject);

begin

stringgrid1.RowCount:=9;

stringgrid1.ColCount:=9;

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[j,i]:='0';

stringgrid1.RowCount:=3;

stringgrid1.ColCount:=3;

UpDown2.Associate:=Edit2;

UpDown1.Associate:=Edit1;

end;

procedure Per(k,n:integer;var a:Tmatrix; var p:integer);

//Перестановка строк если главный элемент = 0

var z:Real;j,i:integer;

begin

z:=abs(a[k,k]);i:=k;p:=0;

for j:=k+1 to n-1 do

begin

if abs(a[j,k])>z then

begin

z:=abs(a[j,k]);i:=j;

p:=p+1;//Счестик перестановок

end;

end;

if i>k then

for j:=k to n-1 do

begin

z:=a[i,j];

a[i,j]:=a[k,j];

a[k,j]:=z;

end;

end;

function Znak(p:integer):integer;

//Для определения знака

begin

if p mod 2=0 then

result:=1 else result:=-1;

end;

procedure Opr(n:integer;var a:tmatrix;var det:real);

var k,i,j,p:integer;r:real;

begin

det:=1.0;

for k:=0 to n-1 do

begin

if a[k,k]=0 then Per(k,n,a,p);//Перестановка строк

det:=znak(p)\*det\*a[k,k];//Вычисление детерминанта

for j:=k+1 to n-1 do //Перещет коэфециентов

begin

r:=a[j,k]/a[k,k];

for i:=k to n-1 do

a[j,i]:=a[j,i]-r\*a[k,i];

end;

end;

end;

procedure TForm7.Button1Click(Sender: TObject);

var KolZero, KolEl : integer;

begin

begin

for i:=0 to StringGrid1.RowCount-1 do

for j:=0 to StringGrid1.ColCount-1 do

if stringgrid1.Cells[i,j]='' then stringgrid1.Cells[i,j]:='0';

end;

Memo1.Visible := true;

Memo1.Clear;

begin

for i:=0 to StringGrid1.RowCount-1 do

for j:=0 to StringGrid1.ColCount-1 do

if StringGrid1.Cells[j,i]='' then StringGrid1.Cells[j,i]:='0';

end;

begin

KolEl := 0; //Количество элементов матрицы

KolZero := 0; // Количество нулей в матрице

for i:=0 to StringGrid1.RowCount-1 do

for j:=0 to StringGrid1.ColCount-1 do

if StrToInt(StringGrid1.Cells[j,i]) = 0 then inc(KolZero);

KolEl := StringGrid1.ColCount \* StringGrid1.RowCount;

If KolEl = KolZero then

begin

Memo1.Lines.Add('Детерминант = 0');

exit;

end;

end;

n:=strtoint(edit1.Text);

SetLength(a,n,n); //Динамический массив

//Записывает матрицу в массив

for k:=0 to n-1 do

for j:=0 to n-1 do

a[k,j]:=strtofloat(StringGrid1.Cells[j,k]);

Opr(n,a,det);//Определитель

Memo1.Lines.Add('Детерминант = '+FloatToStr(det));

end;

procedure TForm7.StringGrid1KeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9' : ;

#8, '-' :;

#13:;

else

key:=char(0);

end;

end;

procedure TForm7.N2Click(Sender: TObject);

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0'; // обнуляем все матрицы при закрытии формы

memo1.Clear;

form7.Visible:=false;

form1.Show;

end;

procedure TForm7.Button2Click(Sender: TObject);

var i,j:integer;

begin

for i:= 0 to stringgrid1.RowCount-1 do

for j:= 0 to stringgrid1.ColCount-1 do

Stringgrid1.Cells[i,j]:='0';

end;

end.