# Основные данные о работе

|  |  |
| --- | --- |
| Версия шаблона | 2.1 |
| Филиал | Современная Гуманитарная Академия филиал в г. Сызрань |
| Вид работы | Курсовая работа |
| Название дисциплины | Программирование на языке высокого уровня |
| Тема | создайте программу ведения базы данных личной видеотеки |
| Фамилия студента | Вайс |
| Имя студента | Алексей |
| Отчество студента | Валентинович |
| № контракта | 03905100601001 |

# Содержание

Введение.... …………………………………………………………………………..3

Основная часть..…. .……………………….…………………………………....…4

* 1. 1.Описание работы программы.……………………………………………....…4
	2. 2. Структуры данных..….………………………………………………………...5
	3. Описание типов.………………………………………………………………5
	4. Описание файлов.……………………………………………………………..6

 Описание процедур.…………………………………………………………..8

* 1. Расчёт объёма оперативной памяти….………………………………….….15
	2. Интерфейс программы….…………………………………………………...16

 Граф переходов между элементами интерфейса………..…………….…...18

* 1. Минимальные системные требования….………………………………..…18

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

Глоссарий…………………………………………………………………………….20

Список используемых источников…………………………………………………22

Приложения ..….………………………………………………………….…………23

# Введение

Курсовой проект по теме: «Создать программу ведения базы данных личной видеотеки».

База данных личной видеотеки очень проста в использовании она использует мало ресурсов, работает практически на любой операционной системе и также на любом компьютере. Программа многофункциональна, внесенную информацию о видео фильмах можно как изменять, так удалять и редактировать. Все данные о дисках записывается в базу данных по пяти полям: название диска, год выпуска, кинокомпания, фамилия режиссера и цена. В этой программе диски можно сортировать по этим же полям, она написана на русском языке, что очень удобно в обращение .

 Программа разрабатывалась в среде программирования Delphi. Среда Delphi представляет собой интегрированную оболочку разработчика, в которую входит набор специализированных программ, ответственных за разные этапы создания готового приложения. Она содержит большой набор различных типов данных и компонентов, облегчающих создание программного продукта под Windows. Краткость языка, легкость программирования на нем, мобильность написанных программ, возможность эффективной реализации и пригодность с точки зрения формальных методов отладки программ обеспечили языку Object Pascal и конкретно среде Delphi успех в тех делах, для которых он предназначался. Сегодня программирование превратилось из искусства в ремесло, вряд ли можно стать профессиональным разработчиком, не изучив внутреннее устройство Windows или структуру компонентов VCL и принципы оптимизации программ, однако такие знания сегодня отходят на второй или третий план. Работодателей интересует прежде всего скорость и качество создания программ в коллективе, а эти характеристики может обеспечить только среда визуального проектирования, способная взять на себя значительные объемы рутинной работы по подготовке приложений, а также согласовывать деятельность группы постановщиков, кодировщиков, тестеров и технических писателей. Возможности Delphi полностью отвечают подобным требованиям и подходят для создания систем любой сложности. Система Delphi позволяет писать как крохотные программы и утилиты для персонального использования, так и корпоративные системы, работающие с базами данных на разных платформах, интернет - решения и коммерческие игры, распределенные ССЖ/СОЙВД/ЗОЛР - приложения и всевозможные Web-службы. При этом обеспечивается совместимость приложений при выходе новых версий Delphi — как друг с другом на уровне исходных текстов, так и с модифицированными версиями стандартных протоколов и технологий благодаря библиотеке независимых и легко настраиваемых компонентов.

# Основная часть

## 1 Описание работы программы

При запуске программы отображается главное окно, в котором имеются четыре кнопки и область обработки информации, управляемые кнопками.

**Кнопка вывода информации.**

Вся информация, находящаяся в файле и выводится в данную таблицу.

**Кнопка поиска информации.**

При нажатии кнопки поиск отображается окно поиска. Поиск происходит по пяти параметрам. Выбираем один из параметров, вводим данные и начинается поиск, найденные данные выводятся в этом же окне в таблице.

**Кнопка удаления.**

Удаление происходит следующим образам: выбираем курсором, какой DVD – фильм вы хотите удалить и следовательно нажимаем кнопку удаление.

**Область редактирования.**

Для редактирования одного из полей, необходимо ввести в первое поле его название из списка, выбрать из выпадающего списка параметр для изменения и во второе поле ввести новую информацию. Обновленная информация появляется мгновенно в таблице.

**Кнопка добавления информации.**

Отображается окно добавления информации. Информация также добавляется по пяти полям. Добавленная информация мгновенно выводится в главном окне.

## 2 Структура данных

 **Описание типов**

**type disk=record**

**name:string[50];**

**reg: string[30];**

**cincom: string[30];**

**stoim:integer;**

**year:integer;**

**end;**

Тип записи **disk** является представлением информации: **stoim** - стоимость DVD - фильма, **name** название DVD – фильма, **reg** – фамилия режиссера, **cincom** – название кинокомпании, **year** – год выхода фильма на DVD.

 **tdvd=array[1..300] of disk**

Тип tdvd является одномерным массивом, содержащим 300 элементов типа disk, т.е. этот массив содержит информацию о 300ах DVD - фильмах, полученную в результате заполнения файла с информацией.

 **recordfile=file of disk**

Типизированный файл «**recordfile**» содержит записи типа **disk** , т.е. содержит исходную информацию о каждом DVD - фильме .

**Описание файлов**

При работе программа использует постоянный файл с информацией

 **‘db\_ disk.db’** – файл с исходной информацией.

 Структурные элементы программы

Таблица 1. Глобальные переменные.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| n | integer | Переменная – счётчик |
| m | integer | Переменная – счётчик |
| dvdfile | string | Название файла |
| f | recordfile | Файл с записями  |
| g | recordfile | Файл с записями |
| Form1 | Tform1 | Идентификатор формы |
| but  | integer  | Определяет нажатую кнопку при появлении сообщения |
| h | HWND | Дескриптор окна |
| horizon | integer | Номер строки в таблице |
| k | integer | Проверка существования телефона |

Таблица 2. Константа.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Значение | Назначение |
| detect | ‘savedvd\’ | Путь к файлам необходимым для работы программы |

Таблица 3. Глобальные переменные модуля Unit2.рas.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| Form2 | Tform2 | Идентификатор формы  |

 Таблица 4. Глобальные переменные модуля unit search.pas

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| Form4 | Tform4 | Идентификатор формы  |
| dvdfile | string | Название файла |
| n | integer | Переменная – счётчик |
| f | recordfile | Файл с записями |
| g | recordfile | Файл с записями |

Таблица 5. Глобальные переменные модуля unit Input.pas

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| Form3 | Tform3 | Идентификатор формы  |
| dvdfile | string | Название файла |
| n | integer | Переменная – счётчик |
| k | integer | Переменная – счётчик |
| f | recordfile | Файл с записями |
| g | recordfile | Файл с записями |

 **Описание процедур**

 Модуль **kursovik.pas**

**procedure tform1.clearlist;**

Данная процедура очищает таблицу вывода информации.

**procedure tform1.vivod (dvdfile:string);**

Данная процедура осуществляет вывод данных записанных в файл.

Параметр **dvdfile** – директория файла.

**Алгоритм:**

 Привязываем к файловой переменной **“f”** переменную **dvdfile** и открываем на чтение. Если файл обнаружен, то в цикле, до тех пор, пока не достигнут конец файла, запускаем счётчик и выводим в ячейки таблицы данные из файла. Если счётчик равен нулю, следовательно, файл пустой, запускается процедура **clearlist**.

 В случае не обнаружения файла, он создаётся в необходимой директории.

Таблица 6. Локальные переменные.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| s | disk | Запись |
| f1 | recordfile | Файл с записями |
| count | integer | Номер по списку |

**procedure tform1.deletef (horizon:integer);**

Данная процедура осуществляет удаление записи из фала.

**Алгоритм:**

В параметр **d** передаётся введённая модель телефона.

Открываем на чтение основной файл **“db\_disk.db”** на чтение и создаём временный “**new.db”**. Перезаписываем его.

До тех пор, пока не достигнут конец файла, читаем записи и если поле записи обозначающей имя не равно переменной **“d”**, пишем во временный файл все наши записи. Удаляем основной файл и переприсваиваем его директорию временному файлу.

Таблица 7. Локальные переменные.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| el | tdvd | Запись |
| f1 | recordfile | Файл с записями |

**procedure tform1.sort\_1(dvdfile: string);**

**procedure tform1.sort\_2(dvdfile: string);**

**procedure tform1.sort\_3(dvdfile: string);**

**procedure tform1.sort\_4(dvdfile: string);**

**procedure tform1.sort\_5(dvdfile: string);**

Процедуры, перечисленные выше почти одинаковы различны только по полям сортировки Во всех процедурах используется алгоритм сортировки методом вставки. Параметр **dvdfile** – директория файла.

Таблица 8. Локальные переменные.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| f | file of disk | Типизированный файл с записями |
| q | file of disk | Типизированный файл с записями |
| c | integer | Счётчик записей |
| i | integer | Элемент массива |
| n | integer | Элемент массива |
| buf | disk | Запись |
| el | tdvd | Массив с записями |

**procedure Tform1.editzap;**

Данная процедура осуществляет редактирование данных записанных в файл.

**Алгоритм**.

Открываем на чтение основной файл с информацией и в переменную name заносим название dvd, один из параметров которого нам необходимо заменить.

Пока нет конца файла, запускаем счётчик **n**. Сравниваем введённое название, с имеющимися в dvd. При совпадении присваиваем переменной m значение счётчика. Переменная **m** будет отвечать за позицию записи в фале. Далее переменной new присваиваем данные, которые надо записать в вместо исходных.

Переходим с помощью функции **seek** на позицию **m-1**, то есть над необходимой записью. Читаем из файла запись.

Затем в **case comboBox3.ItemIndex** выбираем поле которое необходимо заменить и присваиваем ему значение переменной **new**. Если параметр не символьный переводим переменную new в цифровой формат при помощи функции.

Перезаписываем запись в файле. Закрываем его и выводим на экран при помощи процедуры **vivod**.

Таблица 9. Локальные переменные.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| f1 | recordfile | Временный файл с записями |
| q | disk | Запись |
| s | disk | Запись |
| new | string | Новые введённые данные |
| name | string | Введённое название dvd |

**procedure TForm1.FormCreate(Sender: TObject);**

Данная процедура вызывается при создании окна и запускает процедуру **nanol**. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.ComboBox2Change(Sender: TObject);**

В данной процедуре осуществляется выбор и запуск необходимой процедуры сортировки, путём нажатия одного из пунктов выпадающего списка **ComboBox**. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N8Click(Sender: TObject);**

Данная процедура завершает работу программы из меню в случае положительного ответа на появляющееся сообщение. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N7Click(Sender: TObject)**;

Данная процедура выводит информацию из файла при помощи при помощи процедуры **vivod** из меню. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

 **procedure TForm1.N14Click(Sender: TObject);**

**procedure TForm1.N15Click(Sender: TObject);**

**procedure TForm1.N16Click(Sender: TObject);**

**procedure TForm1.N17Click(Sender: TObject);**

**procedure TForm1.N18Click(Sender: TObject);**

Процедуры приведённые выше отвечают за запуск процедур сортировки **sort\_1(derect+'db\_disk.db'),sort\_2(derect+'db\_disk.db'), sort\_3(derect+'db\_disk.db')sort\_4(derect+'db\_disk.db'), sort\_5(derect+'db\_disk.db')**, из меню, соответственно. Процедуры содержат входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

 **procedure TForm1.N5Click(Sender: TObject)**;

Данная процедура отвечает за запуск процедуры удаления **deletef** и вывод обновлённой информации с помощью процедуры **vivod** из меню. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N4Click(Sender: TObject)**;

Данная процедура отвечает за добавление информации в файл с помощью процедуры **input** из меню. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N2Click(Sender: TObject)**;

Данная процедура запускает процедуру вывода **vivod**, путём нажатия на необходимый пункт во всплывающем меню над областью таблицы. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N10Click(Sender: TObject);**

**procedure TForm1.N11Click(Sender: TObject);**

**procedure TForm1.N12Click(Sender: TObject);**

**procedure TForm1.freg1Click(Sender: TObject);**

**procedure TForm1.fcincom1Click(Sender: TObject);**

Процедуры, приведенные выше, сходны по своему принципу. Каждая из них вызывает конкретную процедуру сортировки (**sort\_1, sort\_2, sort\_3, sort\_4, sort\_5**), путём нажатия на пункт, всплывающего меню над областью таблицы. Процедура содержит входную переменную **Sender** - тип t**object**, этот параметр определяет, какой объект программы вызывает данную процедуру.

 **procedure TForm1.N9Click(Sender: TObject);**

Данная процедура запускает процедуру очистки списка **clearlist**, путём нажатия на необходимый пункт во всплывающем меню. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N23Click(Sender: TObject)**;

Данная процедура запускает процедуру очистки списка **clearlist**, путём нажатия на необходимый пункт в меню. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N6Click(Sender: TObject);**

Данная процедура запускает процедуру редактирования **redactor** из меню. Процедура содержит входную переменную **Sender** - тип **tobject**, этот параметр определяет, какой объект программы вызывает данную процедуру.

**procedure TForm1.N21Click(Sender: TObject);**

Данная процедура отображает окно с информацией о программе.

**procedure tform3.input(var f:recordfile; var dvdfile:string)**;

Данная процедура осуществляет запись данных в файл.

Параметр **“f”** – типизированный файл типа **recordfile**. **Dvdfile** – деректория файла.

**Алгоритм:**

Привязываем к файловой переменной **“g”** временный файл **derect+'buffer.db'** и перезаписываем его. Привязываем к файловой переменной **“f”** переменную **dvdfile** и открываем на чтение.

Если в файле находится что-либо, копируем содержимое **“f”** в **“g”** и закрываем **“f”**, если нет, перезаписываем и закрываем **“f”**.

Присваиваем переменным записей введенные поля.

Проверяем наличие введённого dvd в файле, путем сравнения названия введённой диска с уже находящимися в файле в цикле достижения конца файла. Если поля равны, присваиваем переменной **k** значение **1**, если нет, то оставляем по умолчанию равной нулю.

Далее проверяем значение переменой **k**. Если **k=1**, то выводится сообщение о том что данный диск существует и запись не осуществляется, если иначе, то происходит запись в файл.

В конце закрываем **“g”** и удаляем **“f”**. Переприсваиваем директорию **dvdfile** переменной **“g”**. И выводим обновлённый список с помощью процедуры **vivod**.

Таблица 10. Локальные переменные.

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип | Назначение |
| s | disk | Запись |
| q | disk | Запись |

**procedure tform1.search\_1(dvdfile:string; var g:recordfile);**

**procedure tform1.search\_2(dvdfile :string; var g:recordfile);**

**procedure tform1.search\_3(dvdfile:string; var g:recordfile);**

**procedure tform1.search\_4(dvdfile:string; var g:recordfile);**

**procedure tform1.search\_5(dvdfile:string; var g:recordfile);**

Процедуры, приведены выше, почти одинаковы и различается лишь поле, по которому происходит поиск. Во всех процедурах, кроме второй, на экран выводится только те записи, поля которых полностью совпадают, с введёнными нами данными. В третьей, четвёртой и пятой процедурах список выводится отсортированным по названию модели. Параметр **“g”** – типизированный файл типа **recordfile**. Параметр **dvdfile** – директория файла.

**procedure TForm1.Button4Click(Sender: TObject)**;

 Данная процедура отвечает за функцию изменения информации. При нажатии на кнопку курсором вызывается процедура **editzap**.

**procedure TForm1.Button7Click(Sender: TObject);**

 Данная процедура закрывает полностью программу. При нажатии на кнопку отображается окно подтверждения.

 **Расчёт объёма оперативной памяти**

Для определения потребляемой памяти, я использовал стандартный диспетчер задач Windows. Исходя из данных показанных в нём, в процессе работы программа максимально потребляет 5000 кб.

**Интерфейс программы**

При запуске программы появляется окно, с дружественным интерфейсом рис. 1. Все операции выполняются в данном окне. При вызове информации о программе в пункте меню появляется окно, содержащее необходимую информацию рис. 2. Функция добавления изображена на рис. 3. Функция поиска изображена на рис. 4.



Рисунок 1. Главное окно

Рисунок 2.. Информация о программе.

Рисунок 3. Окно добавления новой информации.

Рисунок 4. Окно поиска информации

**Граф перехода между элементами интерфейса.**

 Вход

 Главное окно

 Редактирование

 Вывод Удаление

 списка

 О программе Добавление Поиск

Рисунок 5. Граф перехода между элементами интерфейса.

* 1. **Минимальные системные требования**

1.Процессор: 170 MHz;

2. Оперативная память: 5000kb;

3. Физическая память на жестком диске: 1400MB свободного места;

4. Видеокарта с поддержкой VGA;

5. Устройства ввода/вывода данных: клавиатура, монитор, мышь.

# Заключение

 Программа «личная видеотека» очень проста в использовании и использует очень мало ресурсов. Работает почти на любой операционной системе и на любом компьютере. Программа многофункциональна. Занесенную информацию о DVD – фильмах можно изменять, удалять и редактировать. Информация о дисках записывается в базу данных по пяти полям: название диска, цена, год выпуска, фамилия режиссера, кинокомпания, которая создавала фильм. Также можно сортировать диски по этим же полям, она написана на русском языке. Процесс создания этого приложения в Delphi учит работать с процедурами, с записями и с циклами.

Очень простой интерфейс позволяет работать в программе даже ребенку, и старшему поколению, которому очень трудно обращаться с новыми технологиями, в том числе и компьютерами.

# Глоссарий

|  |  |  |
| --- | --- | --- |
| № п/п | Понятие | Определение |
| 1 | База данных | поименованная, целостная, единая система данных, организованная по определенным правилам, которые предусматривают общие принципы описания, хранения и обработки данных |
| 2 | Банк данных | база данных, объединенная с системой управления базой данных |
| 3 | Данные | последовательность элементарных символов, цифр или букв, являющихся значением некоторого атрибута |
| 4 | Логическое проектирование баз данных | процесс конструирования общей информационной модели предприятия на основе отдельных моделей данных пользователей, которая является независимой от особенностей реально используемой СУБД и других физических условий |
| 5 | Сетевая схема | логическая организация всей базы данных в целом (с точки зрения администратора базы данных), которая включает определение имени базы данных, типа каждой записи и компонентов записей каждого типа |
| 6 | Система управления базами данных | специальный комплекс программ, осуществляющий централизованное управление базой данных |
| 7 | Подсхема | часть базы данных, как она видится пользователям или приложениям |
| 8 | Информация | любые сведения о каком-либо событии, процессе и т.п., являющиеся объектом некоторых операций: восприятия, передачи, преобразования, хранения и использования |
| 9 | Централизованная база данных | база данных, хранящаяся в памяти одной вычислительной системы |
| 10 | Распределенная база данных | состоит из нескольких, возможно, пересекающихся или даже дублирующих друг друга частей, хранимых в различных ЭВМ вычислительной сети |
| 11 | Хранимая процедура | это программа, которая обрабатывает участок базы данных и хранится в базе данных. |

# Список использованных источников

|  |  |
| --- | --- |
|  1. | Михаил Фленов, «Библия Delphi», Санкт Петербург «БХВ- Петербург» 2004 год, 882 стр.  |
| 2. | Владимир Гофман, Анатолий Хамоненко, «Delphi быстрый старт», Санкт Петербург «БХВ-Петербург» 2003 год, 280 стр. |
| 3. | Евгений Марков, Петр Дарахвелидзе, «Программирование в Delphi 7», Санкт Петербург «БХВ-Петербург» 2003 год, 780 стр. |
| 4. | Поган А.М., Царенко Ю.А., “Программирование в Delphi”, Москва «ЭКСМО» 2006 год, 309 стр. |
| 5. | Бобровский А. , “Delphi 5. Учебный курс.” , Санкт – Петербург, Москва, Харьков, Минск, «ЗАО Питербук» , 2004 год, 639 стр. |
| 6. | Turbo Pascal в задачах и программах. – СПб.: БХВ – Петербург,, 2002. – 560 с.: ил |
| 7. | Turbo Pascal / С.А. Немнюгин. – СПб: Издательство «Питер», 2000. – 496 с.: ил. |
| 8. | http://www.wikipedia.com/ru |
| 9. | Власов А.Я. Справочник по программированию на Object Pascal. Киев, 2001г. |
| 10. | Культин Н. Программирование на Object Pascal. Киев, 1998 |

# Приложения

|  |  |
| --- | --- |
| Приложение А |  |
| Приложение Е |  |