

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНА МЕТАЛУРГІЙНА АКАДЕМІЯ УКРАЇНИ**

Г. Г. Швачич, О. В. Овсянніков, Л.М.Петречук

Методичні вказівки до виконання лабораторних робіт з дисципліни

Електронне документознавство

для студентів спеціальностей 8.02010501 «Документознавство та інформаційна діяльність»

Частина 1

Дніпропетровськ НМетАУ 2012

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНА МЕТАЛУРГІЙНА АКАДЕМІЯ УКРАЇНИ**

Г. Г. Швачич, О. В. Овсянніков, Л.М.Петречук

Методичні вказівки до виконання лабораторних робіт з дисципліни

Електронне документознавств

для студентів спеціальностей 8.02010501 «Документознавство та інформаційна діяльність»

Дніпропетровськ НМетАУ 2012

УДК 004 (075.8)

Г.Г. Швачич, О.В. Овсянніков, Л.М.Петречук. Електронне документознавство. Методичні вказівки. – Дніпропетровськ: – НМетАУ, 2012. – 38 с.

Викладені основи ознайомлення й роботи з додатком
Microsoft Office - Outlook.

Призначений для студентів спеціальності
«Документознавство та інформаційна діяльність».

Лл. 31. Бібліогр.: 5 найм.

Відповідальний за випуск

Г.Г. Швачич, канд. техн. наук, проф.

Рецензенти: Б. И. Мороз д-р техн. наук, проф. (Академія митної
Служби України)

Т. И. Пашова канд. техн. наук, доц. (Дніпропетровський державний аграрний
університет)

© Національна металургійна
академія України, 2012

Лабораторная работа №1

Тема: Основные элементы интерфейса приложения *Microsoft Outlook*.
Работа с папкой **Календарь**.

Рассмотрим в качестве примера универсальной системы электронного документа оборота (СЭДО) стандартное приложение пакета Microsoft Office – Outlook.

Microsoft Outlook предназначен для управления перепиской, личными сведениями и организацией рабочей деятельности. Это универсальная записная книжка, электронный ежедневник, средство автоматизации процедур планирования и контроля рабочей деятельности.

Пользуясь Outlook пользователь может:

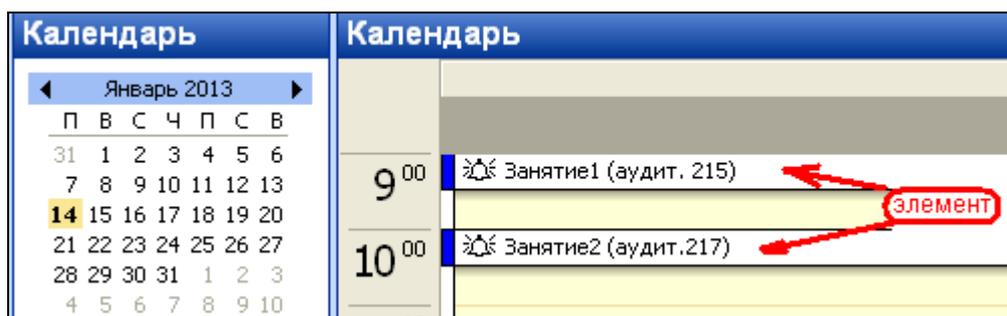
- принимать, отправлять и пересылать почтовые сообщения,
- вести адресную книгу абонентов,
- сортировать полученную информацию,
- работать с факсимильными сообщениями.

Интерфейс Outlook стандартный, как у всех Windows–приложений: строка заголовка, строка главного меню, устанавливаемые панели инструментов.

Действия, совершаемые и сохраняемые с помощью приложения, выполняются в логически соответствующих папках:

- задачи,
- дневник,
- заметки,
- календарь,
- контакты, и др.

Объекты (записи, строки), создаваемые в папках принято называть **элементами**.



Перечень папок Outlook доступен из меню *Вид* → *Область переходов* (или Alt+F1), здесь же осуществляется переход по папкам.

ПАПКА КАЛЕНДАРЬ

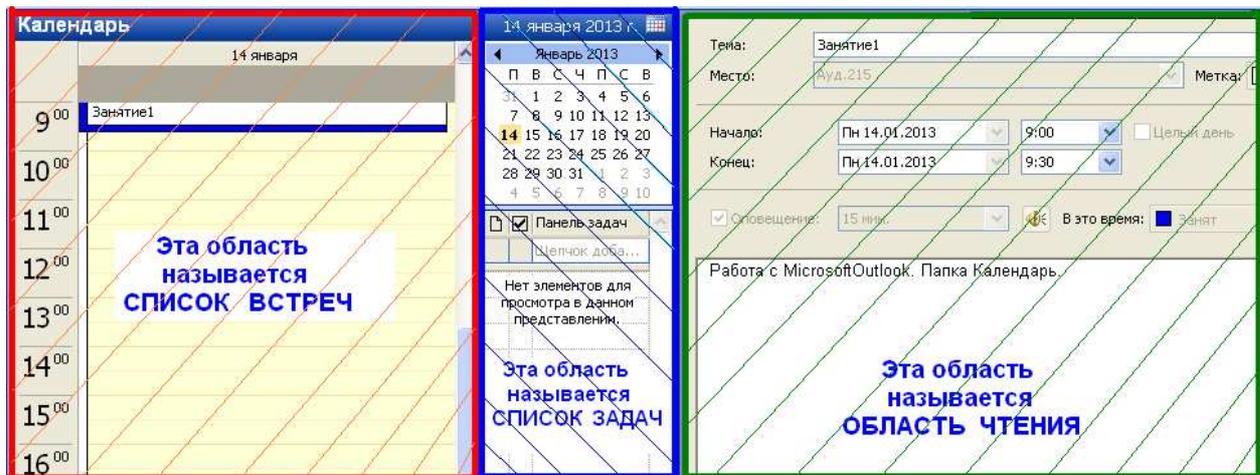
Внешний вид рабочей области папки Календарь напоминает страницу бумажного ежедневника, эта главная область называется *Список встреч*. Рабочий лист каждой из папок Outlook может иметь несколько *представлений*.

Представление – это отображение определенной выборки элементов в специальном графическом формате (формат отображения и предоставления

информации). Представления изменяют способ просмотра данных и помогают увидеть информацию так, как это необходимо в данный момент.

Один из способов изменить текущее представление: меню *Вид* → *Упорядочить по* → *Текущее представление* или соответствующим элементом на панели инструментов.

Так же, важное значение имеют *Область чтения* и *Список задач* (меню Вид). Иногда *Список задач* скрыт под *Списком встреч*.



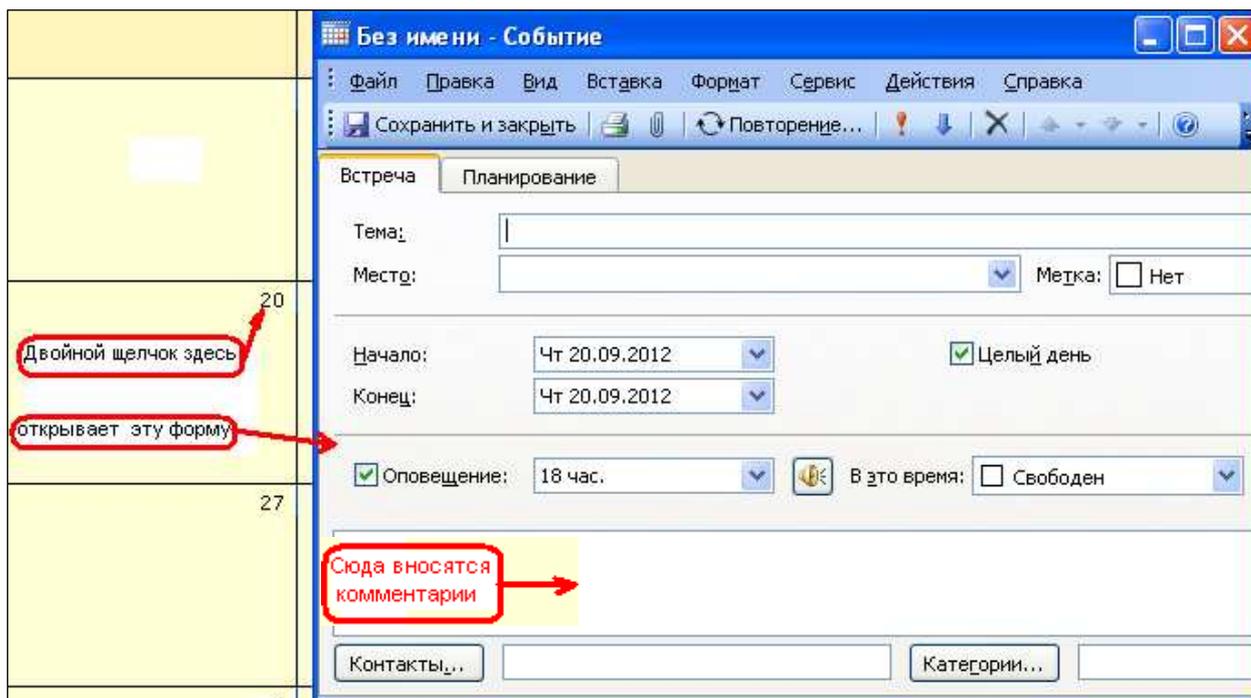
В соответствии с поставленной задачей выбирается определенное представление (День/Неделя/Месяц) и заполняется элемент календаря в поле *Список встреч* (толи почасовое заполнение, толи каждодневное).

Для заполнения (создания) элемента Календаря необходимо вызвать специальную форму. Она вызывается одним из способов:

- меню Файл → Создать → Встреча (или Задача...);
- одноименной кнопкой на панели инструментов  ;
- двойным щелчком по выбранному полю заполнения.

Обязательные для заполнения поля:

- Тема;
- Место;
- Начало : Конец;
- Категория.



Категория – это средство Outlook для упорядочивания элементов и структурирования информации

После заполнения окна данными, необходимо выполнить сохранение, нажав на панели инструментов (ПИ) кнопку **Сохранить и Заккрыть**.

Для удаления элемента (записи в календаре), надо выделить этот элемент мышкой и удалить одним из способов:

- клавишей Delete;

- кнопкой **Удалить**  (в контекстном меню (К/М) удаляемого элемента или непосредственно в форме представления).

Удаленные элементы не исчезают навсегда, а попадают в папку Удаленные.

Чтобы восстановить удаленный элемент надо:

- зайти в паку Удаленные;

- выделить восстанавливаемый элемент и вызвать для него контекстное меню;

- выбрать пункт Переместить в папку и в открывшемся окне выбрать папку, в которую необходимо восстановить удаленный элемент.

Любое резервирование времени в расписании в папке Календарь называется *Встречей* или *Событием*.

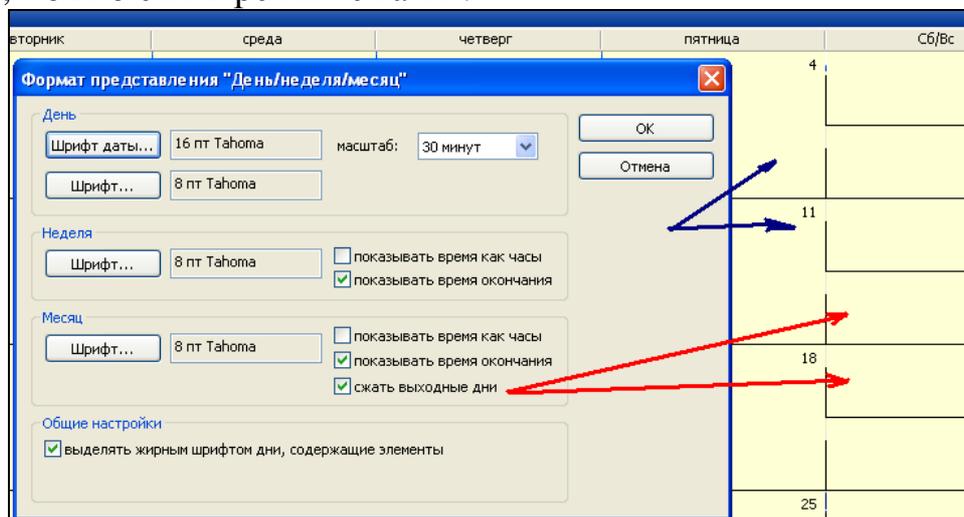
Встреча – это мероприятие, для которого не привлекаются специальные ресурсы.

Событие – то мероприятие, продолжительностью более суток (24 часа): выставки-ярмарки, отпуск, Олимпийские игры.

Если для определенной встречи (события) необходим какой либо дополнительный материал, то он «прикрепляется» к полю комментариев инструментом «скрепка» .

Флажок «Оповещение» регулирует появления окна напоминания для определенной записи в календаре (или в задачах).

В представлении **31 Месяц** область для заполнения информации в дни выходных сжата. Вызвав К/М на поле *Списка встреч* и выбрав пункт *Другие настройки*, можно снять режим сжатия.



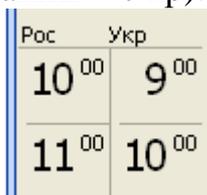
НАСТРОЙКА ВРЕМЕННОЙ ШКАЛЫ И ЧАСОВЫХ ПОЯСОВ

Временная шкала по умолчанию разбита на 30-минутные промежутки и отображает один часовой пояс. Вызвав *на ней* контекстное меню, можно изменить разбивку промежутков и изменить (или добавить второй) часовой пояс.



Еще один способ установить новый часовой пояс: Сервис → Параметры → Настройки → Параметры календаря → Параметры календаря → Часовой пояс. Используя этот же путь можно изменить цветовой фон области *Список встреч*.

Внимание! При установке отображения нескольких часовых поясов во избежание неясностей необходимо заполнять поля **Метка** (напр.: для российского региона - Рос, для Украины – Укр).



АВТОМАТИЧЕСКОЕ ДОБАВЛЕНИЕ (УДАЛЕНИЕ) ПРАЗДНИКОВ

Сервис → Параметры → Настройки → Параметры календаря → Параметры календаря → Добавить праздники.

Чтобы удалить отображение праздников в календаре необходимо перейти из представления День/Неделя/Месяц в представление *По категориям* и вызвав контекстное меню на папке Категории: Праздники – удалить её.

Задание:

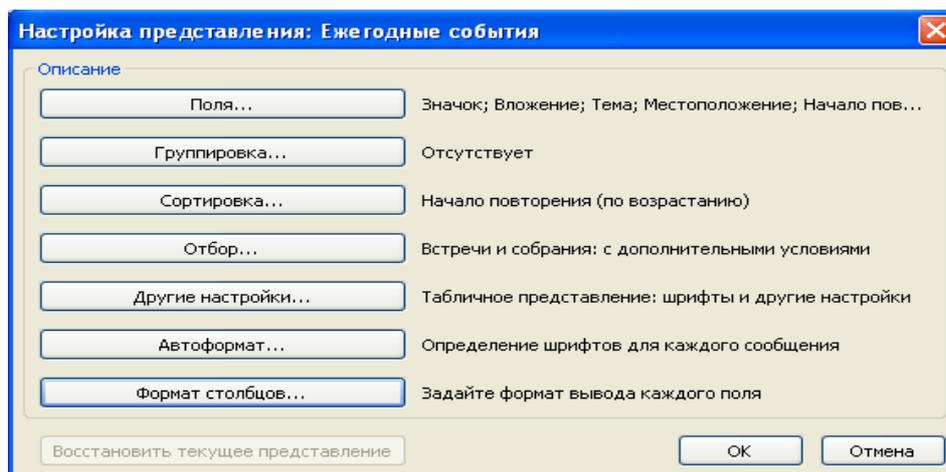
1) Заполнить **календарь** на ближайшие четыре месяца следующей информацией:

- встреч (учебные мероприятия: расписание занятия в академии, различного вида семинары, олимпиады; оздоровительные мероприятия: периодическое посещение конкретного клуба, салона, стадиона, бассейна);
- три вида повторяющихся событий;
- три не повторяющиеся события;
- пять не повторяющихся встреч;
- установить в Календаре праздники России;
- установить дополнительный часовой пояс в Календаре.
- Также внести в календарь десять событий (встреч), на временной промежуток пять лет (например: с 2008 по 2012 годы дни рождения родителей, друзей).

После внесения всей информации в папку Календарь просмотрите ее в разных представлениях: Вид → Упорядочить по → Текущее представление: День/Неделя/Месяц, активные встречи, события... Обратите внимание на различный формат отображения информации, умейте **сформулировать** ответ на вопрос о необходимости использования (применения) различных представлений.

!!!! *Периодические* (повторяющиеся) мероприятия должны быть занесены технически *грамотно*! Все заполнения должны быть определены в поле Метка и отнесены к определенной *категории*.

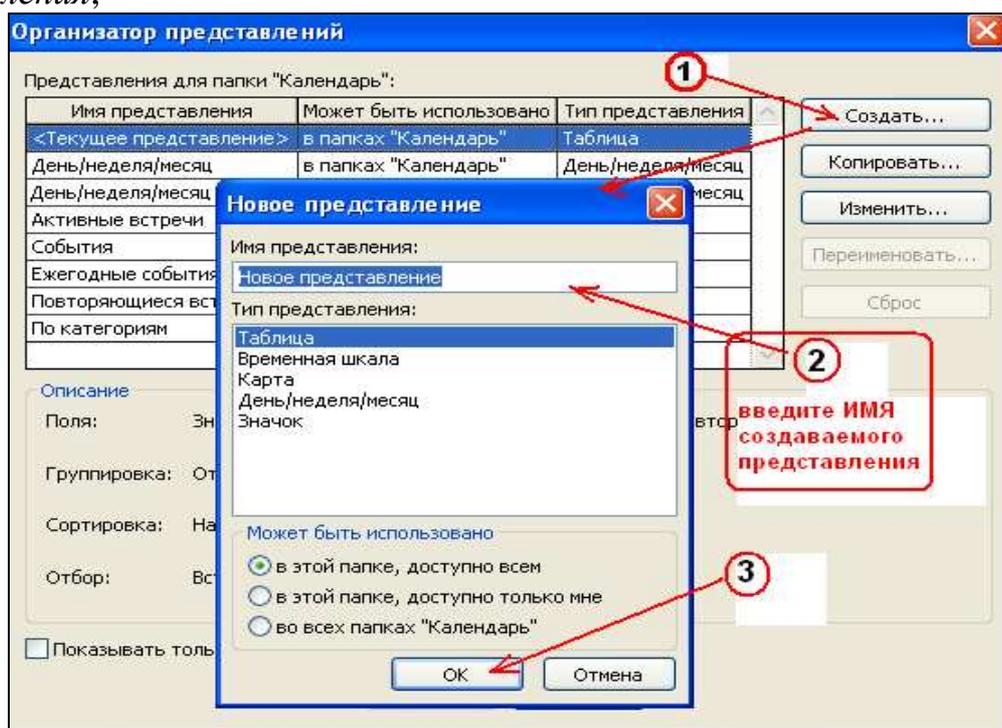
- Используя путь Вид → Упорядочить по → Текущее представление → Изменить текущее представление вызовите окно *настройки представлений* и **отработайте** каждый его элемент: поля, группировка... Это окно является инструментом для изменения формата текущего представления (добавляются/убираются поля, отображаемая информация группируется по определенным критериям и т.п.).



- Создайте три вида (таблица, временная шкала, карта или любые др.) новых представлений с помощью Мастера «Организатор представления»: Вид → Упорядочить по → Текущее представление → Определить представления.

Порядок работы таков:

- нажатие кнопки **Создать...** (1) вызывает окно «Новое представление» (2), в котором заполняется поле «Имя представления» и выбирается тип представления;



- нажатие кнопки **OK** (3) вызывает окно «Настройка представления». Используя инструменты этого окна, создается **новое представление**.

В результате выполненной работы студент должен знать и уметь:

- что такое *представление*;
- перечислите *виды представлений* и дайте их краткую характеристику;
- уметь изменить *формат* предложенного представления или создать новое представление;

- что такое *встреча, событие*;
- что такое «*Оповещение*», способы его настройки и отключения;
- для чего заполняется поле «*Категория*», достоинства заполнения этого поля;
- как внести в календарь элемент *повторяющееся* событие (встречу) и как его удалять;

-как изменить цвет области *Список встреч*;

| Календарь | | Календарь | |
|-------------|---------|-------------|---------|
| понедельник | вторник | понедельник | вторник |
| 1 августа | 2 | 1 августа | 2 |
| | было | стало | |

-как выделить другим цветом определенную запись в ячейке Календаря;

| | |
|------------------------------|----|
| 14 | 15 |
| 9:00 9:30 Занятие1 (Ауд.215) | |

- как *восстановить* удаленный из рабочей папки (Календарь, Задача...) элемент;
- как *перейти* из одного вида *представления* в другой;
- объяснить назначение каждого из **ВИДОВ представлений**;
- изменять **текущее** и создавать **новое представление**;
- как убрать режим *сжатия* в отображении *выходных дней* в поле *Список встреч*;
- знать, как устанавливается «*Область переходов*» и уметь объяснить назначение всех ее элементов;
- через какой пункт меню можно убрать(поставить) отображение *Текущего представления* в «*Область переходов*».
- через какой пункт меню можно убрать(поставить) отображение *Списка задач*, его назначение;
- через какой пункт меню можно убрать (поставить) отображение *Области чтения*, её назначение.

Тема:

Лабораторная работа №2
Работа с папкой *Задачи*

Задачей в терминологии Outlook называется *поручение* личного или служебного характера, *выполнение которого можно проследить*. *Задача* может быть *разовой* (единичной) или *повторяющейся*.

После открытия папки **Задачи**, элементы в ней создаются аналогично элементам папки **Календаря**.

Поля для заполнения:

- тема;
- дата начала (сегодня, завтра, неделя);

Если у Задачи не установлены ни срок окончания ни дата начала, такая задача называется бессрочной.

– срок (сегодня, месяц, неделя), имеется в виду конечная дата выполнения задачи;

– состояние, важность, готово (эти поля предназначены для отслеживания хода выполнения задачи);

– оповещение (устанавливается на определенное время);

– поле «Ответственный» недоступно для редактирования, подразумевается, что при создании задачи ее будет выполнять сам пользователь (он и есть ответственный за задачу). Значение поля может быть изменено, но мы не рассматриваем этот случай;

– категория;

– контакты (визитки), если папка **Контакты** имеет наполнение;

поля на дополнительной вкладке *Подробно*:

– объем работ, реально затрачено;

– дата завершения (поле содержит фактическую дату завершения задачи, которая никак не связана со сроком и датой начала задачи);

– организация;

– расстояние, расходы.

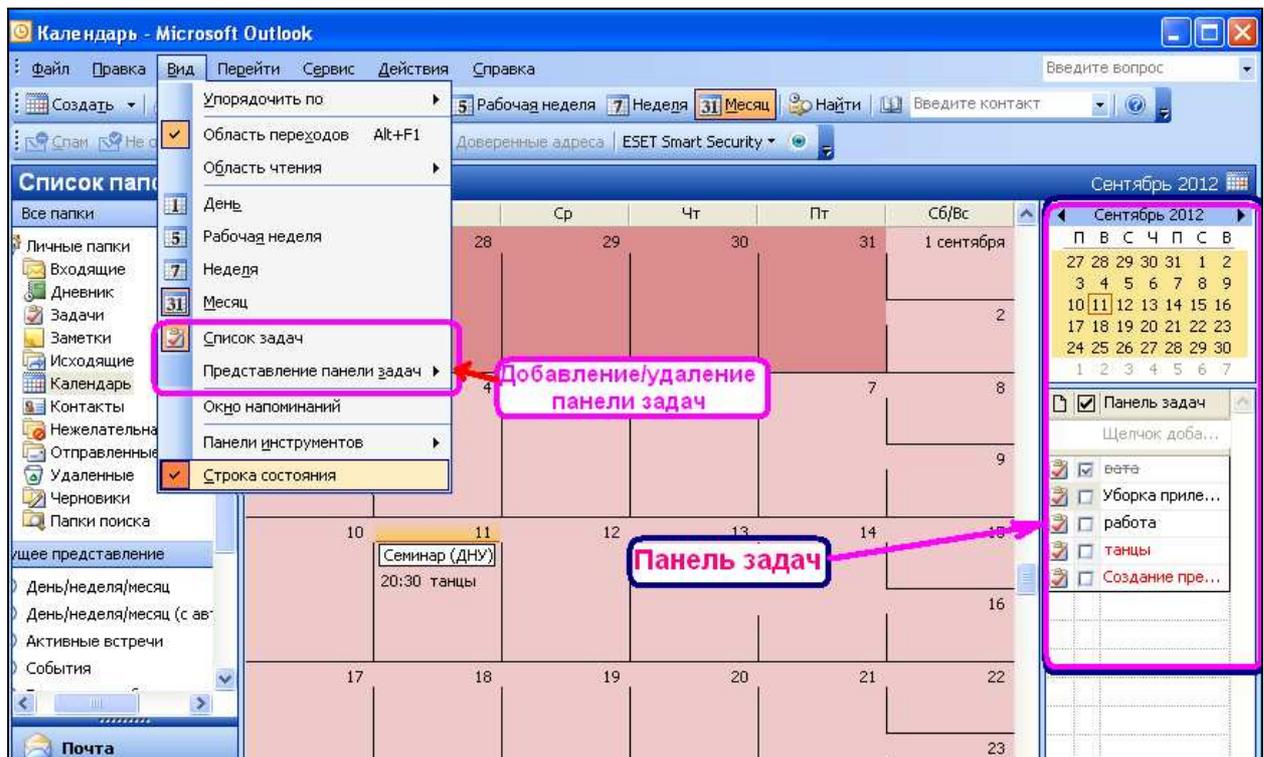
Постановка любой задачи может предполагать наличие разъяснений, инструкций или другой дополнительной информации, которая вводится в поле *Заметки* при нажатии кнопки *Вложить файл* (иконка в виде скрепки).

Повторяющиеся задачи создаются аналогично *повторяющимся встречам* (событиям), но для задач устанавливаются два принципиально разных режима повторения. В первом режиме повторение происходит в заранее *фиксированные* дни, а во втором – *привязывается к дате* завершения предыдущей задачи.

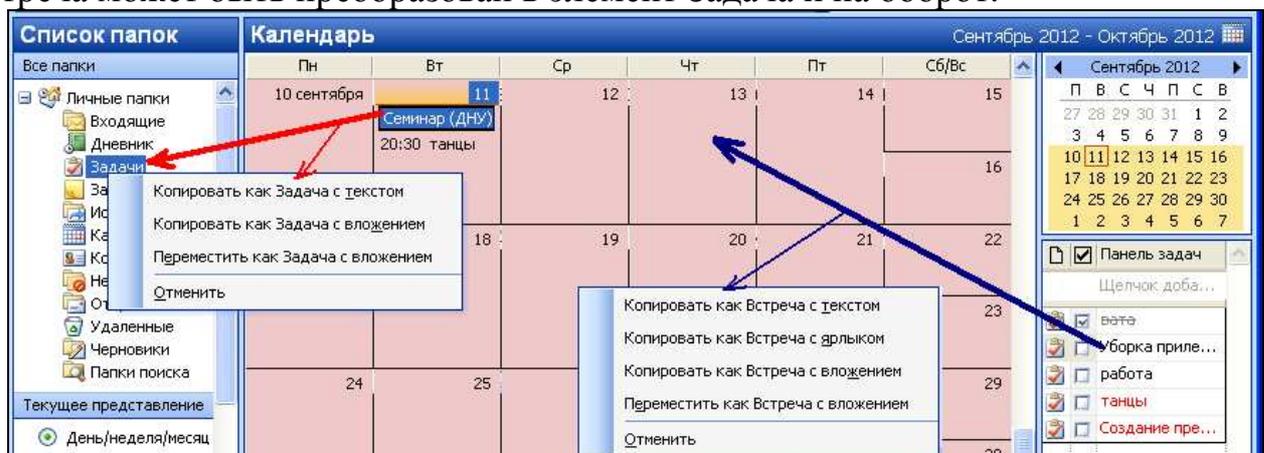
При работе с элементами папки *Задача* добавлено *представление* **Временная шкала для задач**. В этом *представлении* задачи расположены в виде цветной полосы (серого или синего цвета), проведенной вдоль всего временного отрезка, отведенного на решение задачи. У этого представления есть особенность, если у задачи не занесено значение в поле **Срок**, то задача в *представлении* **Временная шкала для задач** не отображается.

Отображение задач в календаре

Список задач можно увидеть в представлении *День/Неделя/Месяц* папки *Календарь*. Для этого используется *Панель задач* (если она не видна, установите ее из меню *Вид*).



Outlook позволяет изменять первоначальные типы элементов. Элемент Встреча может быть преобразован в элемент Задача и на оборот.



Откройте папку календарь. Проверьте, чтобы *панель задач* была установлена. Подведите мышку к одному из элементов папки, нажмите **правую** клавишу мыши и удерживая ее перетащите на папку Задачи. После отпущения мыши откроется контекстное меню, его пункты означают следующее:

- *копировать как задача с текстом*: копируется полное описание встречи (без вложений) в задачу. Поле описания задачи содержит текстовое описание всех полей встречи;
- *копировать как задача с ярлыком*: создается задача с теми же параметрами, что и выше, только вместо текста осуществляется вложение ярлыка элемента Календаря;
- *копировать как задача с вложением*: создается задача с теми же параметрами, что и выше, только вместо текста осуществляется вложение элемента Календаря;

- *Переместить как задача с вложением*: встреча удаляется из расписания, а поле описания задачи содержит внедренный объект.

Выполнить. Создать в папке **Задачи** новые элементы, отображающие Ваши планы и ход их выполнения (напр. каждые 24 дня проходит семинар по направлению «Новые технологии ведения «Электронного документооборота» », или предстоит поездка в Париж на международный студенческий симпозиум, или вы посещаете курс занятий по обучению иностранным языкам и т.п.). Должно быть создано на ближайшие четыре месяца:

- четыре разных вида повторяющихся задач (два из них должны быть завершены);

- шесть обычных задач (три из них должны быть завершены);

При заполнении элементов папки Задачи, для каждого элемента (записи):

- должна существовать дополнительная информация в виде прикрепленных файлов;

- установлена отметка о *проценте готовности* выполняемой задачи и указано *состояние*, в котором находится процесс реализации данной задачи.

После завершения данной лабораторной работы студент должен знать ответы на такие вопросы:

- что такое задача в терминологии Outlook;

- какие типы задач Вы знаете;

- чем отличаются значения в полях *срок* и *дата завершения*;

- как просмотреть только завершенные (или только просроченные задачи);

- какие задачи не отображаются в *представлении* **Временная шкала** для задач;

- как преобразовать элемент Календаря в Задачу и наоборот, Задачу в элемент Календаря.

Лабораторная работа №3

Тема:

Работа с папкой Контакты

Папка Контакты

Папка Контакты служит хранилищем деловых и личных сведений о людях, с которыми требуется поддерживать связь.

Элемент контакт содержит около 100 стандартных полей со сведениями о корреспонденте. При создании элемента папки Контакт практически все поля должны быть заполнены.

Папка Контакты открывается в представлении *Адресные карточки*.

Двойной щелчок на свободной области *панели просмотра информации* или

щелчок по кнопке  (Ctrl+N) открывает форму для заполнения сведений о контакте. Форма имеет пять вкладок: Общие, Подробности, Действия, Сертификаты, Все поля.

Совет!!! Если возле поля установлена кнопка , то заполнение данных желательно проводить в открывшейся, после нажатия кнопки, форме.

Вкладка Общие содержит основные сведения о контакте.

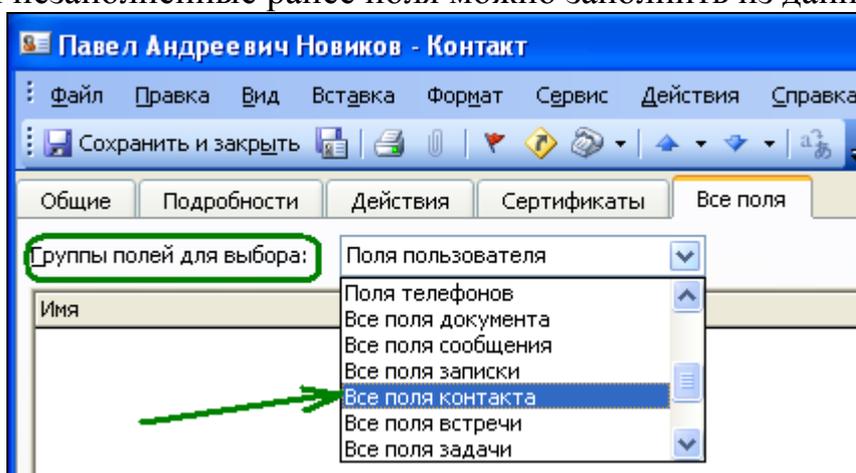
Вкладка Подробности дополняет общую информацию о контакте. Причем, после заполнения полей *День рождения* и *Годовщина* данные о контакте **автоматически** становятся связанными с папкой Календарь (откройте папку календарь и проверьте это).

Вкладка Действия отображает все действия, связанные или назначенные контакту (встречи, контакты, задачи и т.п.). Информация выводится в форме таблицы, состоящей из трех столбцов: значок элемента, тема, папка:

| Тема | Папка |
|---------------------------------------|-----------|
| Павел Андреевич Новиков День рождения | Календарь |
| Павел Андреевич Новиков Годовщина | Календарь |

Вкладка Сертификаты служит для назначения определенного цифрового сертификата безопасности при отправке сообщений данному контакту.

Вкладка Все поля дает возможность пользователю вводить и просматривать информацию, не отображаемую в виде отдельных полей на вкладках формы Контакт. Выберите в поле *Группы полей для выбора* значение *Все поля контакта* и незаполненные ранее поля можно заполнить из данной формы.



Если есть необходимость внесения информации, для которой не существует стандартного поля, то можно создать его специально (щелчок по кнопке **Создать**). Так появляются *Поля пользователя*.

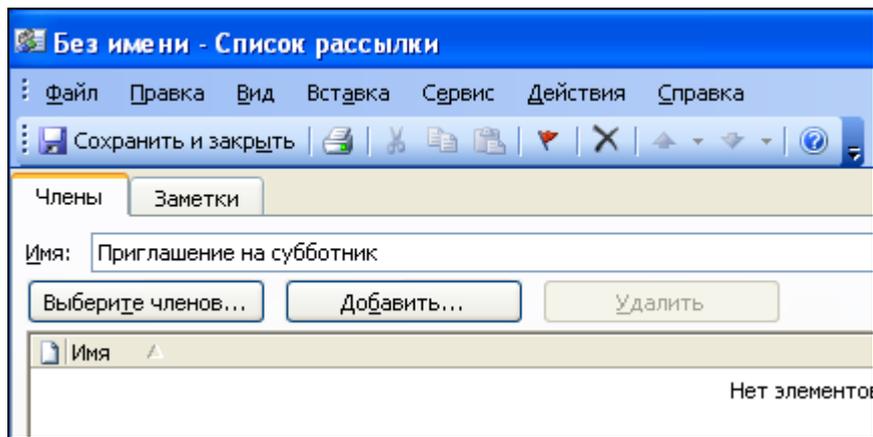
ВЫПОЛНИТЬ:

1. Зарегистрировать на Mail.ru почтовый ящик. Адрес записать на доске в классе для возможности его использования одноклассниками на практическом занятии.
2. Создать десять контактов, занося при этом в поле *электронная почта* достоверный электронный адрес **контакта** (адрес указан на классной доске).

СПИСОК РАССЫЛКИ

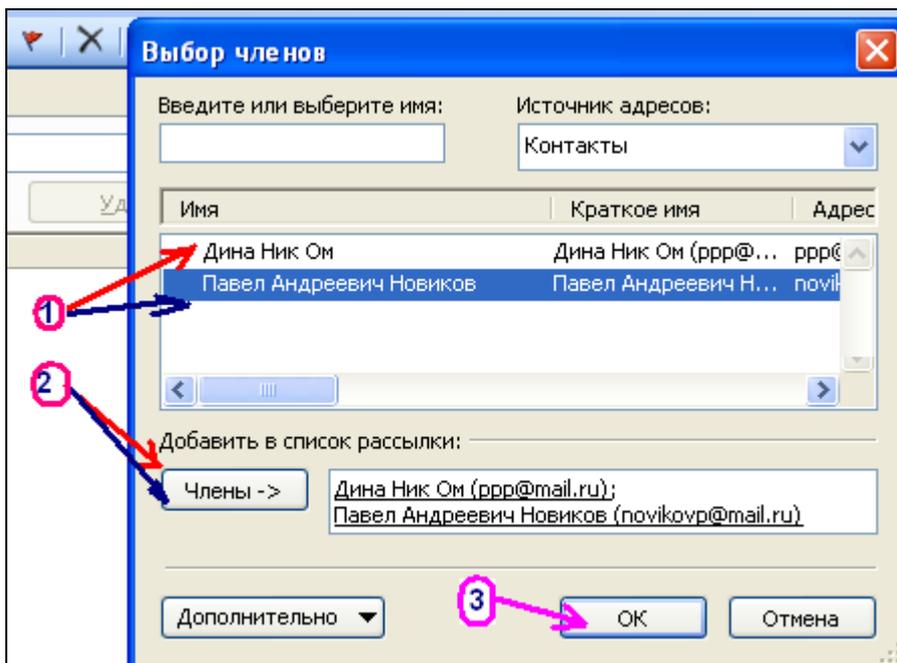
Очень часто на практике требуется выполнение одних и тех же процедур для нескольких контактов (например отправка сообщения или приглашения на собрание всем участникам проекта). OUTLOOK предоставляет возможность объединения контактов в специальный *список рассылки*. В качестве адресата при выполнении той или иной операции достаточно будет один раз выбрать *список рассылки* и, таким образом, избежать многочисленных повторов одних и тех же действий.

В папке Контакты выполнить команду *Действия* → *Создать список рассылки*.



В поле *Имя* обязательно вводите тематику списка рассылки. Далее нажав на кнопку **Выберите членов** в открывшейся форме составляете *список рассылки* из имеющихся *контактов*. Желательно, на страничке Заметки, внести комментарии для создаваемого списка рассылки.

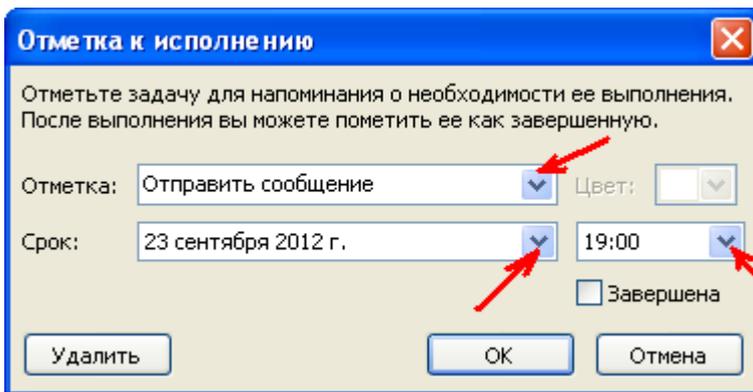
Нажатие на кнопку **Добавить** позволяет не только добавить новую запись в строку адресов, но и создать новый элемент в папке Контакты.



ОТМЕТКА К ИСПОЛНЕНИЮ

Команда *ОТМЕТКА К ИСПОЛНЕНИЮ*  используется для оповещения пользователя о запланированном действии для контакта.

Выберите в папке Контакт определенный элемент и в пункте меню *Действия* команду *К исполнению*. Появится диалоговое окно:



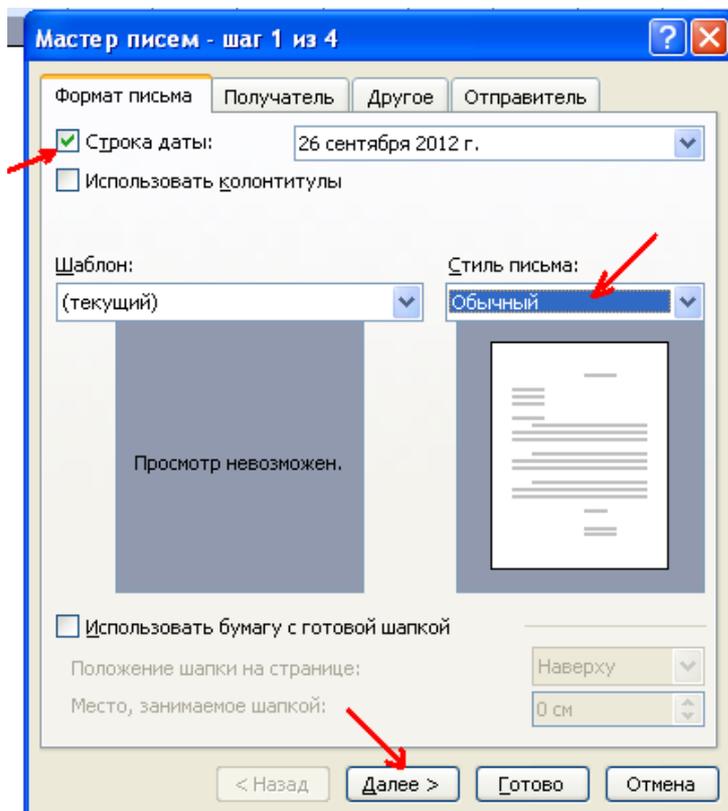
После создания отметки *К исполнению* в форме контакт появится информационная строка с предупреждением о выполнении действия до определенного срока.

Для удаления всей информации об отметке *К исполнению* щелкните на кнопке **Удалить** в диалоговом окне *Отметка к исполнению*.

ПИСЬМО КОНТАКТУ

OUTLOOK может быть полезен для создания письма адресату, сведения о котором есть в папке Контакты. Для этого:

- выделите контакт адреса в папке Контакты;
- выполните команду Действия → Новое письмо. Для создания письма используется Мастер писем, в котором пошагово составляется письмо.



После того, как все параметры заданы, Мастер писем генерирует по ним письмо, в которое остается лишь ввести текст сообщения на месте, выделенном текстом *Введите текст*.

Контрольные вопросы:

- функциональное назначение папки Контакты;
- представления папки Контакты;
- обязательные поля для заполнения папки Контакты;
- что такое список рассылки и как он составляется;
- отметка К исполнению – для чего она нужна и как она работает;
- создание письма адресату с помощью Мастера.

Лабораторная работа №4

Тема Outlook и электронная почта

Microsoft Outlook имеет все необходимое для интенсивной работы с электронной почтой. Outlook позволяет принимать, отправлять и пересылать почтовые сообщения, добавлять абонентов в адресную книгу, сортировать полученную информацию.

Напомним, что работа в Интернет и, в частности, работа с электронной почтой связана с такими ключевыми понятиями, как:

- сервер (Специальная программа, расположенная на удаленной машине и предоставляющая свои услуги программам-клиентам. Применительно к организации процесса обмена сообщениями, сервером является удаленная машина, на которой располагается почтовый ящик с Вашими сообщениями.);
- клиент (Специальная программа, которая использует услуги, предоставляемые сервером. В нашем случае такой программой является Outlook, который позволяет просматривать сообщения, пришедшие в почтовый ящик на сервере.);
- протокол (Совокупность правил, определяющих алгоритм передачи данных от сервера к клиенту и наоборот.);
- электронный адрес (уникальный идентификатор, определяющий почтовый ящик, в который приходят сообщения).

Для работы с электронной почтой с помощью Outlook необходимо следующее:

- компьютер, подключенный к локальной сети, имеющей выход в Интернет;
- почтовый ящик в Интернете (или на почтовом сервере локальной сети), который будет использоваться для приема, хранения и отправки корреспонденции;
- для пересылки корреспонденции Outlook должен обладать дополнительными сведениями о почтовом сервере и почтовом ящике пользователя, которые хранятся в учетных записях Outlook.

Выполнить

Зарегистрируйте почтовый ящик в Интернете на сервере mail.ru. Каждый студент, зарегистрировавший почтовый ящик, выходит к классной доске и записывает на ней адрес своего ящика (для возможности обмена почтовыми сообщениями).

Формат электронного адреса (e-mail) имеет следующий вид:
UserName@domain_name,

где

UserName – это имя почтового ящика пользователя, оно должно быть уникальным для домена, который обслуживается сервером в Интернете.

Домен (первого уровня), это некоторое кодовое обозначение длиной 2-3 буквы (.ru -Россия, .fr - Франция и т.п.). Имена доменов следующих уровней имеют большую длину.

domain_name – имя домена для этого пользователя. Имена доменов используются для представления IP-адреса (цифрового адреса) в Интернете. Имена доменов обычно просты для восприятия: mail.ru, microsoft.com, иногда они состоят из нескольких частей mail.admiral.ru.

Для отправки почтовых сообщений на сервер используется протокол Интернета SMTP (Simple Mail Transfer Protocol).

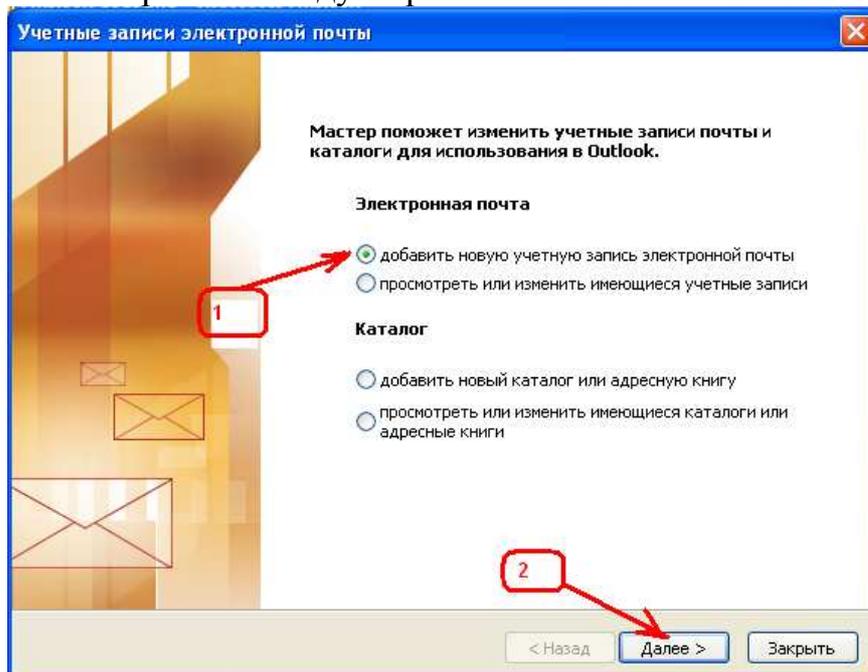
Для приема почтовых сообщений с почтового сервера используется протокол Интернета – POP3 (Post Office Protocol version 3).

Для сервера mail.ru адрес сервера SMTP - smtp.mail.ru, и адрес сервера POP3 – pop.mail.ru.

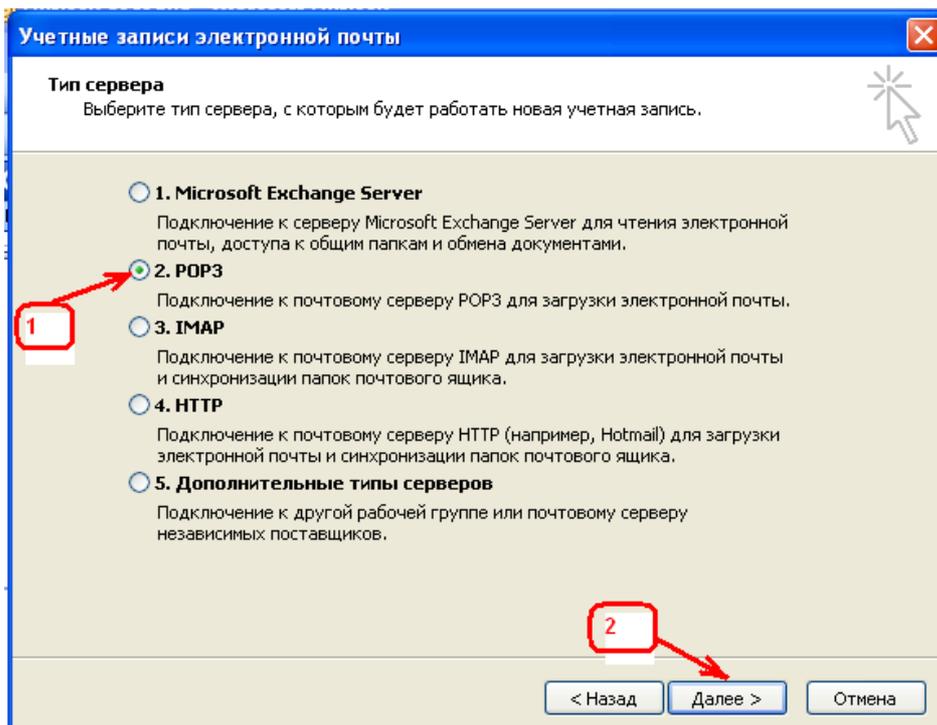
Создайте учетную запись.

Все параметры, связанные с электронной почтой, хранятся в учетных записях Outlook.

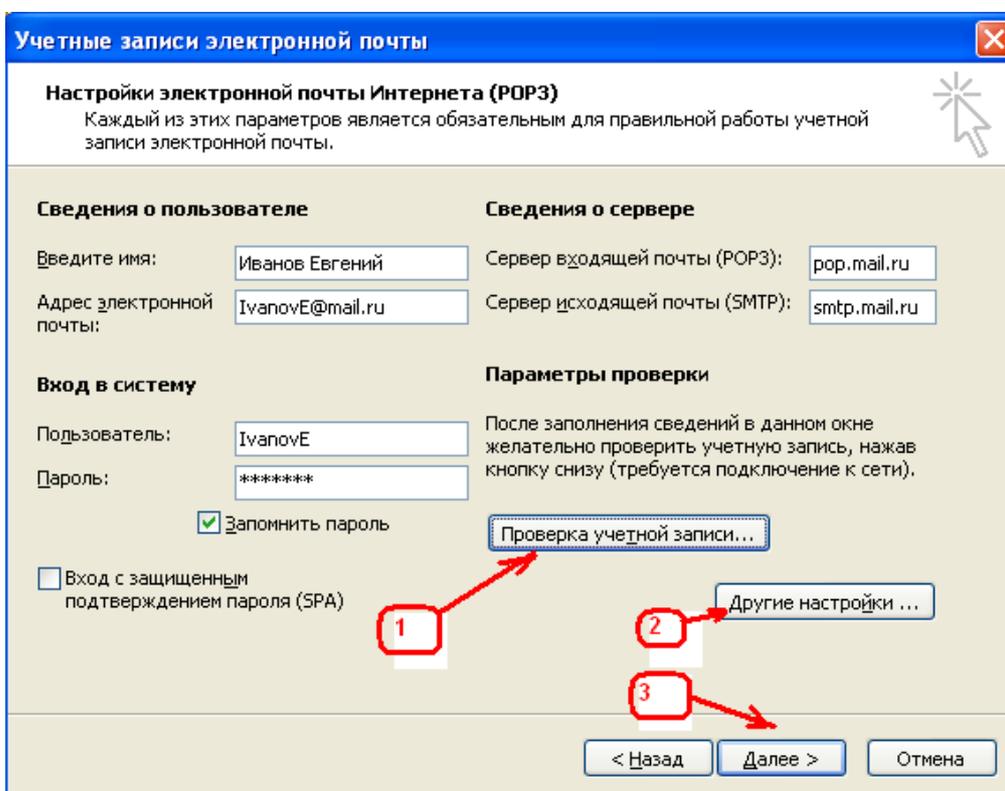
1. Выберите команду Сервис → Учетные записи электронной почты.

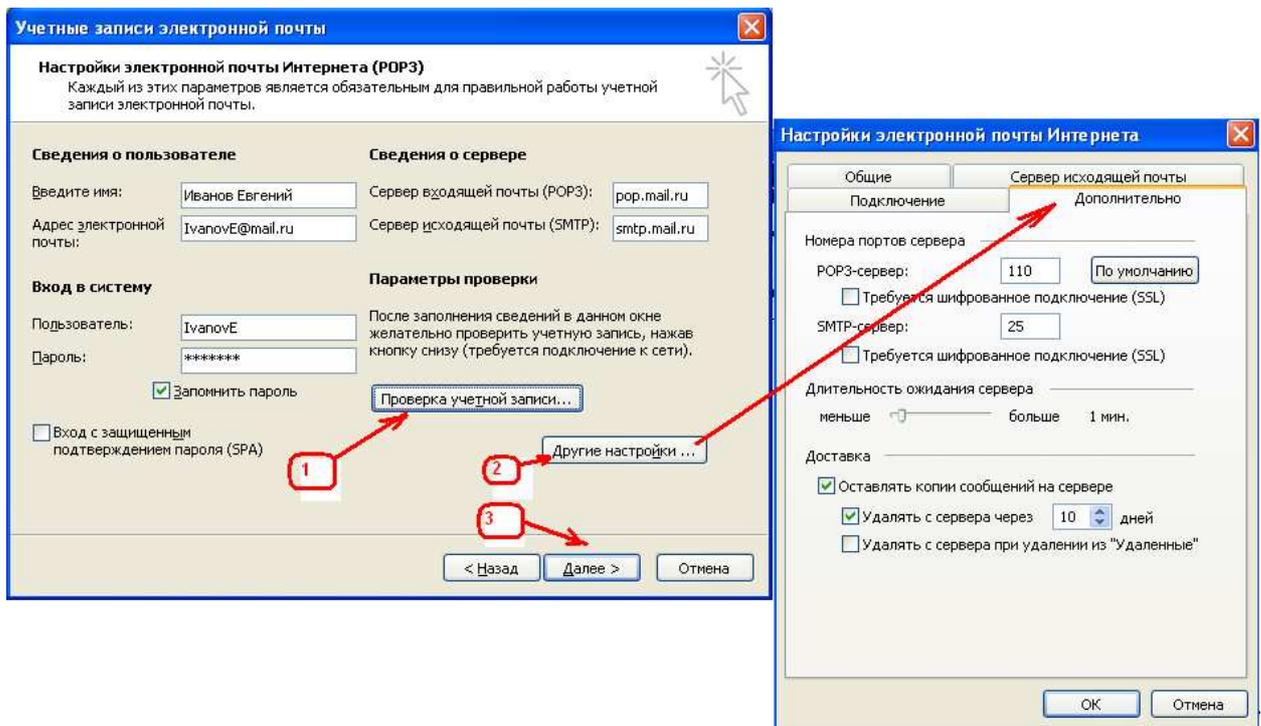


2. Выбор типа сервера, с которым будет работать новая учетная запись

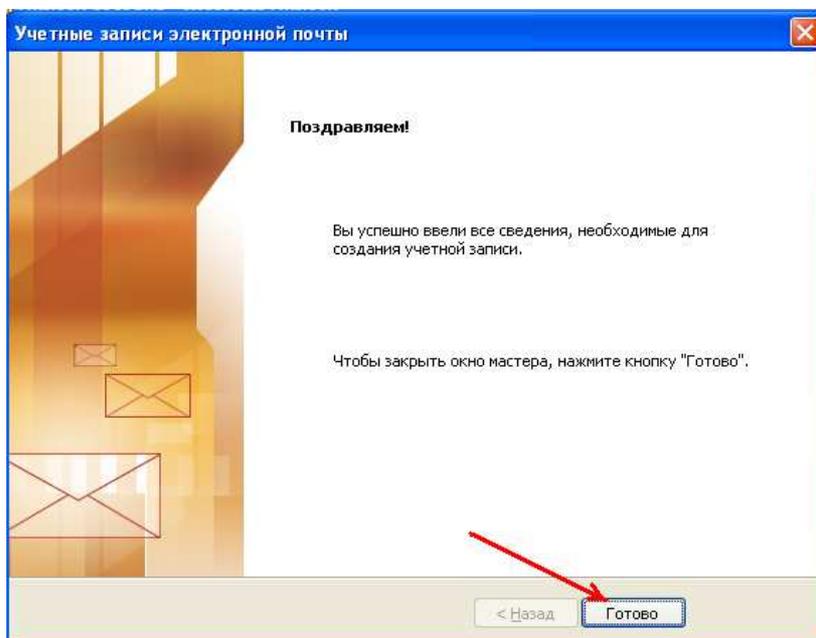


3. Настройки электронной почты Интернета POP3





4. Завершение настройки учетной записи.



После настройки учетной записи обменяйтесь электронными сообщениями с одноклассниками, а также попробуйте разослать сообщения нескольким абонентам, используя «Список рассылки».

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ, МОЛОДЕЖИ И СПОРТА УКРАИНЫ
НАЦИОНАЛЬНАЯ МЕТАЛЛУРГИЧЕСКАЯ АКАДЕМИЯ УКРАИНЫ

Кафедра прикладной математики и вычислительной техники

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРАКТИЧЕСКИМ ЗАНЯТИЯМ ПО ДИСЦИПЛИНЕ

«Электронное документоведение»

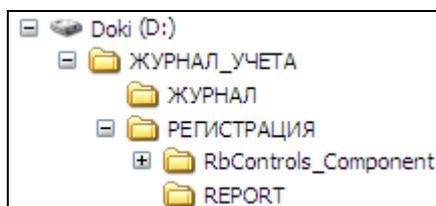
Часть 2

«ЭЛЕКТРОННЫЙ ЖУРНАЛ»

Днепропетровск, 2012

I. ПРИЛОЖЕНИЕ «РЕГИСТРАЦИЯ»

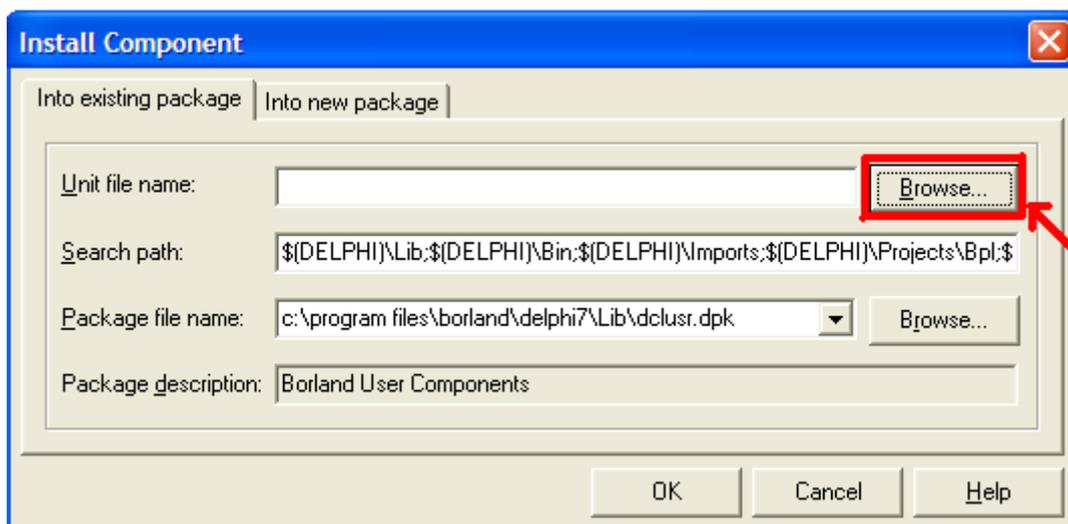
1. Создайте на диске D (Dok) папку **ЖУРНАЛ_УЧЕТА**, а в ней ещё две папки **РЕГИСТРАЦИЯ** и **ЖУРНАЛ**. В папке **РЕГИСТРАЦИЯ** создайте папку **REPORT**.



В папку **ЖУРНАЛ_УЧЕТА** поместите папки **RbControls_Component** и **Comp_DBOLE** (где находятся папки – спросить у преподавателя) – эти папки содержат компоненты, применяющиеся при дизайне приложения. Откройте среду Delphi и установите компоненты папки **RbControls_Component** в соответствии с нижеприведенной схемой (должна добавиться новая страничка **RbControls** в палитре компонентов):

Шаг 1. Пункт меню **Component (Компонент) → InstallComponent**

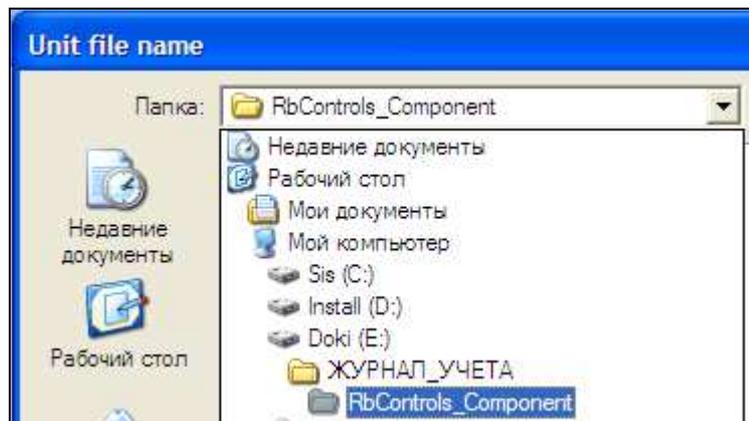
В открывшемся окне нажмите кнопку **Browse...**.



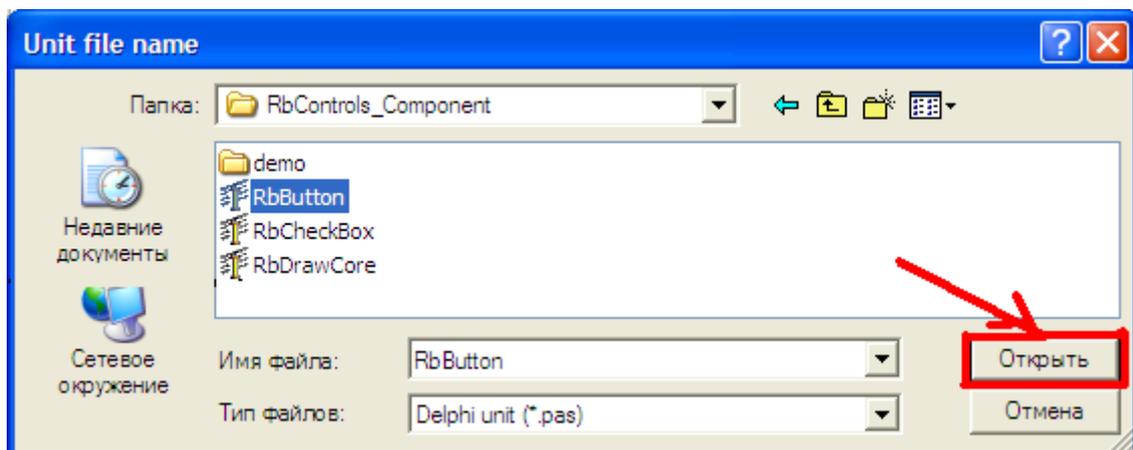
Шаг 2. После нажатия кнопки **Browse...** открывается окно выбора имени подключаемого файла.



В адресном окне (поле **Папка**) указать папку **RbControls_Component**.



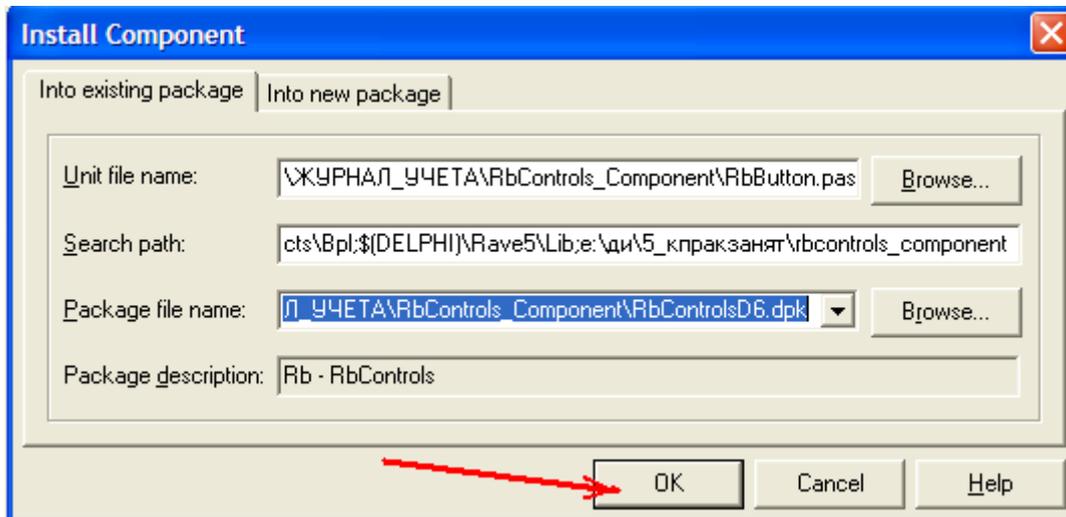
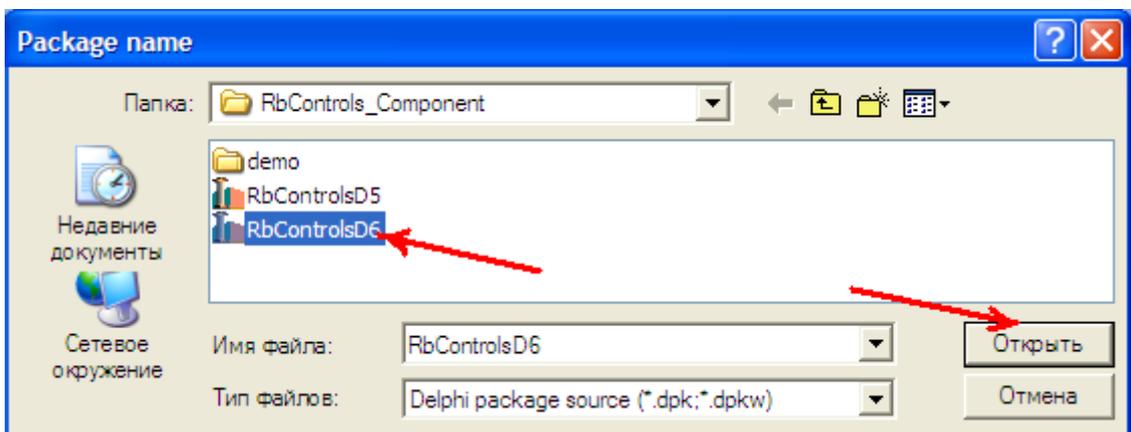
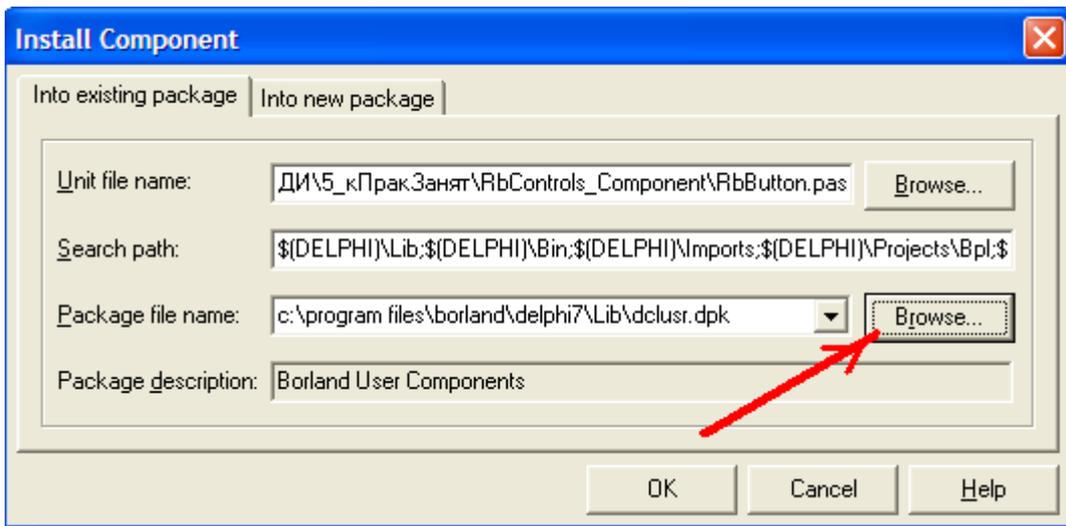
Шаг 3. В папке **RbControls_Component** выбрать любой файл (например RbButton).

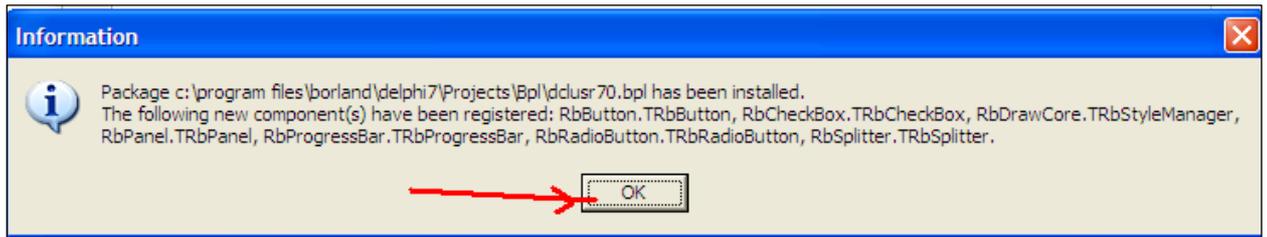
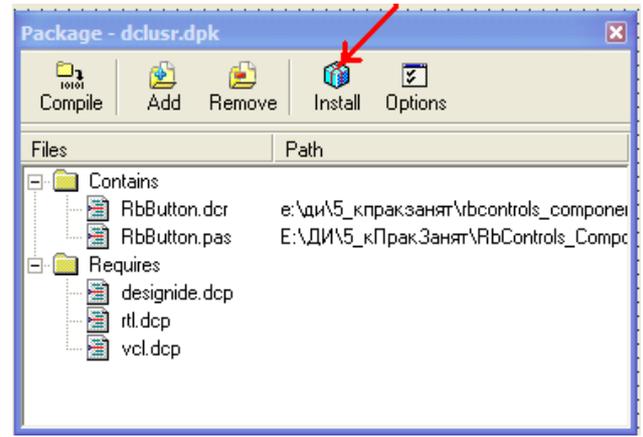
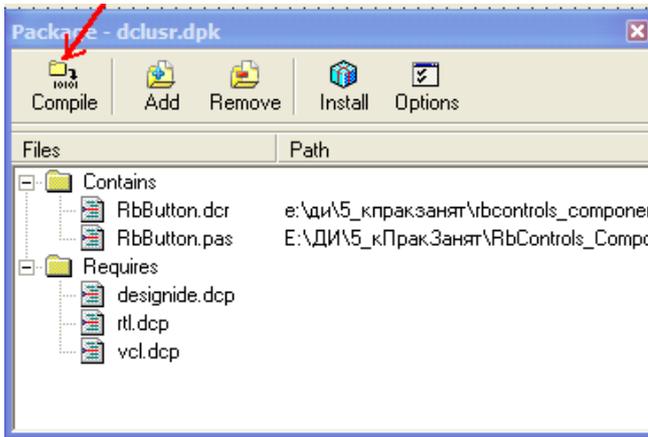


Шаг 4. После нажатия кнопки **Открыть** в окне инсталляции компонента (**Install Component**) в поле **Unit file name** указывается путь нахождения инсталлируемых компонентов.



Далее заполняется поле **Package file name**, последовательность заполнения смотри на рисунках:





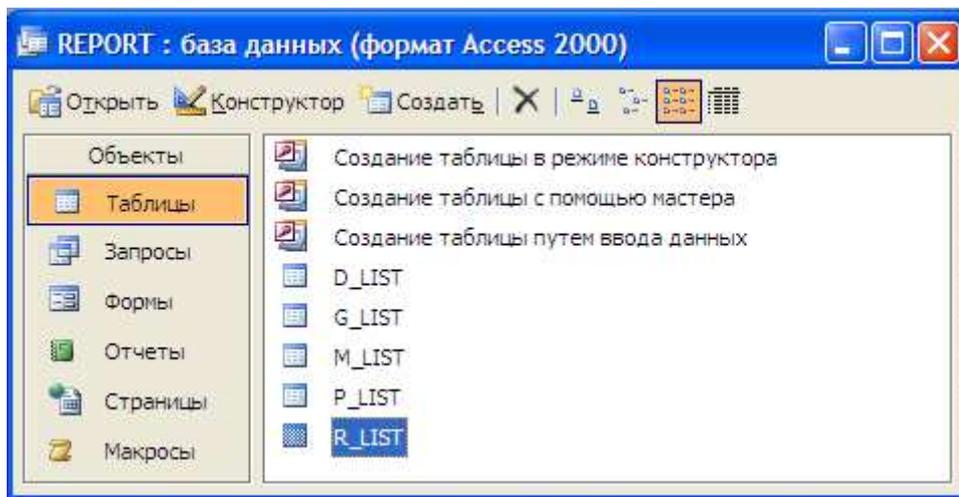
Примечание: если после описанных действий (по подключению компонентов) на экране находятся какие-либо информационные окна их надо просто закрыть. → .



Вид компонентов странички **RbControls** на палитре компонентов:

Аналогичным образом установите компонент **DBOleContainer** () , находящийся в папке **Comp_DBOLE**.

2. Создайте в среде Access базу данных **Report.mdb** (сохраните её в папке **REPORT**), содержащую пять таблиц: **D_LIST**, **G_LIST**, **M_LIST**, **P_LIST**, **R_LIST**, структура которых представлена на рис (рис .).



| D_LIST : таблица | | |
|------------------|----------|------------|
| | Имя поля | Тип данных |
| ? | DISCIP | Текстовый |

| G_LIST : таблица | | |
|------------------|----------|------------|
| | Имя поля | Тип данных |
| ? | PGROUP | Текстовый |

| M_LIST : таблица | | |
|------------------|----------|------------|
| | Имя поля | Тип данных |
| ? | NUM | Счетчик |
| | DISCIP | Текстовый |
| | PGROUP | Текстовый |
| | PNAME | Текстовый |
| | MDAT | Дата/время |
| | TEMA | Текстовый |
| | LEC | Числовой |
| | PRACT | Числовой |
| | LAB | Числовой |
| | KURS | Числовой |
| | KONS | Числовой |
| | PKR | Числовой |
| | KKR | Числовой |
| | DIFZ | Числовой |
| | EXAM | Числовой |
| | DIPL | Числовой |
| | ASPIR | Числовой |
| | METOD | Числовой |
| | POTOK | Текстовый |

| P_LIST : таблица | | |
|------------------|----------|--------------|
| | Имя поля | Тип данных |
| ? | PNAME | Текстовый |
| | JOB | Текстовый |
| | REPORT | Поле объекта |
| | PAS | Числовой |
| | LEK | Числовой |
| | PRACT | Числовой |
| | LAB | Числовой |
| | KURS | Числовой |
| | KONS | Числовой |
| | PKR | Числовой |
| | KKR | Числовой |
| | DIFZ | Числовой |
| | EXAM | Числовой |
| | DIPL | Числовой |
| | ASPIR | Числовой |
| | METOD | Числовой |

| R_LIST : таблица | | |
|------------------|----------|--------------|
| | Имя поля | Тип данных |
| ? | NUM | Счетчик |
| | DISCIP | Текстовый |
| | PGROUP | Текстовый |
| | JOURNAL | Поле объекта |
| | POTOK | Текстовый |

3. В папку **РЕГИСТРАЦИЯ** поместите файл CodeStr , а в паку **ЖУРНАЛ** поместите файлы CodeStr и DBOleContainer (где взять файлы ресурсов CodeStr и DBOleContainer – спросить у преподавателя) .

4. Создание приложения «Регистрация».

1. Откройте среду Delphi, сохраните файлы **Unit1** под именем **Reg**, а **Project1** под именем **RegUser** в папку РЕГИСТРАЦИЯ.

2. Переименуйте главную форму, назначив свойству **Name** → значение **RegForm**, а **Caption** → **Регистрация пользователей**.



3. На главную форму наносим компоненты **ADOTable1** и свойству **Name** → значение **P_TB**, **DataSource1**. Далее устанавливаем компонент **RbPanel1** и свойству **Align** → **alTop**. Затем уже на панель наносим **DBGrid1** и **DBNavigator1**. В соответствии с рис. наносим кнопки:

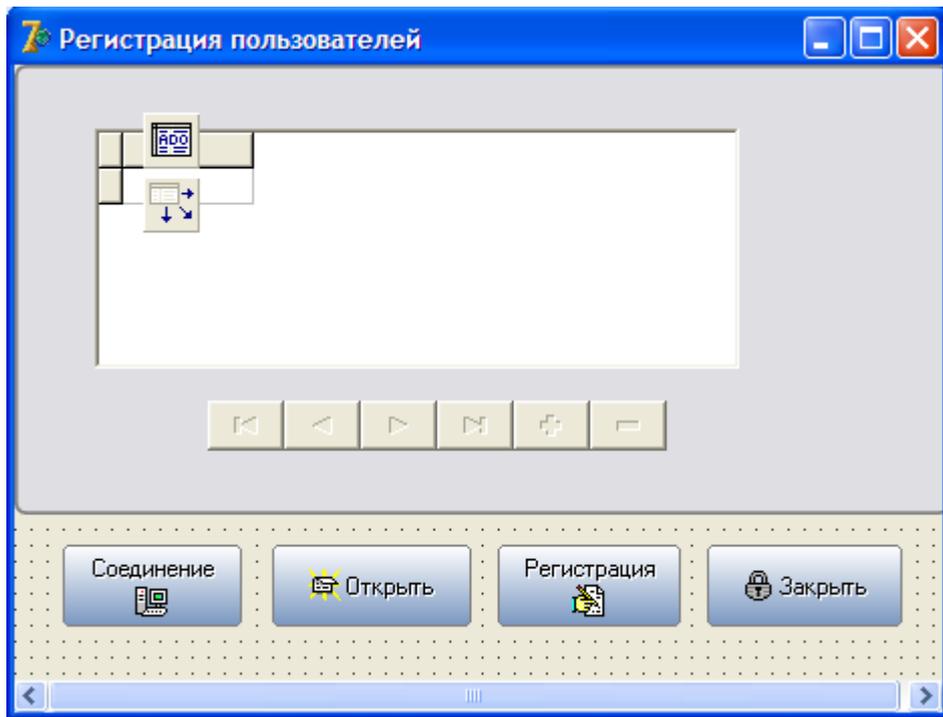
- **RbButton1**:*Name* → **ConectBtn**, *Caption* → **Соединение**, *Cursor* → **crHandPoint**;

- **RbButton2**:*Name* → **OpenBtn**, *Caption* → **Открыть**, *Cursor* → **crHandPoint**;

- **RbButton3**:*Name* → **RegBtn**, *Caption* → **Регистрация**, *Enabled* → **False**, *Cursor* → **crHandPoint**;

- **RbButton4**:*Name* → **CloseBtn**, *Caption* → **Закреть**, *Cursor* → **crHandPoint**.

4. Картинки на кнопки подгружаются через свойство **Glyph**, а их расположение на кнопке – свойство **Layout** (все картинки находятся в папке **Images**, спросить у преподавателя).



5. Скомпилируйте приложение и после его остановки сохраните проект командой File → SaveAll. Сверните среду **Delphi**.

6. Подключение **Link** файла

Для обеспечения диалогового доступа к нашей базе данных (**Report.mdb** → папка **REPORT**), используется специальный файл ***.udl**, он сохраняет условия соединения, что позволяет в дальнейшем повторно не выполнять команду соединения с БД.

Выполните следующие действия.

Скопируйте в папку проекта (папка **РЕГИСТРАЦИЯ**) файл **DBDEMOS.UDL**, находящийся в папке:

Program Files /

Common Files /

System /

Ole DB /

Data Link /

DBDEMOS.UDL

Измените имя файла **DBDEMOS** на **LocalLink** .

Запустите файл **LocalLink.udl** (выполнив двойной щелчок мышью по файлу) и установите соединение с Вашей базой данных, используя кнопку **Browse**.

7. Диалоговое соединение с базой данных

Откройте разрабатываемый проект в **Delphi**.

Установите связь с данными (база данных REPORT.mdb) через компонент **ADOTable (P_TB)**, щелкнув по кнопке [...] свойства **ConnectionString** - в открывшемся окне (рис. 8) нажмите кнопку **Browse** (опция **User Data Link File**). В диалоговом окне выберите файл **LocalLink.udl**, находящийся в папке проекта (папка **РЕГИСТРАЦИЯ**).

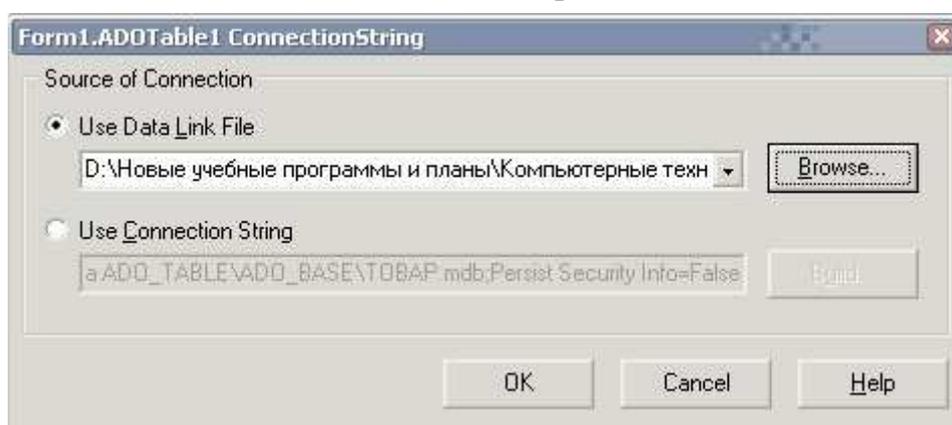
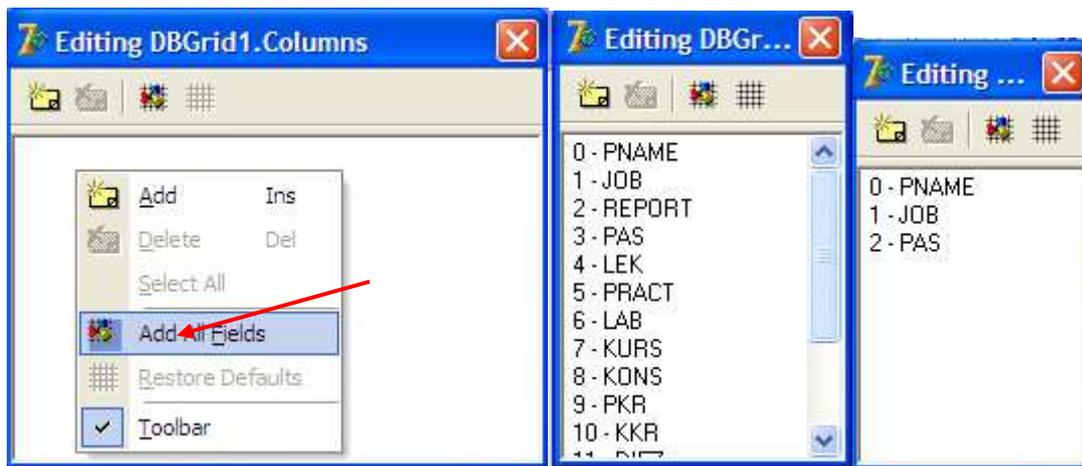


Рис. 8 Выбор соединения

8. У компонента **ADOTable (P_TB)** для свойства **TableName** → **P_LIST**, **Active** → **true**.

9. Свяжите между собой компоненты **ADOTable (P_TB)**, **DataSource1**, **DBGrid1** и **DBNavigator1**.

10. Выполните двойной щелчок мышкой по компоненту **DBGrid1** и вызвав контекстное меню на открывшейся форме командой **AddAllFields** добавьте все поля таблицы **P_LIST** в **DBGrid1**, а затем удалите отображение всех полей из **DBGrid1**, оставив только поля **PNAME**, **JOB** и **PAS**.



Выполните двойной щелчок по компоненту **ADOTable (P_TB)** и аналогично добавьте к нему все поля таблицы (удалять ничего не надо!). Внешний вид приложения показан на рис.

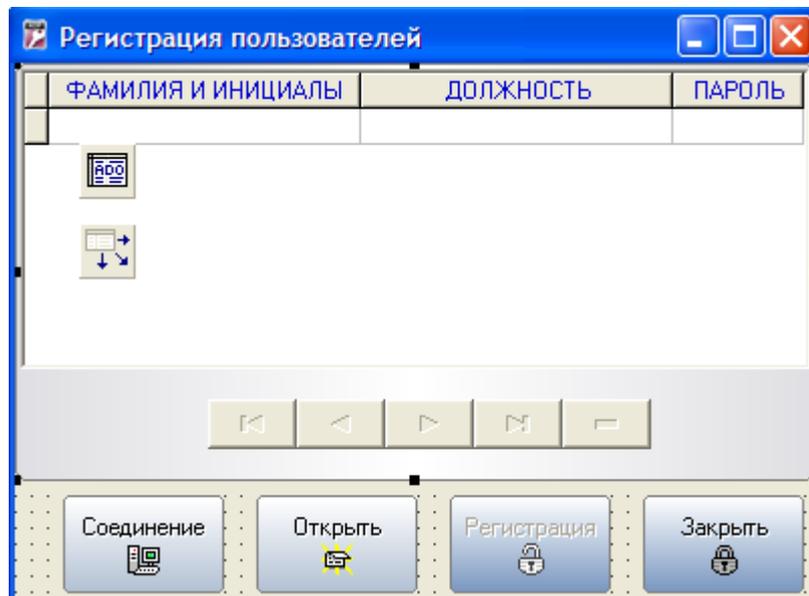


Рис. Внешний вид приложения

Скомпилируйте приложение и после его остановки сохраните проект командой **File → SaveAll**.

11. В раздел **USES** интерфейсной части **Unit'a** добавьте названия используемых в дальнейшем модулей **shellapi** и **CodeStr**.

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, DBCtrls, ExtCtrls, Grids, DBGrids, Db, ADODB, **shellapi**, **CodeStr**;

В разделе **var** самостоятельно объявите переменную **LocalPath : String**.

var

RegForm: TRegForm;
LocalPath: **string**;

12. Значение переменной **LocalPath** определите в обработчике события **OnCreate** формы.

```
procedure TRegForm.FormCreate(Sender: TObject);  
  
begin  
    P_TB.Close;  
    LocalPath := ExtractFilePath(application.ExeName);  
    P_TB.ConnectionString := 'FILE NAME=' + LocalPath + 'LocalLink.udl';  
end;
```

13. В обработчике события **OnClick** кнопки **Соединение** вызовите **ShellApi** функцию **ShellExecute**:

```
procedure TRegForm.ConnectBtnClick(Sender: TObject);  
begin  
    P_TB.Close;  
    LocalPath := ExtractFilePath(application.ExeName);  
    ShellExecute(RegForm.Handle, 'open', PCHAR(LocalPath + 'LocalLink.udl'), nil, '', SW_SHOW);  
end;
```

13. Скомпилируйте приложение, проверьте работу кнопки **Соединение** и остановив приложение до сохранения проекта (при отсутствии ошибок) командой **File → SaveAll**.

14. Для кнопки **Открыть (OpenBtn)** для следующих свойств установите такие значения:

- AllowAllUp → true;
- GroupIndex → 1.

В обработчике события **OnClick** кнопки **Открыть** напишите соответствующий метод:

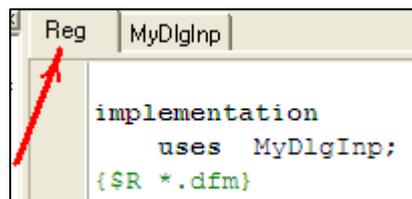
```
procedure TRegForm.OpenBtnClick(Sender: TObject);  
begin  
    P_TB.Close;  
    if OpenBtn.Down then  
    begin  
        P_TB.ConnectionString := 'FILE NAME=' + LocalPath + 'LocalLink.udl';  
        P_TB.Open;  
    end;  
    RegBtn.Enabled := OpenBtn.Down;  
end;
```

15. Скомпилируйте приложение, проверьте работу кнопки **Открыть** и после его остановки до сохранения проекта командой **File → SaveAll**.

Регистрация пользователей (в нашем случае преподавателей) выполняется в отдельной форме, для этого подключим к проекту ещё одну форму.

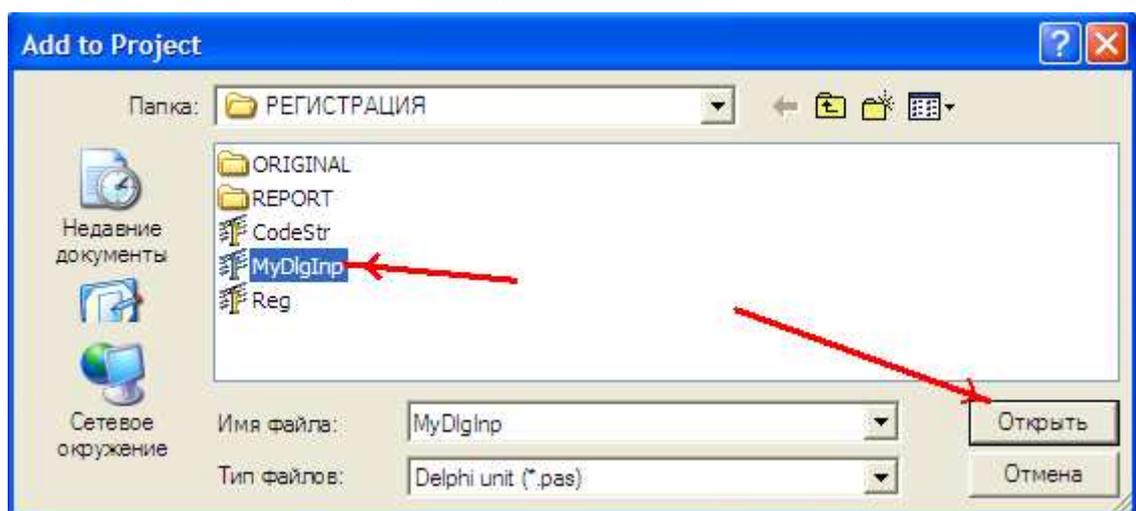
16. Выполните меню **File**→**New**→**Form**. Для новой формы установите значение свойства **Name**→ *MyDialogInp*, а **Caption** → **ОКНО РЕГИСТРАЦИИ** и сохраните её в папку проекта (папка **РЕГИСТРАЦИЯ**) командой **File**→**SaveAs** , переименовав **Unit1** → *MyDlgInp*. Подключите новую форму к проекту:

- пропишите в **unit**-е **Reg** секции **implementation** нового пользователя



```
implementation
uses
  MyDlgInp;
{$R *.dfm}
```

- добавьте форму к проекту через меню **Project**→**Add to Project**



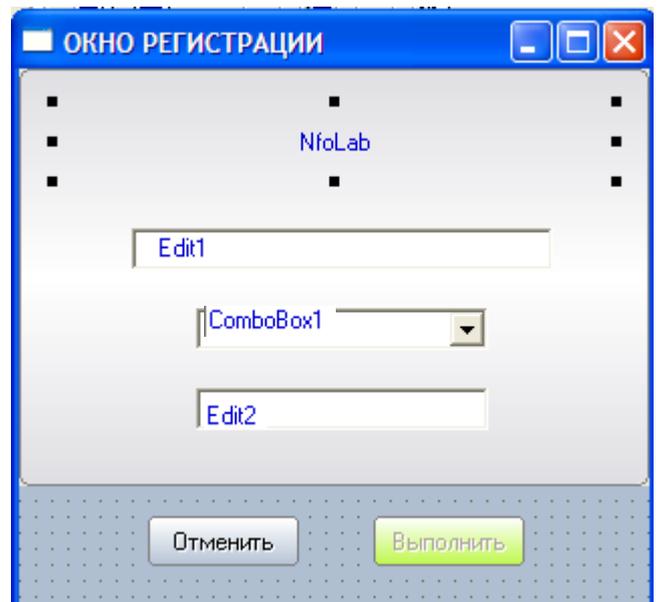
17. Дизайн формы «ОКНО РЕГИСТРАЦИИ»

Установите на форму объекты:

- **RbButton1**: Name → *CancelBtn*,
Caption

→ **Отменить**,

ModalResult→*mrCancel*, **Cursor**→
crHandPoint;



- **RbButton2**: Name → OKBtn, Caption → Выполнить, ModalResult → mrOk, Enabled → False, Cursor → crHandPoint.

- **RbPanel1** (свойство Align → alTop) и на нее нанесите компоненты:

а) **Lable1**: Name → NfoLab, Caption → NfoLab, AutoSize → False, Aligment → taCenter, WordWrap → True;

б) **Edit1**;

в) **ComboBox**: Items → Профессор

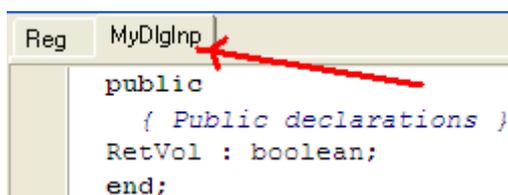
Доцент

Ст.преподаватель

Ассистент;

г) **Edit2**: Name → PasEdit, PasswordChar → * (знак умножения).

18. Для осуществления процедуры регистрации пользователя вводится логическая переменная **RetVol**, ее необходимо объявить в **unit-е MyDlgInp**



19. В обработчике событий **FormActivate** формы **MyDialogInp** присвойте переменной **RetVol** значение **ЛОЖЬ**.

```
procedure TMyDialogInp.FormActivate(Sender: TObject);
begin
  RetVol := false;
end;
```

20. Для кнопки **Выполнить** запишите следующую процедуру:

```
procedure TMyDialogInp.OKBtnClick(Sender: TObject);
begin
  RetVol := true;
end;
```

21. Процедура для обработчика событий **Edit1Change** компонента **Edit1**

```
procedure TMyDialogInp.Edit1Change(Sender: TObject);
begin
  If Edit1.Text <> '' then OKBtn.Enabled := true else OKBtn.Enabled := false;
end;
```

22. В обработчике события **OnClick** кнопки **Регистрация (RegBtn)** – главная форма - напишите соответствующий метод:

```

procedure TRegForm.RegBtnClick(Sender: TObject);
begin
  MyDialogInp.ComboBox1.Visible := true;
  MyDialogInp.NfoLab.Font.Color := clBlack;
  MyDialogInp.Edit1.Text := '';
  MyDialogInp.PasEdit.Text := '';
  MyDialogInp.NfoLab.Caption := 'ВВЕДИТЕ ФАМИЛИЮ, ИНИЦИАЛЫ, ДОЛЖНОСТЬ И ПАРОЛЬ';
  MyDialogInp.ShowModal;           // Вызов диалогового окна
  if MyDialogInp.RetVol = true then // Если флаг действительный, то:
  begin
    P_TB.Append;                    // Добавляем запись
    P_TBNAME.Value := MyDialogInp.Edit1.Text; // Присваиваем значение поля
    P_TBJOB.Value := MyDialogInp.ComboBox1.Text;
    P_TBPAS.Value := CRC32(-1, trim(MyDialogInp.PasEdit.Text));
    P_TB.Post;                      // Сохраняем запись
    P_TB.Refresh;                   // Обновляем DataSet
  end;
end;

```

В обработчике события OnClick кнопки **Закреть (CloseBtn)** – главная форма - напишите соответствующий метод:

```

procedure TRegForm.CloseBtnClick(Sender: TObject);
begin
  Close;
end;

```

23. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll** .

II. ПРИЛОЖЕНИЕ «РЕГИСТРАЦИЯ»

Первый этап создания проекта (приложение «РЕГИСТРАЦИЯ») завершен. Следующий этап – создание приложения «ЖУРНАЛ УЧЕТА». Для этого:

1. Откройте новое приложение **File→New→Application** и сохраните проект в папку «Журнал», переименовав при этом: **Unit1 → Main, Project1 → Report**.

2. Присвойте свойствам формы **Form1** следующие значения: **Name → MainForm; Caption → СЭД «Индивидуальная работа преподавателя и ее учет»**.

Далее на форму наносятся пять панелей (компоненты класса **RbPanel**), каждая из которых будет отвечать определенным логически связанным функциям.

- Первая панель (*контрольная* – **CtrlPanel**), содержит кнопки доступа и вывода отчетов.

- Вторая панель (**PN1**) содержит сетки (DBGrid-ы) с информацией о дисциплинах и преподавателях.

- Третья панель (**PN2**) содержит компоненты, формирующие составные пункты в *журнале отчетности преподавателя*.

- Четвертая панель (**PN3**) содержит *сводную информацию* из верхних таблиц.

- Пятая панель (*заставочная* – **PN4**) играет роль заставки при запуске приложения (она закрывает собой все панели, кроме *контрольной*), после правильного доступа в систему заставочная панель скрывается, открывая доступ пользователю к работе.

Предупреждение!!! Будьте внимательны при установке панелей (компоненты **RbPanel**) и размещении на них дочерних компонентов. Строго следите за соответствием Вашего дизайна приведенным ниже рисункам с расположением компонентов. Путаница в расположении компонентов приведет к неправильной работе приложения или к его «зависанию».

Внимание!!! Во избежание путаницы, вначале наносите все компоненты в соответствии с рисунком, а затем устанавливайте для их свойств соответствующие значения!!! Придерживайтесь этой рекомендации на протяжении создания каждой из пяти панелей!!!

Формирование первой (контрольной) панели.

3. Нанесите на форму компонент **RbPanel1** и измените значения таких свойств: *Name* → **CtrlPanel**, *Align* → **alTop**, *BorderColor* → **clNavy**, *BorderWidth* → **5**, *Antialiased* → **False**

4. На саму *контрольную* панель (**CtrlPanel**) установите ещё две панели:

- **RbPanel2**: *Name* → **RepPanel**, *Align* → **alRight** .

- **RbPanel3** *Name* → **SelPanel**, *Align* → **alRight** .



5. Далее в соответствии с рисунком на соответствующие панели нанесите компоненты **RbButton1...7** и установите для их свойств такие значения:

- **RbButton1**: *Name* → **ALLREPBTN**, *Caption* → **ОТЧЕТ №3**, *Cursor* → **crHandPoint**, *Hint* → **Сводный отчет**, *ShowHint* → **True**.

- **RbButton2**: *Name* → **SelPerBtn**, *Caption* → **ОТЧЕТ №2**, *Cursor* → **crHandPoint**, *Hint* → **Отчет по дисциплинам**, *ShowHint* → **True**, *Enabled* → **False**.

- **RbButton3**: *Name* → **SelRepBtn**, *Caption* → **ОТЧЕТ №1**, *Cursor* → **crHandPoint**, *Hint* → **Отчет по периодам**, *ShowHint* → **True**, *Enabled* → **False**.

- **RbButton4**: *Name* → **FLTRBTN**, *Caption* → **Выбрать**, *Cursor* → **crHandPoint**, *Hint* → **Выбрать период**, *ShowHint* → **True**, *AllowAllUp* → **True**, *GroupIndex* → **1**, *Glyph* → **подгрузить соответствующую картинку**.

- **RbButton5**: *Name* → **ConectBtn**, *Caption* → **Соединение**, *Cursor* → **crHandPoint**, *Hint* → **Соединение с базой данных**, *ShowHint* → **True**, *Glyph* → **подгрузить соответствующую картинку**, *Layout* → **blGlyphBottom**.

- **RbButton6**: *Name* → **OpenBtn**, *Caption* → **Доступ**, *Cursor* → **crHandPoint**, *Hint* → **Доступ к базе данных**, *ShowHint* → **True**, *AllowAllUp* → **True**, *GroupIndex* → **1**. **Подгрузить соответствующую картинку**.

- **RbButton7**: *Name* → **UREPBTN**, *Caption* → **ОТЧЕТЫ**, *Cursor* → **crHandPoint**, *Hint* → **Доступ к формированию отчетов**, *ShowHint* → **True**, *AllowAllUp* → **True**, *GroupIndex* → **1**, *Enabled* → **False**. **Подгрузить соответствующую картинку**.



6. Добавьте на панель **RepPanel** (бывший компонент RbPanel2) такие компоненты (SpinEdit - палитра компонентов Samples):

- **SpinEdit1** : Name → BNSpinEdit, MaxValue→2050, MinValue→2010, Value → 2012;
- **SpinEdit2** : Name → ENSpinEdit, MaxValue→2050, MinValue→2010, Value → 2013;
- **Label1**: Caption → Учебный период, WordWrap → True, Aligment → taCenter.

7. Добавьте на панель **SelPanel (RbPanel3)** два компонента (палитра компонентов - Win32)

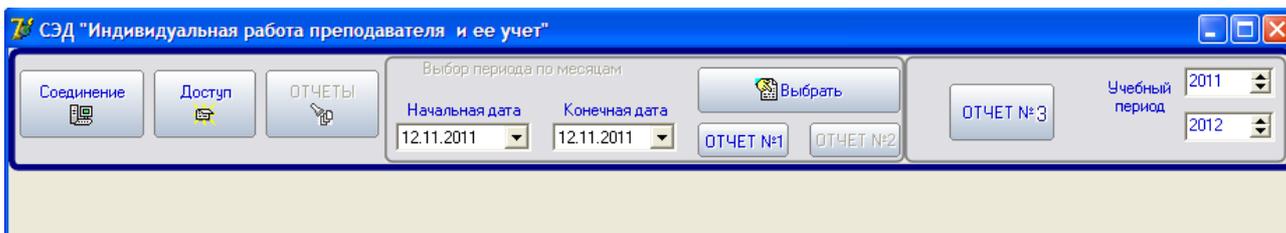
DateTimePicker и два компонента Label.



- для компонента DateTimePicker1: **Name** → BeginDTP, - выбор начальной даты отчета;
- для компонента DateTimePicker2: **Name** → EndDTP, - выбор конечной даты отчета;
- для компонента Label2: **Caption** → Начальная дата;
- для компонента Label3: **Caption** → Конечная дата.

8. Добавьте следующие изменения для компонентов:

- SelPanel (RbPanel3): **Caption** → Выбор периода по месяцам; **Enabled** → False;
- RepPanel (RbPanel2): **Enabled** → False.



Установите на главную форму (MainForm) компонент **ScrollBar1** и выполните для его свойства **Align** → alClient. Все остальные панели будут устанавливаться в компонент **ScrollBar1**.

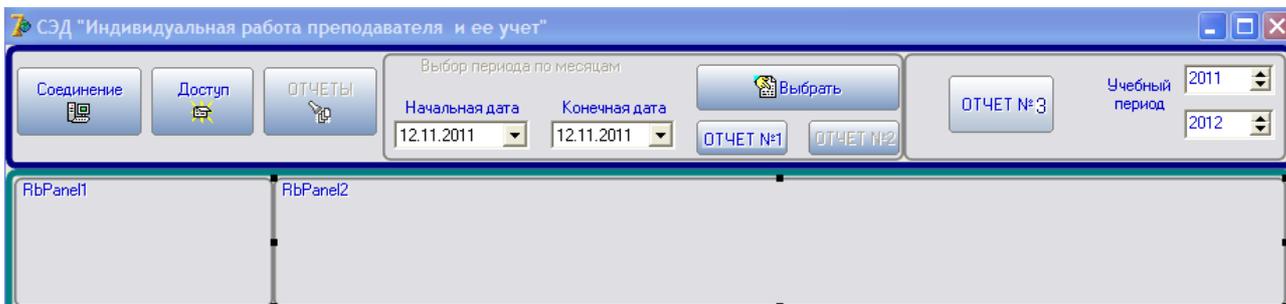
Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll**.

Формирование второй панели.

9. Установите в компонент **ScrollBar1** компонент **RbPanel** и задайте его свойствам следующие значения:

- **Name** → PN1, **Align** → alTop, **BorderColor** → clTeal, **BorderWidth** → 5, **Enabled** → False.

10. В компонент панель PN1 поместите еще два компонента **RbPanel1** и **RbPanel2**.

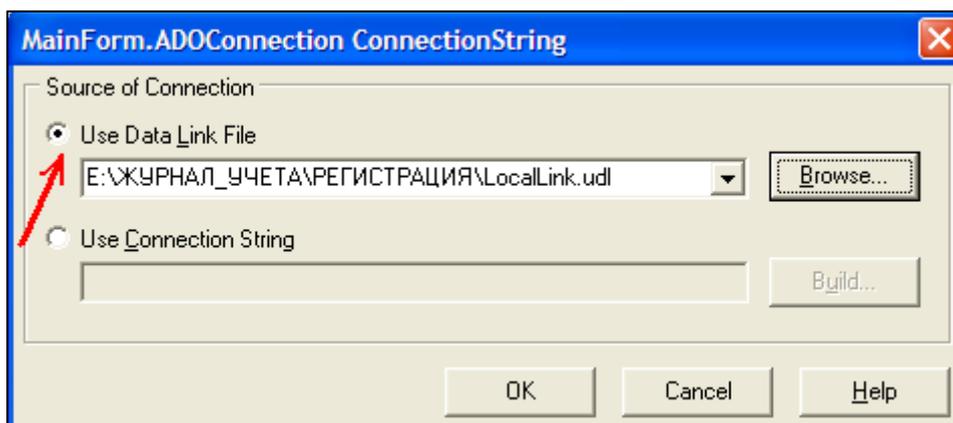


Для компонента RbPanel1: **Align** → alLeft, **Antialiased** → False;

RbPanel2 → **Align** → alClient, **Antialiased** → False.

11. Установите на форму (ИЛИ НА ПАНЕЛИ) компоненты:

ADOConnection1: - **ConnectionString** → выберите файл



LocalLink.udl, находящийся в папке проекта (папка **РЕГИСТРАЦИЯ**).

- **Name** → ADOConnection;
- **Provider** → Microsoft.Jet.OLEDB.4.0;
- **LoginPromt** → False;
- **Mode** → cmShareDenyNone.

*{Скомпилируйте приложение, проверьте его работу и после его остановки
досохраните проект командой **File** → **SaveAll** .}*

- ADOTable1 :
- **Name** → D_LIST_TB,
 - **Connection** → ADOConnection,
 - **TableName** → D_LIST
 - **Active** → True



- ADOTable2 :
- **Name** → P_LIST_TB,
 - **Connection** → ADOConnection,
 - **TableName** → P_LIST
 - **Active** → True

DataSource1: **Name** → D_LIST_DS; **DataSet** → D_LIST_TB

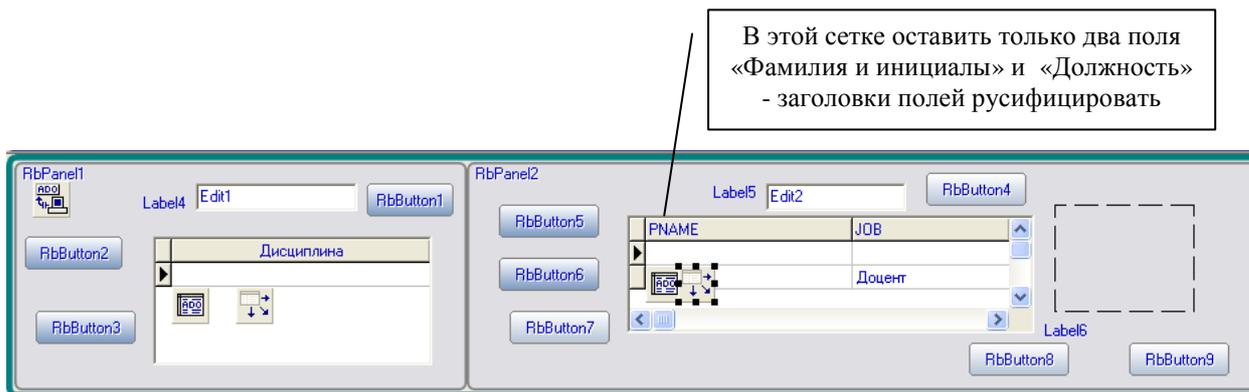
DataSource2: **Name** → P_LIST_DS ; **DataSet** → P_LIST_TB.

*{Скомпилируйте приложение, проверьте его работу и после его остановки
досохраните проект командой **File** → **SaveAll** .}*

12. В компонент **RbPanel1** установите такие компоненты:

- DBGrid1 : **Name** → D_LIST_Grid1; **DataSource** → D_LIST_DS,

Options: **dgEditing** → False, **dgRowSelect** → True.



- RbButton1 : **Name** → FLT_D, **Caption** → F, **AllowAllUp** → True, **GroupIndex** → 1, **Glyph** → подгрузить соответствующую картинку, **Hint** → Установить фильтр, **ShowHint** → True, **Cursor** → crHandPoint.

- RbButton2 : **Name** → ADD_D, **Glyph** → подгрузить соответствующую картинку, **Hint** → Создать запись, **ShowHint** → True, **Cursor** → crHandPoint.

- RbButton3 : **Name** → DEL_D, **Glyph** → подгрузить соответствующую картинку, **Hint** → Удалить запись, **ShowHint** → True, **Cursor** → crHandPoint.

- Edit1;

- Label4: **Caption** → Быстрый поиск.

13. В компонент **RbPanel2** установите такие компоненты:

- **DBGrid2** : **Name** → P_LIST_Grid, **DataSource** → P_LIST_DS, 
Options: **dgEditing** → False, **dgRowSelect** → True.

- **RbButton4** : **Name** → FLT_P, **Caption** → F, **AllowAllUp** → True, **GroupIndex** → 1, **Glyph** → подгрузить соответствующую картинку, **Hint** → Установить фильтр, **ShowHint** → True, **Cursor** → crHandPoint.

- **RbButton5** : **Name** → EDIT_P, **Glyph** → подгрузить соответствующую картинку, **Hint** → Годовой план, **ShowHint** → True, **Cursor** → crHandPoint.

- **RbButton6** : **Name** → ADD_P, **Glyph** → подгрузить соответствующую картинку, **Hint** → Создать запись, **ShowHint** → True, **Cursor** → crHandPoint.

- **RbButton7** : **Name** → DEL_P, **Glyph** → подгрузить соответствующую картинку, **Hint** → Удалить запись, **ShowHint** → True, **Cursor** → crHandPoint.

- **RbButton8** : **Name** → **INSCont2**, **Caption** → **IN**, **Glyph** → подгрузить соответствующую картинку, **Hint** → Отчет преподавателя, **ShowHint** → True , **Cursor** → **crHandPoint**.

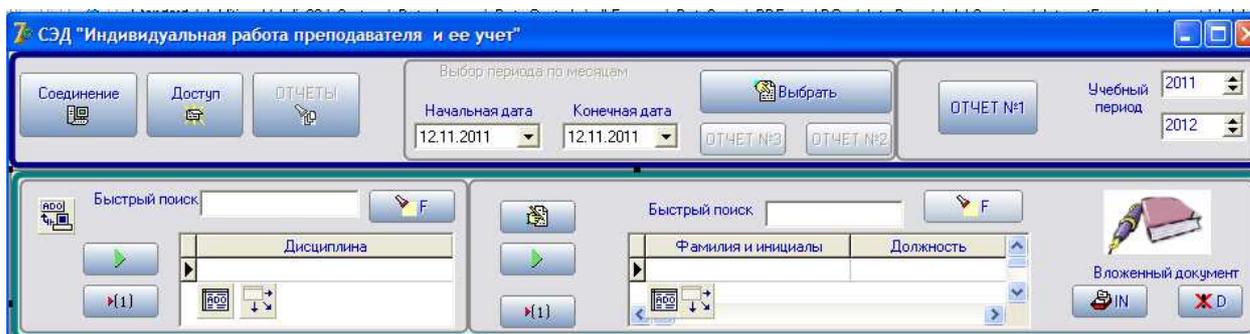
- **RbButton9** : **Name** → **DelCont2**, **Caption** → **D**, **Glyph** → , **Hint** → Удалить документ, **ShowHint** → True , **Cursor** → **crHandPoint**.

- **Edit2**, **Name** → **EditF**;

- **Label5**: **Caption** → Быстрый поиск.

- **Label6**: **Caption** → Вложенный документ.

- **Image1**: **Picture** → подгрузить соответствующую картинку.



13. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll** .

Формирование третьей панели.

15. Установите в компонент **ScrollBar1** компонент **RbPanel** (**RbPanel3**), а в него поместите еще два компонента **RbPanel** (**RbPanel4** и **RbPanel5**).

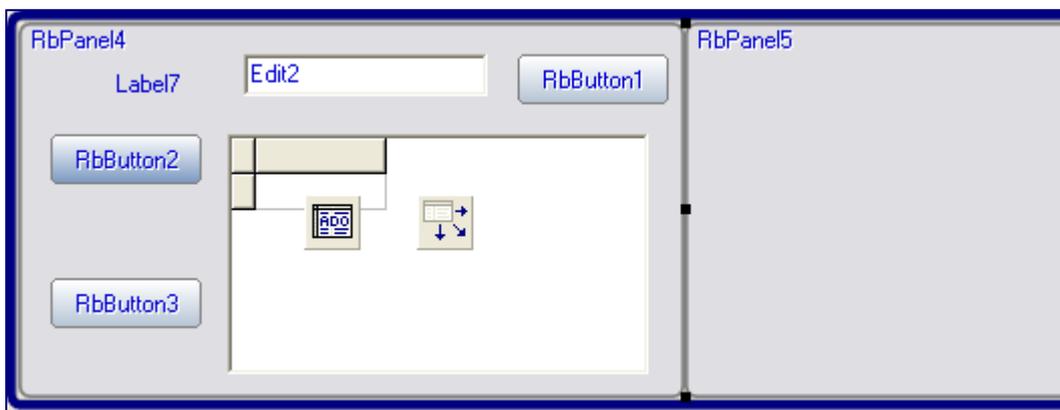
Задайте свойствам компонентов **RbPanel3**, **RbPanel4** и **RbPanel5** следующие значения:

- **Name** → **PN2**, **Align** → **alTop**, **BorderColor**→ **clNavy**, **BorderWidth**→**5**, **Enabled** → **False**.

- **RbPanel4**: **Align** → **alLeft** , **Antialiased** → **False**;

- **RbPanel5** : **Align** → **alClient**. **Antialiased** → **False**.

16. На панель **RbPanel4** нанесите компоненты в соответствии с



рисунком:

17. **ADOTable1** : Name → G_LIST_TB, Connection → ADOConnection, TableName → G_LIST, Active → True

- **DataSource1**: Name → G_LIST_DS; DataSet → G_LIST_TB

- **DBGrid1** : Name → G_LIST_Grid, DataSource → G_LIST_DS, 

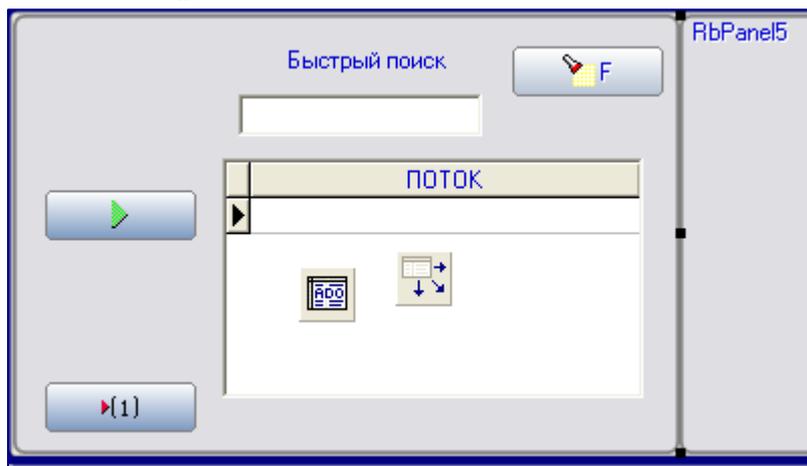
Options: dgEditing → False, dgRowSelect → True.

18. Выполните следующие действия и приведите панель **RbPanel4** в

соответствие с ниже

представленным рисунком.

- **RbButton1** : Name → FLT_G, Caption → F, AllowAllUp → True, GroupIndex → 1, Glyph → подгрузить соответствующую картинку, Hint



→ Установить фильтр, **ShowHint** → True, **Cursor** → crHandPoint.

- **RbButton2** : Name → ADD_G, Glyph → подгрузить соответствующую картинку, Hint → Создать запись, **ShowHint** → True, **Cursor** → crHandPoint.

- **RbButton3** : Name → DEL_G, Glyph → подгрузить соответствующую картинку, Hint → Удалить запись, **ShowHint** → True, **Cursor** → crHandPoint.

- **Edit2**;

- **Label7**: **Caption** → Быстрый поиск.

Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll** .

19. На панель **RbPanel5** нанесите компоненты в соответствии с рисунком:



20. Перечисленным ниже компонентам установите следующие значения для их свойств:

- **ADOTable1** : Name → R_LIST_TB, Connection → ADOConnection, TableName → R_LIST, Active → True;

- **DataSource1**: Name → R_LIST_DS; DataSet → R_LIST_TB

- **DBGrid1** : Name → R_LIST_Grid, DataSource → R_LIST_DS, Options: dgEditing → False, dgRowSelect → True.



21. Выполните двойной клик мышкой по сетке (**DBGrid - R_LIST_Grid**). В открывшемся диалоговом окне вызовите контекстное меню и добавьте все поля к сетке. Из всех добавленных полей оставьте PGROUP и DISCIP, поменяйте поля местами обычным перетаскиванием мышкой.

22. Русифицируйте заголовки полей в таблицах: **PGROUP** : **Caption** → ПОТОК/ГРУППА ; **DISCIP**: **Caption** → ДИСЦИПЛИНА.

23. Для указанных компонентов установите такие значения их свойствам:

- **RbButton1** : Name → ADD_R, Glyph → загрузить соответствующую картинку, Hint → Добавить поток, ShowHint → True , Cursor → crHandPoint.

- **RbButton2** : Name → ADD_R_G, Glyph → загрузить соответствующую картинку, Hint → Добавить группу в поток, ShowHint → True , Cursor → crHandPoint.

- **RbButton3** : Name → DEL_R, Glyph → загрузить соответствующую картинку, Hint → Удалить запись, ShowHint → True , Cursor → crHandPoint.

- **RbButton4** : Name → INSCont1, Caption → IN, Glyph → подгрузить соответствующую картинку, Hint → Журнал группы / Программа дисциплины, ShowHint → True , Cursor → crHandPoint.

- **RbButton5** : Name → DelCont1, Caption → D, Glyph → подгрузить соответствующую картинку, Hint → Удалить документ, ShowHint → True , Cursor → crHandPoint.

- **RbButton6** : Name → ADD_M1, Caption → Записать, Glyph → подгрузить соответствующую картинку, Layout → blGlyphBottom, Hint → Добавить мероприятие в отчет, ShowHint → True , Cursor → crHandPoint.

- **Label8** : Caption → ОТНОШЕНИЕ ПОТОК: ГРУППА, ДИСЦИПЛИНА (Установить фильтры ПОТОК: ГРУППА-ДИСЦИПЛИНА)

- **Label9**: Caption → Вложенный документ.

- **Edit3** : Name → ТЕМАEdit (все символы набраны в режиме EN языка!)

- **Label10** : Caption → Тема занятий.

- **Image2**: Picture → подгрузить соответствующую картинку.

- **Label11** : Caption → Кол-во часов.

- **Label12** : Caption → Дата проведения.

- **SpinEdit1** : Name → ClockEdit , MaxValue→100, MinValue→1, Value → 2.

- **DateTimePicker1**,

- **RbRadioButton1** : Name → LecRb, Caption → Лекционные занятия.

- **RbRadioButton2** : Name → PractRb, Caption → Практические занятия

- **RbRadioButton3** : Name → LabRb, Caption → Лабораторные работы

- **RbRadioButton4** : Name → KursRb , Caption → Курсовое проектиров.

- **RbRadioButton5** : Name → PkrRB, Caption → Проверка контр. работ

- **RbRadioButton6** : Name → KkrRB, Caption → Комплексные КР

- **RbRadioButton7** : Name → DifzRB, Caption → Проведение диф. зачета

- **RbRadioButton8** : Name → ExamRB, Caption → Проведение экзамена

- **RbRadioButton9** : Name → DiplRb, Caption → Рук. диплом. проект.

- **RbRadioButton10** : Name → AspRB , Caption → Рук. аспирантами

- **RbRadioButton11** : Name → KonsBb, Caption → Консультации

- **RbRadioButton12** : Name → MetodRb, Caption → Методическая работа

- **GroupBox1**: Name → InfoBox.

Для автоматического внесения надписей в поле «Тема занятий» (или в случае его очистки) заполните обработчики событий **onClick** объектов **RbRadioButton** соответствуя нижеприведенным примерам:

```
procedure TMainForm.KonsBbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := 'Консультация';
end;

procedure TMainForm.MetodRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := 'Методическая работа';
end;
```

```
procedure TMainForm.LecRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := '';
end;

procedure TMainForm.PractRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := '';
end;
```

```
procedure TMainForm.LabRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := '';
end;

procedure TMainForm.KursRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := '';
end;
```

```
procedure TMainForm.PkrRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := 'Проверка К.Р.';
end;

procedure TMainForm.KkrRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := 'Проведение ККР';
end;
```

```
procedure TMainForm.DifzRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := 'Проведение диф. зачета';
end;

procedure TMainForm.ExamRbClick(Sender: TObject);
begin
    ТЕМАEdit.Text := 'Проведение экзамена';
end;
```

```

procedure TMainForm.DiplRbClick(Sender: TObject);
begin
    TEM&Edit.Text := 'Руководство диплом. проект.';
end;

```

```

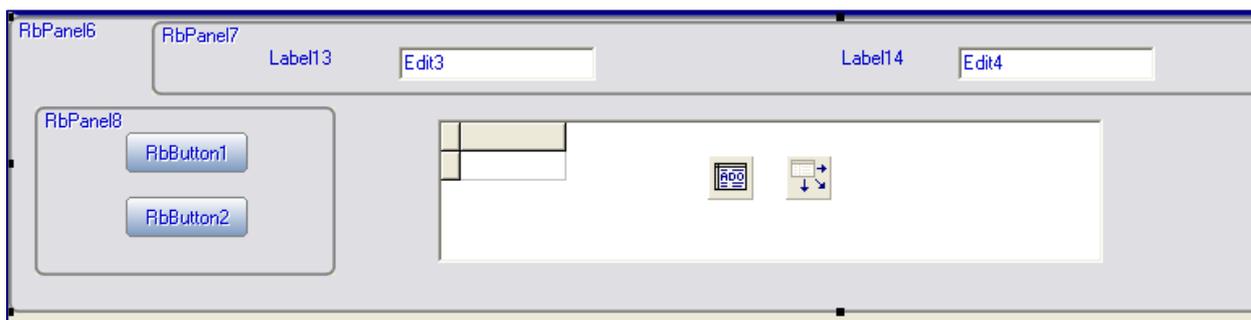
procedure TMainForm.AspRbClick(Sender: TObject);
begin
    TEM&Edit.Text := 'Руководство работами асп.';
end;

```

Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll** .

Формирование четвертой панели.

24. Установите в компонент **ScrollBox1** компонент **RbPanel** (на рис. RbPanel6) и измените значения таких свойств: **Name** → **PN3**, **Align** → **alTop**, **BorderColor** → **clTeal**, **BorderWidth**→**5**, **Enabled** → **False**.



25. На компонент панель **PN3** (на рис. RbPanel6) нанесите ещё два компонента **RbPanel** (**RbPanel 7** и **RbPanel8**) и сетку (**DBGrid**).

- **RbPanel 7** : **Align** → **alTop**. **Antialiased** → **False**
- **RbPanel 8** : **Align** → **alLeft**. **Antialiased** → **False**
- **DBGrid1** : **Align** → **alClient**.

26. Далее в соответствии с рисунком добавьте остальные компоненты и установите для их свойств такие значения:

- **RbButton1** : **Name** → **ADD_M**, **Glyph** → загрузить соответствующую картинку, **Hint** → Редактировать мероприятие, **ShowHint** → **True** , **Cursor** → **crHandPoint**.
- **RbButton2** : **Name** → **DEL_M**, **Glyph** → загрузить соответствующую картинку, **Hint** → Удалить запись, **ShowHint** → **True** , **Cursor** → **crHandPoint**.
- **Edit3** : **Name** → **FEEditGroup**

- **Edit4** : Name → FEditDisc
- **Label13** : Caption → Поиск: по группам
- **Label14** : Caption → по дисциплинам
- **ADOTable1** :
 - Name → M_LIST_TB,
 - Connection → ADOConnection,
 - TableName → M_LIST
 - Active → True

DataSource1: Name → M_LIST_DS; DataSet → M_LIST_TB

- **DBGrid1** : Name → M_LIST_Grid, DataSource → M_LIST_DS, +

Options: dgEditing → False, dgRowSelect → True.

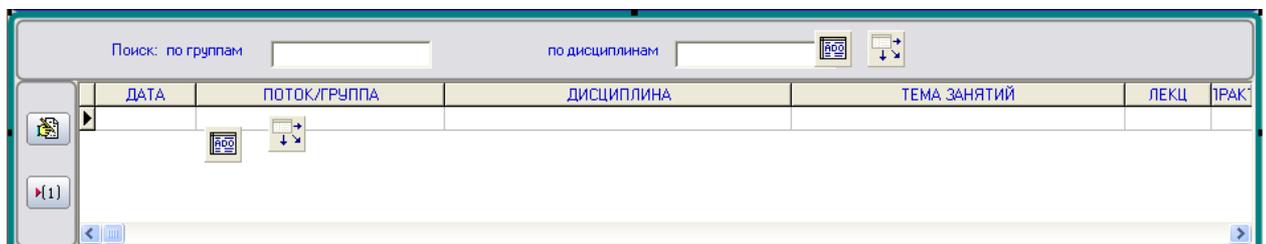
27. Выполните двойной клик мышкой по сетке (**DBGrid**). В открывшемся диалоговом окне вызовите контекстное меню и добавьте все поля к сетке.

28. Удалите отображение полей NUM, ПОТОК и KURS

29. Последовательность полей и русификация их заголовков должны соответствовать рисунку:

| | | |
|------------|---|--------------|
| 0 - M DAT | — | ДАТА |
| 1 - PGROUP | — | ПОТОК/ГРУППА |
| 2 - DISCIP | — | ДИСЦИПЛИНА |
| 3 - TEMA | — | ТЕМА ЗАНЯТИЙ |
| 4 - LEC | — | ЛЕКЦ |
| 5 - PRACT | — | ПРАКТ |
| 6 - LAB | — | ЛАБ |
| 7 - KONS | — | КОНС |
| 8 - PKR | — | ПКР |
| 9 - KKR | — | ККР |
| 10 - DIFZ | — | ЗАЧ |
| 11 - EXAM | — | ЭКЗ |
| 12 - DIPL | — | ДИПЛ |
| 13 - ASPIR | — | АСП |
| 14 - METOD | — | МЕТ |
| 15 - PNAME | — | Ф.И.О. |

30. Внешний вид четвертой панели:



31. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**.

Формирование пятой панели

32. Поставьте в компонент **ScrollBox1** компонент **RbPanel** и задайте его свойствам следующие значения:

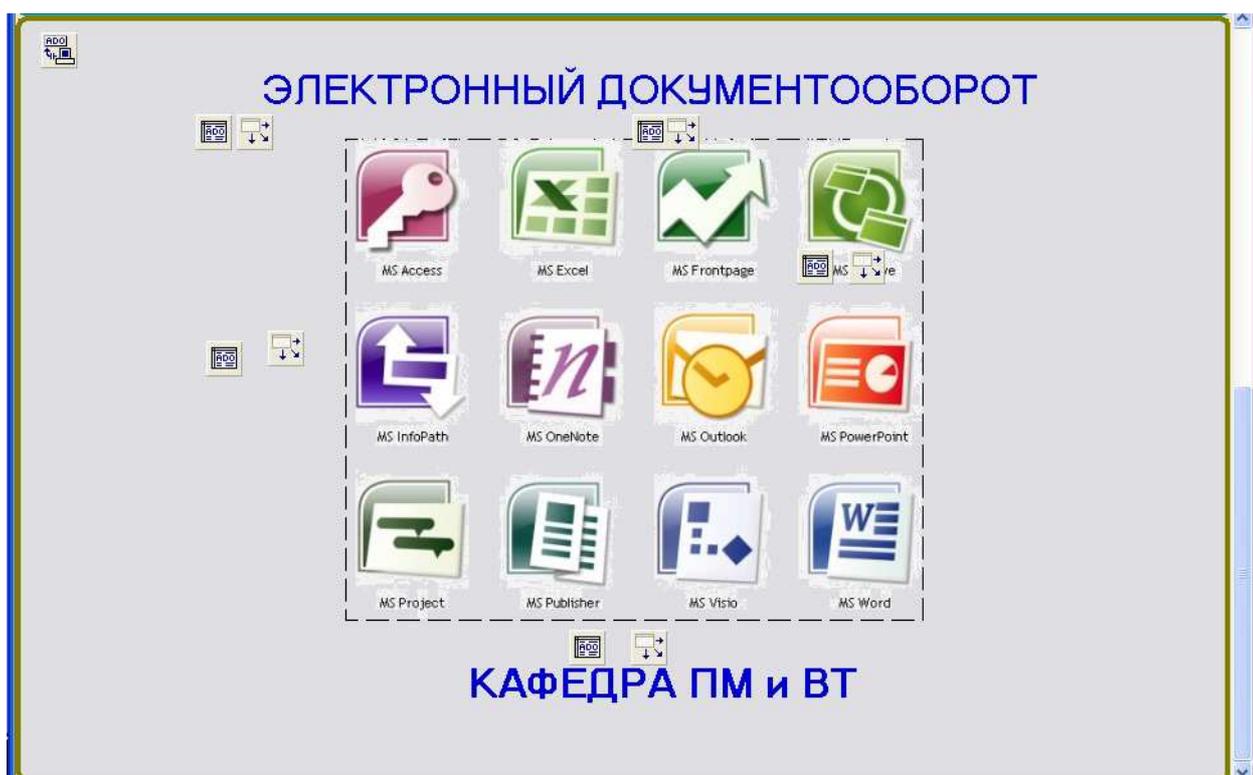
- **Name** → PN4, **Align** → alTop, **BorderColor**→ clOlive,
BorderWidth→5, **Height** → 600, **Antialiased** → False

33. Нанесите на панель (**PN4**) компоненты :

- **Image3** : **Transparent** → True, **Picture** → подгрузить соответствующую картинку.

- **Label13** : **Caption** → ЭЛЕКТРОННЫЙ ДОКУМЕНТООБОРОТ

- **Label14** : **Caption** → КАФЕДРА ПМ и ВТ

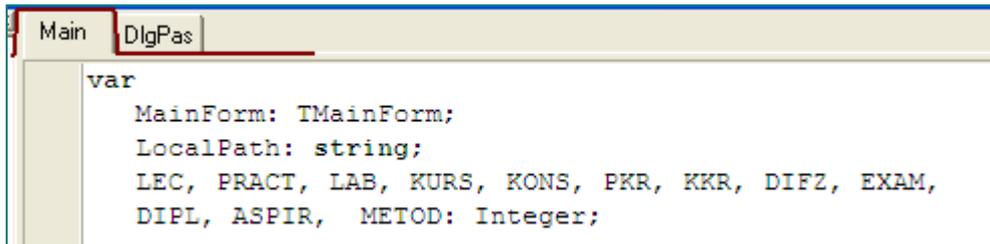


34. Подключите к каждому компоненту ADOTable (всего в проекте пять компонентов ADOTable) поля таблиц, с которыми он связан – для этого выполните двойной щелчок мышкой по компоненту ADOTable и через контекстное меню (команда Add all fields) добавьте поля.

После завершения дизайна главной формы начинаем постепенное заполнение окна *программного кода*.

Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll** .

35. В секции **var** Unit-а Main объявите переменные, участвующие в



```
var
    MainForm: TMainForm;
    LocalPath: string;
    LEC, PRACT, LAB, KURS, KONS, PKR, KKR, DIFZ, EXAM,
    DIPL, ASPIR, METHOD: Integer;
```

проекте:

В обработчик события создания формы FormCreate внесите следующую программу:

```
procedure TMainForm.FormCreate(Sender: TObject);
var
    TEMP, YEAR : string;
begin
    ADOConnection.Connected := false; // Закрываем соединение
    {Закрываем таблицы}
    D_LIST_TB.Active := false;
    G_LIST_TB.Active := false;
    P_LIST_TB.Active := false;
    R_LIST_TB.Active := false;
    M_LIST_TB.Active := false;
    {Определяем путь к udl файлу и описываем соединение}
    LocalPath := ExtractFilePath(application.ExeName);
    ADOConnection.ConnectionString := 'FILE NAME=' + LocalPath + 'LocalLink.udl';
    {Присваиваем текущее значение даты}
    DateTimePicker1.Date := Date;
    BeginDTP.Date := Date;
    EndDTP.Date := Date;
    {+++++++ Автоматическая установка учебного периода ++++++++}
    TEMP := DateToStr(Date); // присваиваем системную дату
    Delete(TEMP, 1, 3); // удаляем число месяца
    Delete(TEMP, 3, 5); // удаляем год (оставляем только месяц)
    YEAR := DateToStr(Date); // присваиваем системную дату
    Delete(YEAR, 1, 6); // удаляем число и месяц (оставляем только год)

    IF TEMP < '09' then // Если начался новый календарный год, то:
    begin
        BNSpinEdit.Value := StrToInt(YEAR) - 1; // Начало учебного периода = текущий год - 1
        ENSpinEdit.Value := StrToInt(YEAR); // Конец учебного периода = текущему году
    end else // В противном случае:
    begin
        BNSpinEdit.Value := StrToInt(YEAR); // Начало учебного периода = текущему году
        ENSpinEdit.Value := StrToInt(YEAR) + 1; // Конец учебного периода = текущий год + 1
    end;
end;
```

```

{=====}
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0; KKR := 0;
DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
{=====}
PN1.Visible := false;
PN2.Visible := false;
PN3.Visible := false;
PN3.Align := alClient;
PN4.Align := alClient;
end;

```

Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll** .

36. Для обеспечения диалогового доступа к базе данных (приложение «ЖУРНАЛ») используется ранее созданный UDL файл. Скопируйте файл LocalLink.udl в папке «Регистрация» и поместите его в папку «ЖУРНАЛ».

37. В описание используемых в проекте пользователей добавьте раздел **shellapi:**

```

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, RbDrawCore, RbPanel, RbButton, StdCtrls, Spin, ComCtrls, Grids,
  DBGrids, ExtCtrls, DB, ADODB, RbRadioButton, shellapi;

```

38. В обработчик события ConnectBtn (кнопка «Соединение» на контрольной панели) внесите следующую программу:

```

procedure TMainForm.ConnectBtnClick(Sender: TObject);
begin
  ADOConnection.Connected := false;
  ShellExecute(MainForm.Handle, 'open', PCHAR(LocalPath + 'LocalLink.udl'), nil, '', SW_SHOW);
end;

```

39. Скомпилируйте приложение, проверьте работу кнопки Соединение и после его остановки сохраните проект командой **File → SaveAll** .

ДИЗАЙН ДИАЛОГОВОГО ОКНА «ДОСТУП»

1. Подключите к создаваемому проекту новую форму – при сохранении формы (команда File → SaveAs, сохранять в папку создаваемого проекта «ЖУРНАЛ»), Unit переименуйте в DlgPas. После подключения созданной формы к проекту

в секции **implementation** задекларировать пользователей:

```

implementation
uses Main;
{SR *.DFM}
  
```

```

implementation
uses DlgPas;
  
```

не забудьте

НОВЫХ

2. Для вновь созданной формы выполните:

- *Name* → MyPas, *Caption* → ДОСТУП, *Position* → poScreenCenter.

3. На форму нанесите:

- **RbButton1**: *Name* → CancelBtn, *Caption* → Отменить,

ModalResult → mrCancel, *Cursor* → crHandPoint;

- **RbButton2**, *Name* → ОКBtn, *Caption* → Выполнить,

ModalResult → mrOk, *Enabled* → False, *Cursor* → crHandPoint.

- **RbPanel1** : *Align* → alTop, *Antialiased* → False.

Далее на панель (**RbPanel1**) нанесите:

- **Label** : *Name* → NfoLab, *Caption* → NfoLab, *Alignment* → taCenter, *AutoSize* → False, *Height* → 41, *Width* → 281,

Transparent → True, *WordWrap* → True.

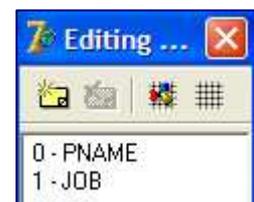
- **DBDrid1** : *DataSource*

MainForm.P_LIST_DS , в сетке оставьте только указанные на рисунке поля

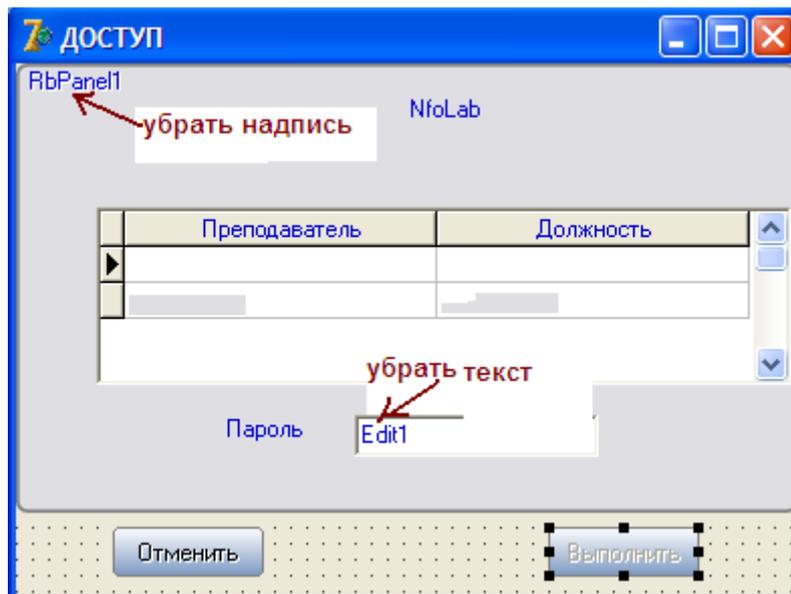
Options → dgEditing → False

- **Edit1**, *PasswordChar* → * (знак умножения);

- **Label1** : *Caption* → Пароль.



. Скомпилируйте приложение, проверьте его работу и после его



остановки досохраниите проект командой **File → SaveAll** .

4. Объявите логическую переменную RetVol:

```
Main | DlgPas
public
  { Public declarations }
  RetVol : boolean;
end;
```

5. В обработчике событий **FormActivate** формы **MyPas** присвойте переменной **RetVol** значение **ЛОЖЬ**.

```
procedure TMyPas.FormActivate(Sender: TObject);
begin
  RetVol := false;
end;
```

6. Для кнопки «**Выполнить**» запишите следующую процедуру:

```
procedure TMyPas.OKBtnClick(Sender: TObject);
begin
  RetVol := true;
end;
```

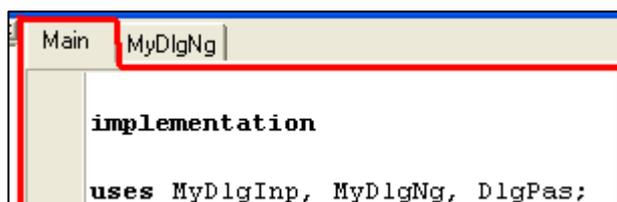
7. Заполните процедуру для обработчика событий **Edit1Change**

```
procedure TMyPas.Edit1Change(Sender: TObject);
begin
  If Edit1.Text <> '' then OKBtn.Enabled := true else OKBtn.Enabled := false;
end;
```

8. Скомпилируйте приложение, проверьте его работу и после его остановки досохраниите проект командой **File → SaveAll** .

ДИЗАЙН ДИАЛОГОВОГО ОКНА «ГОДОВОЙ ПЛАН»

1. Подключите к создаваемому проекту новую форму – при сохранении формы (команда File → SaveAs, сохранять в папку создаваемого проекта «ЖУРНАЛ»), Unit переименуйте в MyDlgNg. После подключения созданной формы к проекту не забудьте в секции **implementation** (каждого из модулей) задекларировать новых пользователей:



2. Для вновь созданной формы **Form1** выполните: Name → MyNg,Form, Caption → Годовой план, Position → poScreenCenter.

3. На форму нанесите:

- **RbButton1**, Name → ОКBtn, Caption → Выполнить, ModalResult → mrOk, Cursor → crHandPoint.

- **RbPanel1**: Align → alTop, Antialiased → False.

Далее на панель (**RbPanel1**) нанесите:

- **Label**: Name → NfoLab, Caption → NfoLab, Alignment → taCenter, AutoSize → False, Height → 41, Width → 329, Transparent → True, WordWrap → True.

- двенадцать компонентов **Label** (Label1 - Label12). Расположить их на панели **RbPanel1** и заполнить только значение свойства **Caption** в соответствии с рис.,

- двенадцать компонентов **DBEdit** (DBEdit1 - DBEdit12). Расположить их на панели **RbPanel1** в соответствии с рис.:

- DBEdit1: DataSource → MainForm.P_LIST_DS, DataField → LEK;
- DBEdit2: DataSource → MainForm.P_LIST_DS, DataField → PRACT;
- DBEdit3: DataSource → MainForm.P_LIST_DS, DataField → LAB;
- DBEdit4: DataSource → MainForm.P_LIST_DS, DataField → KURS;
- DBEdit5: DataSource → MainForm.P_LIST_DS, DataField → PKR;
- DBEdit6: DataSource → MainForm.P_LIST_DS, DataField → KKR;
- DBEdit7: DataSource → MainForm.P_LIST_DS, DataField → DIFZ;

- DBEdit8: DataSource → MainForm.P_LIST_DS, DataField → EXAM;
- DBEdit9: DataSource → MainForm.P_LIST_DS, DataField → DIPL;
- DBEdit10: DataSource → MainForm.P_LIST_DS, DataField → ASPIR;
- DBEdit11: DataSource → MainForm.P_LIST_DS, DataField → KONS;
- DBEdit12: DataSource → MainForm.P_LIST_DS, DataField → METHOD;

4. Объявите логическую переменную RetVol:

```

Main  MyDlgNg
public
  { Public declarations }
  RetVol : boolean;
end;

```

5. В обработчике событий **FormActivate** формы MyNg,Form присвойте переменной **RetVol** значение **ЛОЖЬ**.

```

procedure TMyPas.FormActivate(Sender: TObject);
begin
  RetVol := false;
end;

```

6. Для кнопки «**Выполнить**» запишите следующую процедуру:

```

procedure TMyPas.OKBtnClick(Sender: TObject);
begin
  RetVol := true;
end;

```

7. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll** .

После выполнения окна «Годовой план» продолжим работу с главной формой.

1. Для установки связей в приложении созданы личные процедуры: procedure **Link**, procedure **RedFilter**, их необходимо объявить в разделе Private :

```
private
  { Private declarations }
procedure Link;
procedure RedFilter;
public
  { Public declarations }
end;
```

Личные процедуры **обязательно** записываются сразу после секции **implementation** (**вся** процедура прописывается пользователем):

```
implementation
  uses DlgPas;
  {$R *.dfm}
  {+++++++ Личные методы =====}
  procedure TMainForm.Link;
  begin
    R_LIST_TB.Filtered := false;
    R_LIST_TB.Filter := 'ПОТОК = ' + QuotedStr(G_LIST_TBPGROUP.Value);
    R_LIST_TB.Filtered := true;
  end;
  procedure TMainForm.RedFilter;
  begin
    M_LIST_TB.Filtered := false;
    M_LIST_TB.Filter := 'PNAME = ' + QuotedStr(P_LIST_TBPNNAME.Value);
    M_LIST_TB.Filtered := true;
  end;
```

Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll** .

2. В папке с разрабатываемым проектом (папка «Журнал») должен находиться файл ресурсов CodeStr (при отсутствии файла обратитесь к

преподавателю). Допишите в раздел используемых в проекте модулей модуль CodeStr:

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, RbButton, RbDrawCore, RbPanel, ComCtrls, StdCtrls, Spin, DB, ADODB, Grids, DBGrids, ExtCtrls, shellapi, **CodeStr**;

В обработчик событий OpenBtn (кнопка «Доступ» на контрольной панели) внесите следующую программу:

```
procedure TMainForm.OpenBtnClick(Sender: TObject);
var
  Pas: integer;
  SP: Boolean;
begin
  SP := false;
  ADOConnection.Connected := OpenBtn.Down; // Устанавливаем соединение
  {Открываем таблицы}
  D_LIST_TB.Active := OpenBtn.Down;
  G_LIST_TB.Active := OpenBtn.Down;
  P_LIST_TB.Active := OpenBtn.Down;
  R_LIST_TB.Active := OpenBtn.Down;
  M_LIST_TB.Active := OpenBtn.Down;
  {Устанавливаем индексы сортировки}
  D_LIST_TB.IndexFieldNames := 'DISCIP';
  G_LIST_TB.IndexFieldNames := 'PGROUP';
  P_LIST_TB.IndexFieldNames := 'PNAME';
  R_LIST_TB.IndexFieldNames := 'DISCIP';
  M_LIST_TB.IndexFieldNames := 'MDAT';
  {Делаем видимыми панели}
  PN1.Enabled := OpenBtn.Down;
  PN2.Enabled := OpenBtn.Down;
  PN3.Enabled := OpenBtn.Down;
```

```

Image2.Visible := OpenBtn.Down;
Link; // Устанавливаем динамическую связь
RedFilter; //
//P_LIST_TB.AfterScroll(nil); //
{=====}
if OpenBtn.Down then
begin
  if FLT_P.Down = true then FLT_P.Click;
  ADD_P.Enabled := false;
  DEL_P.Enabled := false;
  MyPas.NfoLab.Font.Color := clBlack;
  MyPas.Edit1.Text := '';
  MyPas.NfoLab.Caption := 'ВЫБЕРИТЕ В СПИСКЕ ФАМИЛИЮ, ИНИЦИАЛЫ И ВВЕДИТЕ ПАРОЛЬ';
  MyPas.ShowModal; // Вызов диалогового окна
  if MyPas.RetVol = true then // Если флаг действительный, то:
  begin
    if MyPas.Edit1.Text = 'TOPKEY' then
    begin
      if FLT_P.Down = true then FLT_P.Click;
      EditF.Enabled := true;
      PN3.Visible := true;
      PN2.Visible := true;
      PN1.Visible := true;
      PN4.Visible := false;
      UREPBTN.Enabled := OpenBtn.Down;
      ADD_P.Enabled := true;
      DEL_P.Enabled := true;
      FLT_P.Enabled := true;
      EditF.Enabled := true;
      SP := true;
    end;
  end;
end;

```

```

Pas := CRC32(-1, trim(MyPas.Edit1.Text));
if Pas = P_LIST_TBPAS.Value then
begin
if FLT_P.Down = false then FLT_P.Click;
FLT_P.Enabled := false;
PN3.Visible := true;
PN2.Visible := true;
PN1.Visible := true;
PN4.Visible := false;
UREPBTN.Enabled := OpenBtn.Down;
EditF.Enabled := false;
SP := true;
end;

if SP = false then ShowMessage('НЕ ВЕРНЫЙ ПАРОЛЬ! ДОСТУП ЗАПРЕЩЕН!');

end;
end else
begin
PN3.Visible := false;
PN2.Visible := false;
PN1.Visible := false;
PN4.Visible := true;
UREPBTN.Down := false;
UREPBTN.Enabled := false;
end;
end;

```

3. Для компонента P_LIST_TB в обработчик событий AfterScroll внесите следующий код:

```

procedure TMainForm.P_LIST_TBAfterScroll(DataSet: TDataSet);
begin
RedFilter;
if P_LIST_TBREPORT.IsNull = true then Image1.Visible := false else Image1.Visible := true;
end;

```

4. Скомпилируйте приложение, проверьте работу кнопок **Соединение** и **Доступ** и после его остановки сохраните проект командой **File → SaveAll**

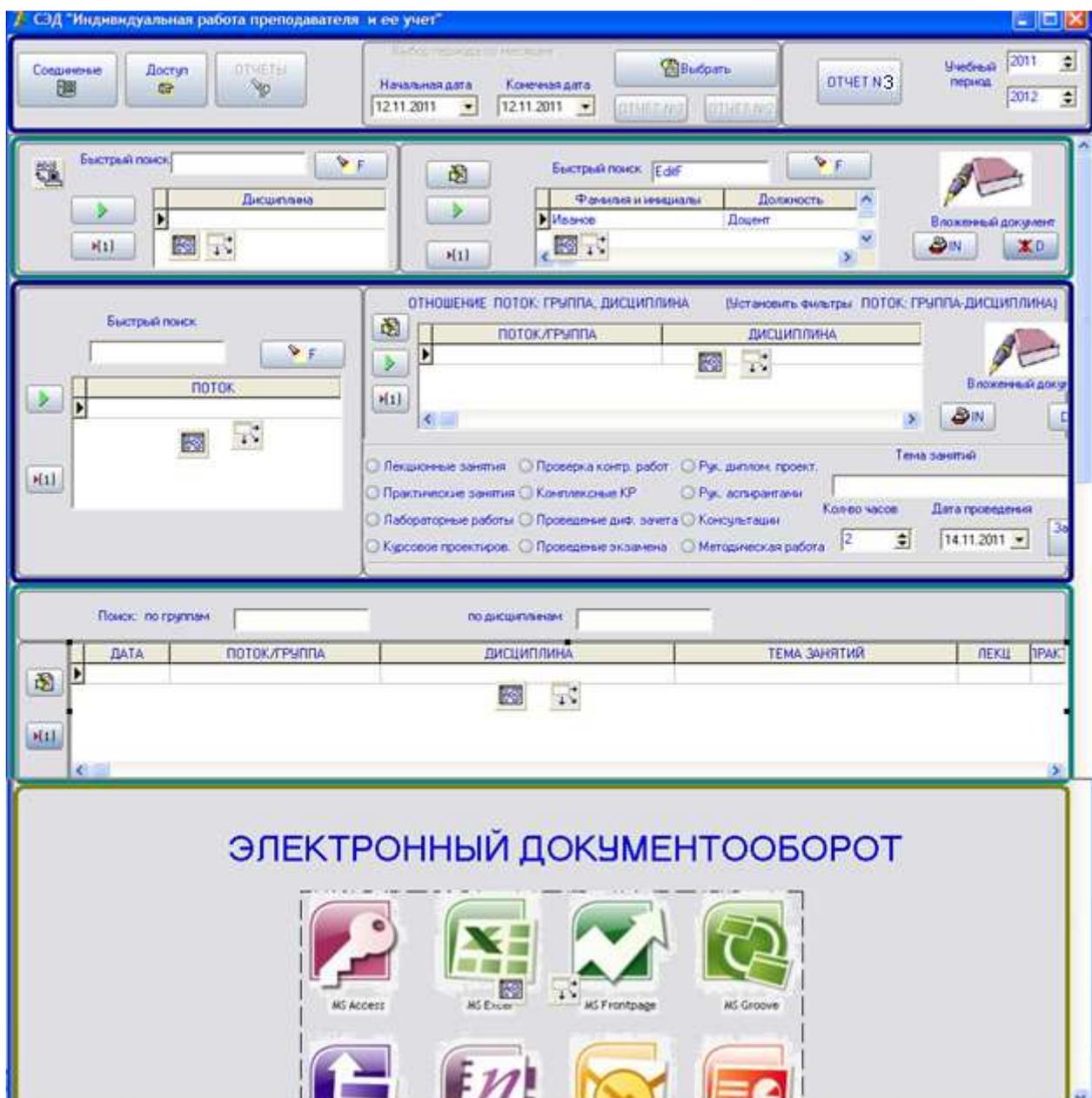
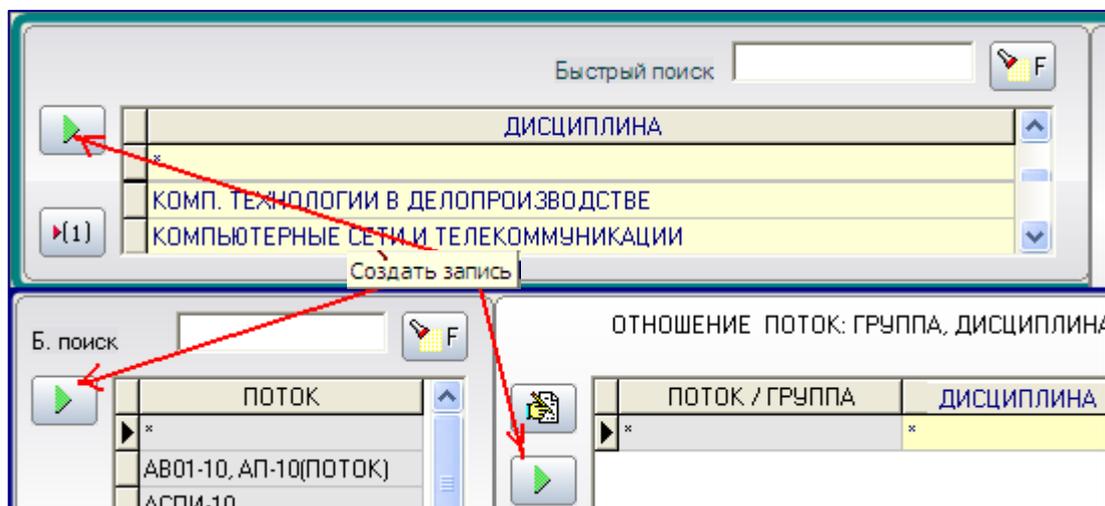
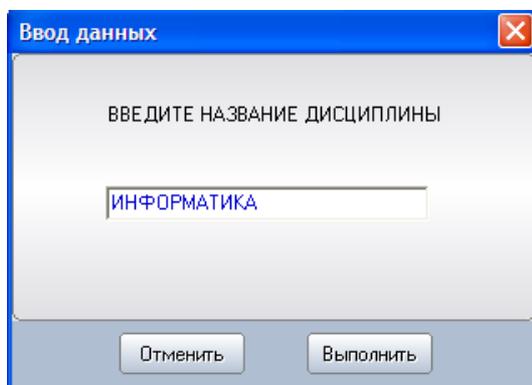


Рис. Дизайн главной формы

Ввод названия дисциплины (панель PN1), потока и группы(панель PN2) выполняется через окно **Ввод данных**. Для этого единожды создаем форму (**Ввод данных**) и она будет открываться при нажатии на одну из указанных кнопок.



ДИЗАЙН ОКНА «Ввод данных»



1. Подключите к проекту новую форму, предварительно сохранив ее Unit под именем **MyDlgInp** в папке проекта «Журнал». Не забудьте затем в секциях **implementation** Unit-ов проекта объявить вновь подключенный Unit (в Main → uses DlgPas, MyDlgInp, и т.п.).
2. Саму форму **Form1** переименуйте **Name** → MyDialogInp, **Caption** → Ввод данных.
3. Нанесите на форму компоненты:
 - **RbButton1**: Name → CancelBtn, Caption → Отменить, ModalResult → mrCancel, Cursor → crHandPoint;

- **RbButton2**, Name → OKBtn, Caption → Выполнить, ModalResult → mrOk, Enabled → False, Cursor → crHandPoint.

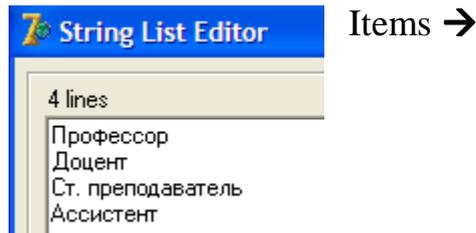
- **RbPanel1** : *Align* → alTop, *Antialiased* → False.

Далее на панель (RbPanel1) нанесите:

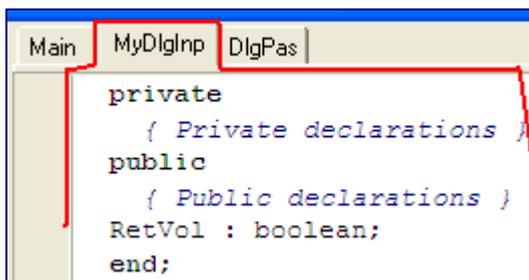
- **Label** : Name → NfoLab, Caption → NfoLab, Alignment → taCenter, AutoSize → False, Height → 41, Width → 281, Transparent → True, WordWrap → True

- **Edit1**,

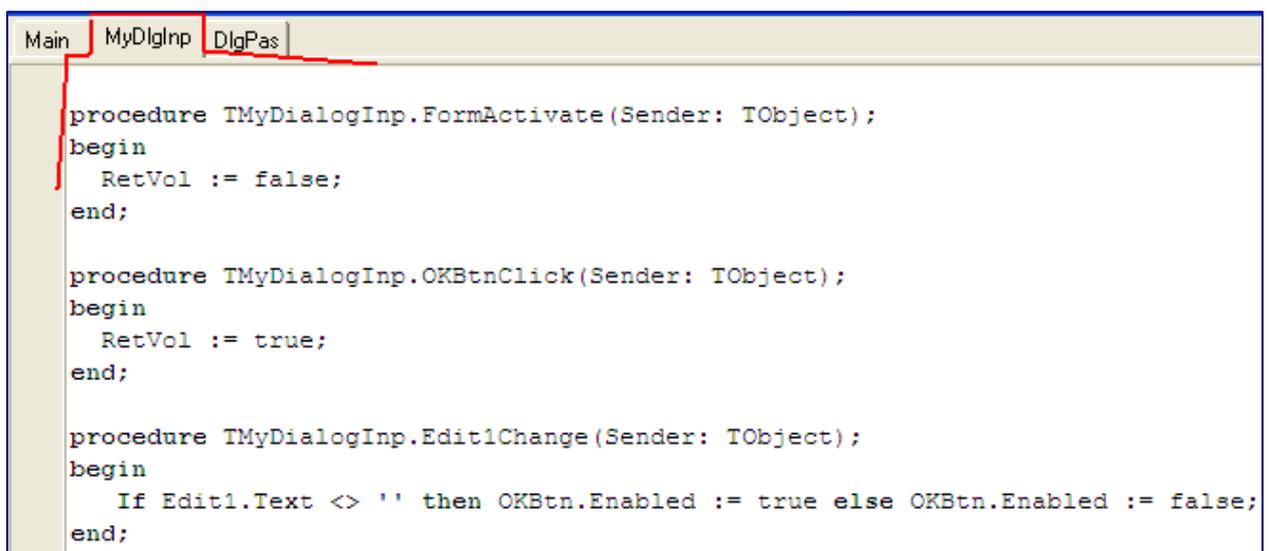
- **ComboBox1** :



4. Объявите логическую переменную **RetVol**:



5. Напишите для соответствующих обработчиков событий их программные коды:



Скомпилируйте приложение, проверьте работу и после его остановки сохраните проект командой **File → SaveAll**.

6. Для кнопки позволяющей создать запись ДИСЦИПЛИНЫ (компонент ADD_D на панели PN1) запишите следующий программный

```
procedure TMainForm.ADD_DClick(Sender: TObject);
begin
  MyDialogInp.ComboBox1.Visible := false;
  MyDialogInp.NfoLab.Font.Color := clBlack;
  MyDialogInp.Edit1.Text := '';
  MyDialogInp.NfoLab.Caption := 'ВВЕДИТЕ НАЗВАНИЕ ДИСЦИПЛИНЫ';
  MyDialogInp.ShowModal; // Вызов диалогового окна
  if MyDialogInp.RetVol = true then // Если флаг действительный, то:
  begin
    D_LIST_TB.Append; // Добавляем запись
    D_LIST_TB.DISCIP.Value := MyDialogInp.Edit1.Text; // Присваиваем значение поля
    D_LIST_TB.Post; // Сохраняем запись
    D_LIST_TB.Refresh; // Обновляем DataSet
  end;
end;
```

КОД:

Для кнопки позволяющей удалить ДИСЦИПЛИНУ (компонент DEL_D на панели PN1) запишите следующий программный код:

```
procedure TMainForm.DEL_DClick(Sender: TObject);
begin
  if D_LIST_TB.RecordCount <> 0 then
  begin
    MyDialog.NfoLab.Font.Color := clRed;
    MyDialog.NfoLab.Caption := 'ДИСЦИПЛИНА: ' + D_LIST_TB.DISCIP.Value + ' БУДЕТ УДАЛЕНА!';
    MyDialog.ShowModal; // Вызываем диалоговое окно
    if MyDialog.RetVol = true then // Если флаг действительный, то:
    begin
      D_LIST_TB.Delete; // Удаляем запись, если не начало таблицы
      D_LIST_TB.Refresh; // Обновляем DataSet
    end;
  end;
end;
```

Скомпилируйте приложение, проверьте работу «оживленных» кнопок и после его остановки сохраните проект командой **File → SaveAll**.

7. Для кнопок (панель PN1), позволяющих редактировать *плановые академические часы, регистрировать преподавателей, удалять записи* (компоненты EDIT_P, ADD_P и DEL_P соответственно) заполните следующие обработчики событий:

```

procedure TMainForm.EDIT_PClick(Sender: TObject);
begin
if P_LIST_TB.RecordCount > 0 then
  begin
    MyNgForm.NfoLab.Caption := 'Плановые академические часы ' +
      P_LIST_TBJOB.Value + ' ' + P_LIST_TBPNAMЕ.Value ;
    P_LIST_TB.Edit;
    MyNgForm.showModal;
    P_LIST_TB.Post;
  end;
end;

```

```

procedure TMainForm.ADD_PClick(Sender: TObject);
begin
  MyDialogInp.ComboBox1.Visible := true;
  MyDialogInp.NfoLab.Font.Color := clBlack;
  MyDialogInp.Edit1.Text := '';
  MyDialogInp.NfoLab.Caption := 'ВВЕДИТЕ ФАМИЛИЮ, ИНИЦИАЛЫ И ДОЛЖНОСТЬ';
  MyDialogInp.ShowModal;           // Вызов диалогового окна
  if MyDialogInp.RetVol = true then // Если флаг действительный, то:
  begin
    P_LIST_TB.Append;                // Добавляем запись
    P_LIST_TBPNAMЕ.Value := MyDialogInp.Edit1.Text; // Присваиваем значение поля
    P_LIST_TBJOB.Value := MyDialogInp.ComboBox1.Text;
    P_LIST_TB.Post;                  // Сохраняем запись
    P_LIST_TB.Refresh;              // Обновляем DataSet
  end;
end;

```

```

procedure TMainForm.DEL_PClick(Sender: TObject);
begin
if P_LIST_TB.RecordCount <> 0 then
  begin
    MyDialog.NfoLab.Font.Color := clRed;
    MyDialog.NfoLab.Caption := 'ПРЕПОДАВАТЕЛЬ: ' +
      P_LIST_TBPNAMЕ.Value + ' БУДЕТ УДАЛЕН !';
    MyDialog.ShowModal;           // Вызываем диалоговое окно
    if MyDialog.RetVol = true then // Если флаг действительный, то:
    begin
      P_LIST_TB.Delete; // Удаляем запись, если не начало таблицы
      P_LIST_TB.Refresh; // Обновляем DataSet
    end;
  end;
end;

```

7. Для кнопки позволяющей создать запись ПОТОК (компонент ADD_G на панели PN2) запишите следующий программный код:

```
procedure TMainForm.ADD_GClick(Sender: TObject);
begin
  MyDialogInp.ComboBox1.Visible := false;
  MyDialogInp.NfoLab.Font.Color := clBlack;
  MyDialogInp.Edit1.Text := '';
  MyDialogInp.NfoLab.Caption := 'ВВЕДИТЕ НАЗВАНИЕ ПОТОКА';
  MyDialogInp.ShowModal;           // Вызов диалогового окна
  if MyDialogInp.RetVol = true then // Если флаг действительный, то:
  begin
    G_LIST_TB.Append;                // Добавляем запись
    G_LIST_TBPGROUP.Value := MyDialogInp.Edit1.Text; // Присваиваем значение поля
    G_LIST_TB.Post;                 // Сохраняем запись
    G_LIST_TB.Refresh;              // Обновляем DataSet
  end;
end;
```

8. Для кнопки позволяющей удалить название ПОТОКА (компонент DEL_G на панели PN2) запишите следующий программный код:

```
procedure TMainForm.DEL_GClick(Sender: TObject);
begin
  if G_LIST_TB.RecordCount <> 0 then
  begin
    MyDialog.NfoLab.Font.Color := clRed;
    MyDialog.NfoLab.Caption := 'ГРУППА, ПОТОК: ' + G_LIST_TBPGROUP.Value + ' БУДУТ УДАЛЕНЫ!';
    MyDialog.ShowModal;           // Вызываем диалоговое окно
    if MyDialog.RetVol = true then // Если флаг действительный, то:
    begin
      G_LIST_TB.Delete; // Удаляем запись, если не начало таблицы
      G_LIST_TB.Refresh; // Обновляем DataSet
    end;
  end;
end;
```

Скомпилируйте приложение, проверьте работу «оживленных» кнопок и после его остановки сохраните проект командой **File → SaveAll**.

9. Для кнопки позволяющей создать запись новой группы в R_LIST_Grid (компонент ADD_R_G на панели PN2) запишите следующий программный код:

```
procedure TMainForm.ADD_R_GClick(Sender: TObject);
begin
  MyDialogInp.ComboBox1.Visible := false;
  MyDialogInp.NfoLab.Font.Color := clBlack;
  MyDialogInp.Edit1.Text := '';
  MyDialogInp.NfoLab.Caption := 'ВВЕДИТЕ НАЗВАНИЕ ГРУППЫ ВХОДЯЩЕЙ В ПОТОК';
  MyDialogInp.ShowModal;           // Вызов диалогового окна
  if MyDialogInp.RetVol = true then // Если флаг действительный, то:
  begin
    R_LIST_TB.Append;              // Добавляем запись
    R_LIST_TBPGROUP.Value := MyDialogInp.Edit1.Text; // Присваиваем значение поля
    R_LIST_TBPOTOK.Value := G_LIST_TBPGROUP.Value;
    R_LIST_TBDISCIP.Value := D_LIST_TBDISCIP.Value;
    R_LIST_TB.Post;               // Сохраняем запись
    R_LIST_TB.Refresh;           // Обновляем DataSet
  end;
end;
```

10. Для кнопки позволяющей создать запись потока в R_LIST_Grid (компонент ADD_R на панели PN2) запишите следующий программный код:

```
procedure TMainForm.ADD_RClick(Sender: TObject);
begin
  MyDialog.NfoLab.Font.Color := clBlue;
  MyDialog.NfoLab.Caption := 'БУДУТ ДОТАВЛЕННЫ: ДИСЦИПЛИНА - '
  + D_LIST_TBDISCIP.Value + ', ПОТОК - ' + G_LIST_TBPGROUP.Value;
  MyDialog.ShowModal;           // Вызываем диалоговое окно
  if MyDialog.RetVol = true then // Если флаг действительный, то:
  begin
    R_LIST_TB.Append;
    R_LIST_TBDISCIP.Value := D_LIST_TBDISCIP.Value;
    R_LIST_TBPGROUP.Value := G_LIST_TBPGROUP.Value;
    R_LIST_TBPOTOK.Value := G_LIST_TBPGROUP.Value;
    R_LIST_TB.Post;
    R_LIST_TB.Refresh;           // Обновляем DataSet
    Link;
  end;
end;
```

11. Для кнопки позволяющей удалить запись ПОТОК/ГРУППА (компонент DEL_R на панели PN2) запишите следующий программный код

```
procedure TMainForm.DEL_RClick(Sender: TObject);
begin
if R_LIST_TB.RecordCount <> 0 then
begin
MyDialog.NfoLab.Font.Color := clRed;
MyDialog.NfoLab.Caption := 'ЗАПИСЬ: БУДЕТ УДАЛЕНА !';
MyDialog.ShowModal; // Вызываем диалоговое окно
if MyDialog.RetVol = true then // Если флаг действительный, то:
begin
R_LIST_TB.Delete; // Удаляем запись, если не начало таблицы
R_LIST_TB.Refresh; // Обновляем DataSet
end;
end;
end;
```

12. Для вывода информационных данных, которые будут занесены в таблицы БД, создан личный метод (пользовательская процедура) **TransInfo**. Для работы фильтра создана личная процедура **SelfFilter**. Объявите эти процедуры в разделе **private {Private declarations}**:

```
private
( Private declarations )
procedure Link;
procedure TransInfo;
procedure RedFilter;
procedure SelfFilter;
public
( Public declarations )
end;
```

и запишите их программные коды.

```
procedure TMainForm.TransInfo;
begin
if (R_LIST_TB.RecordCount <> 0) and (P_LIST_TB.RecordCount <> 0) then
begin
InfoBox.Caption := '';
InfoBox.Caption := ' ' + 'Преподаватель: ' + P_LIST_TB.PNAME.Value +
' Группа: ' + R_LIST_TB.PGROUP.Value + ' Дисциплина: ' + R_LIST_TB.DISCIP.Value
+ ' ';
end else
InfoBox.Caption := '';
end;
```

```

procedure TMainForm.SelFilter;
var
BG, EN: String;
begin
BG := DateToStr(BeginDTP.Date);
EN := DateToStr(EndDTP.Date);
M_LIST_TB.Filtered := false;
M_LIST_TB.Filtered := 'MDAT>='+BG+'AND'+ 'MDAT<='+EN+'AND'+ 'PNAME='+QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
end;
(=====)

```

13. Для компонента G_LIST_TB в обработчик событий AfterScroll внесите следующий код:

```

procedure TMainForm.G_LIST_TBAfterScroll(DataSet: TDataSet);
begin
Link;
end;

```

14. Для компонента R_LIST_TB, в обработчик событий AfterScroll, внесите следующий код:

```

procedure TMainForm.R_LIST_TBAfterScroll(DataSet: TDataSet);
begin
TransInfo;
if R_LIST_TBJOURNAL.IsNull = true then Image2.Visible := false else Image2.Visible := true;
end;

```

15. Для компонента P_LIST_Grid в соответствующих обработчиках событий (OnCellClick и OnKeyUp) запишите следующие коды:

```

procedure TMainForm.P_LIST_GridCellClick(Column: TColumn);
begin
TransInfo;
end;

procedure TMainForm.P_LIST_GridKeyUp(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
TransInfo;
end;

```

16. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**.

ФОРМИРОВАНИЕ ФОРМЫ «СООБЩЕНИЕ»

1. Подключите к проекту новую форму, предварительно сохранив ее Unit в папке проекта под именем MyDlg. Не забудьте затем в секции **implementation** Unit –а **Main** объявить вновь подключенный Unit :

uses DlgPas, MyDlgInp, MyDlg;

Саму форму **Form1** переименуйте **Name** → MyDialog, **Caption** → Сообщение , **Position** → poScreenCenter.

2. Нанесите на форму компоненты:

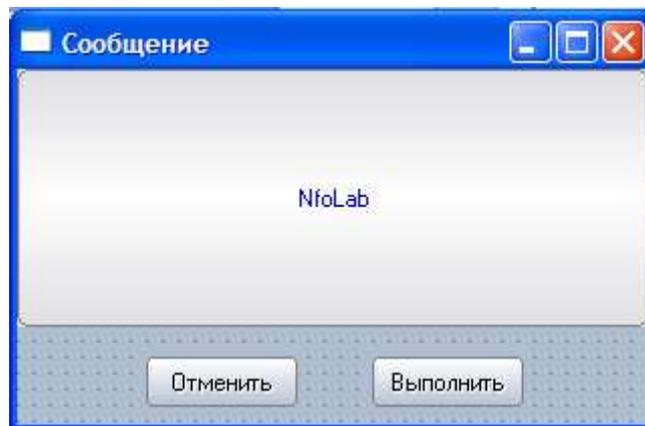
- **RbButton1**: Name → CancelBtn, Caption → Отменить, ModalResult→mrCancel, Cursor→ crHandPoint;

- **RbButton2**, Name →OKBtn, Caption → Выполнить, ModalResult→mrOk, Cursor→ crHandPoint.

- **RbPanel1** : Align → alTop, Antialiased → False.

Далее на панель (**RbPanel1**) нанесите:

- **Label** : Name → NfoLab, Caption → NfoLab, Alignment → taCenter, AutoSize → False, Height → 97, Width → 281, Transparent → True, WordWrap



→ True.

3. Объявите в модуле **MyDlg** переменную RetVol:

```
public
  { Public declarations }
  RetVol : boolean;
end;
```

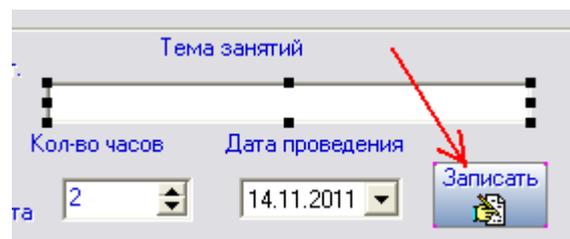
4. Для обработчика событий **FormActivate** и кнопки «Выполнить» запишите соответственно программы:

```
procedure TMyDialog.FormActivate(Sender: TObject);
begin
  RetVol := false;
end;

procedure TMyDialog.OKBtnClick(Sender: TObject);
begin
  RetVol := true;
end;
```

Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll** .

5. Перейдите на главную форму. Для компонента **ADD_M1** (кнопка «Записать» на панели PN2) в его обработчик событий **ADD_M1Click** внесите следующий программный код:



```

procedure TMainForm.ADD_M1Click(Sender: TObject);
Var
DT : string;
begin
  DT := DateToStr(DateTimePicker1.Date);
  MyDialog.NfoLab.Font.Color := clBlue;
  MyDialog.NfoLab.Caption := 'БУДУТ ДОТАВЛЕННЫ: ' + InfoBox.Caption ;
  MyDialog.ShowModal;           // Вызываем диалоговое окно
  if MyDialog.RetVol = true then // Если флаг действительный, то:
  begin
    M_LIST_TB.Append;
    M_LIST_TBDISCIP.Value := R_LIST_TBDISCIP.Value;
    M_LIST_TBPGROUP.Value := R_LIST_TBPGROUP.Value;
    M_LIST_TBPOTOK.Value := R_LIST_TBPOTOK.Value;
    M_LIST_TBPNAME.Value := P_LIST_TBPNAME.Value;
    M_LIST_TBMDAT.Value := StrToDate(DT);
    M_LIST_TBTEMA.Value := TEMAEdit.Text;

    if LecRb.Checked then M_LIST_TBLEC.Value := ClockEdit.Value;
    if PractRb.Checked then M_LIST_TBPRACT.Value := ClockEdit.Value;
    if LabRb.Checked then M_LIST_TBLAB.Value := ClockEdit.Value;
    if KursRb.Checked then M_LIST_TBKURS.Value := ClockEdit.Value;
    if PkrRB.Checked then M_LIST_TBPKR.Value := ClockEdit.Value;
    if KkrRB.Checked then M_LIST_TBKKR.Value := ClockEdit.Value;
    if DifzRB.Checked then M_LIST_TBDIFZ.Value := ClockEdit.Value;
    if ExamRB.Checked then M_LIST_TBEXAM.Value := ClockEdit.Value;
    if DiplRb.Checked then M_LIST_TBDIPL.Value := ClockEdit.Value;
    if AsprB.Checked then M_LIST_TBASPIR.Value := ClockEdit.Value;

    if KonsBb.Checked then
    begin
      M_LIST_TBPGROUP.Value := '*...*';
      M_LIST_TBDISCIP.Value := 'КОНСУЛЬТАЦИЯ';
      M_LIST_TBKONS.Value := ClockEdit.Value;
      M_LIST_TBPOTOK.Value := 'КОНСУЛЬТАЦИЯ';
    end;
    if MetodRb.Checked then
    begin
      M_LIST_TBPGROUP.Value := '*...*';
      M_LIST_TBDISCIP.Value := 'МЕТОДИЧЕСКАЯ РАБОТА';
      M_LIST_TBMETHOD.Value := ClockEdit.Value;
      M_LIST_TBPOTOK.Value := 'МЕТОДИЧЕСКАЯ РАБОТА';
    end;
    M_LIST_TB.Post;
    M_LIST_TB.Refresh;           // Обновляем DataSet
  end;
end;

```

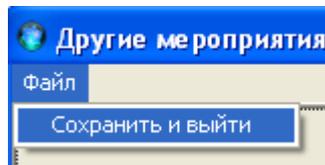
Создание формы «Другие мероприятия»

1. Подключите к проекту новую форму, предварительно сохранив ее Unit под именем **pCont** в папке проекта «Журнал». Не забудьте затем в секциях **implementation** Unit-ов проекта объявить вновь подключенный Unit (в Main → uses DlgPas, MyDlgInp, **pCont**, и т.п.).

2. Саму форму **Form1** переименуйте Name → PContForm, Caption → Другие мероприятия.

3. Нанесите на форму компоненты: MainMenu1 и DBOleContainer1.

4. Создайте два пункта меню в соответствии с рисунком. У пункта «Файл» для свойства Name → File1, Caption → &Файл. В пункте «Сохранить и выйти» для свойства Name → SaveOle1, Caption → Сохранить и выйти.



В соответствии с рисунком заполните обработчик событий для пункта «Сохранить и выйти».

```
procedure TpContForm.SaveOle1Click(Sender: TObject);  
begin  
    MainForm.P_LIST_TB.Edit;  
    MainForm.P_LIST_TB.Post;  
    close;  
end;
```

Для формы PContForm в обработчик события onClose внесите код:

```
procedure TpContForm.FormClose(Sender: TObject; var Action: TCloseAction);  
begin  
    MainForm.P_LIST_TB.Edit;  
    MainForm.P_LIST_TB.Post;  
end;
```

Для компонента DBOleContainer1: Name → DBOleContainer1, Alignment → alClient, DataSource → MainForm.P_LIST_DS, DataField → REPORT, Height → 300, Width → 360.

5. Перейти на главную форму MainForm.

Для компонента **INSCont2** (кнопка) в соответствующий обработчик событий внести программный код:

```

procedure TMainForm.INSCont2Click(Sender: TObject);
begin
if P_LIST_TB.RecordCount <> 0 then // Если имеются записи, то:
begin
  MyDialog.NfoLab.Font.Color := clBlue;
  MyDialog.NfoLab.Caption := 'ОТЧЕТ ПРЕПОДАВАТЕЛЯ ' + P_LIST_TB.PNAME.Value;
  MyDialog.ShowModal; // Вызываем диалоговое окно
  if MyDialog.RetVol = true then // Если флаг действительный, то:
  begin
    pContForm.ShowModal; // Открываем форму контейнера
    P_LIST_TB.Edit; // Режим редактирования
  end;
  // P_LIST_TB.Post; // Сохраняем загрузку
  P_LIST_TB.Refresh; // Обновляем DataSet
end;
end;

```

Для компонента **DelCont2** (кнопка) в соответствующий обработчик событий внести программный код:

```

procedure TMainForm.DelCont2Click(Sender: TObject);
begin
if P_LIST_TB.RecordCount <> 0 then
begin
  MyDialog.NfoLab.Font.Color := clRed;
  MyDialog.NfoLab.Caption := 'ОТЧЕТ ПРЕПОДАВАТЕЛЯ ' + P_LIST_TB.PNAME.Value +
    ' БУДЕТ УДАЛЕН ИЗ КОНТЕЙНЕРА !';
  MyDialog.ShowModal; // Вызываем диалоговое окно
  if MyDialog.RetVol = true then // Если флаг действительный, то:
  begin
    pContForm.DBOLEContainer1.DestroyObject;
    if Assigned(pContForm.DBOLEContainer1.DataSource) then
      pContForm.DBOLEContainer1.DataSource.Edit;
      P_LIST_TB.Refresh; // Обновляем DataSet
    end;
  end;
end;

```

6. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**

Создание формы «Текущая успеваемость групп. Рабочие материалы.»

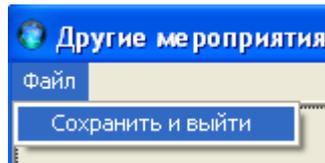
1. Подключите к проекту новую форму, предварительно сохранив ее Unit под именем **Cont** в папке проекта «Журнал». Не забудьте затем в секциях

implementation Unit-ов проекта объявить вновь подключенный Unit (в Main → uses DlgPas, MyDlgInp, Cont, и т.п.).

2. Саму форму **Form1** переименуйте Name → ContForm, Caption → Текущая успеваемость групп. Рабочие материалы.

3. Нанесите на форму компоненты: MainMenu1 и DBOleContainer1.

4. Создайте два пункта меню в соответствии с рисунком. У пункта «Файл» для свойства Name → File1, Caption → &Файл. В пункте «Сохранить и выйти» для свойства Name → SaveOle1, Caption → Сохранить и выйти.



В соответствии с рисунком заполните обработчик событий для пункта «Сохранить и выйти».

```
procedure TContForm.SaveOle1Click(Sender: TObject);
begin
    MainForm.R_LIST_TB.Edit;
    MainForm.R_LIST_TB.Post;
    close;
end;
```

Для формы ContForm в обработчик события onClose внесите код:

```
procedure TContForm.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    MainForm.R_LIST_TB.Edit;
    MainForm.R_LIST_TB.Post;
end;
```

Для компонента DBOleContainer1: Name → DBOleContainer1, Alignment → alClient, DataSource → MainForm.R_LIST_DS, DataField → JOURNAL, Height → 300, Width → 420.

5. Перейти на главную форму MainForm.

Для компонента **INSCont1** (кнопка) в соответствующий обработчик событий внести программный код:

```

procedure TMainForm.INSCont1Click(Sender: TObject);
begin
  if R_LIST_TB.RecordCount <> 0 then // Если имеются записи, то:
  begin
    MyDialog.NfoLab.Font.Color := clBlue;
    MyDialog.NfoLab.Caption := 'ЖУРНАЛ ГРУППЫ ' + R_LIST_TBPGROUP.Value ;
    MyDialog.ShowModal; // Вызываем диалоговое окно
    if MyDialog.RetVol = true then // Если флаг действительный, то:
    begin
      ContForm.ShowModal; // Открываем форму контейнера
      R_LIST_TB.Edit; // Режим редактирования
    end;
    R_LIST_TB.Refresh; // Обновляем DataSet
  end;
end;

```

Для компонента **DelCont1** (кнопка) в соответствующий обработчик событий внести программный код:

```

procedure TMainForm.DelCont1Click(Sender: TObject);
begin
  if R_LIST_TB.RecordCount <> 0 then
  begin
    MyDialog.NfoLab.Font.Color := clRed;
    MyDialog.NfoLab.Caption := 'ЖУРНАЛ ГРУППЫ ' + R_LIST_TBPGROUP.Value +
    ' БУДЕТ УДАЛЕН ИЗ КОНТЕЙНЕРА !';
    MyDialog.ShowModal; // Вызываем диалоговое окно
    if MyDialog.RetVol = true then // Если флаг действительный, то:
    begin
      ContForm.DBOLEContainer1.DestroyObject;
      if Assigned(ContForm.DBOLEContainer1.DataSource) then
        ContForm.DBOLEContainer1.DataSource.Edit;
        R_LIST_TB.Refresh; // Обновляем DataSet
    end;
  end;
end;

```

6. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**.

ФИЛЬТРЫ. ПОЛЯ БЫСТРОГО ПОИСКА

1. Для компонента **FLT_D** (кнопка **F**) в обработчик события **onClick** запишите:

```

procedure TMainForm.FLT_DClick(Sender: TObject);
var
BM : TBookmark; // Объявление локальной переменной класса TBookmark
temp : string;
begin
    BM := D_LIST_TB.GetBookmark; // Получаем позицию закладки

    temp := D_LIST_TBDISCIP.Value;
    if FLT_D.Down then // Если кнопка нажата, то:
    begin
        D_LIST_TB.Filtered := false; // Сбрасываем фильтр
        D_LIST_TB.Filter := 'DISCIP = ' + QuotedStr(temp); // Определяем
                                                    // значение фильтра
        D_LIST_TB.Filtered := true; // Устанавливаем фильтр
    end else // В противном случае
    begin
        D_LIST_TB.Filtered := false; // Сбрасываем фильтр
        edit1.OnChange(nil); // Возвращаем к тонкому фильтру
    end;

    D_LIST_TB.GotoBookmark(BM); // Переходим в позицию закладки
    D_LIST_TB.FreeBookmark(BM); // Очищаем закладку
end;

```

Для компонента Edit1 (поле Быстрого поиска) в обработчик события onChange запишите:

```

procedure TMainForm.Edit1Change(Sender: TObject);
begin
if Length(Edit1.Text) > 0 then //Если вводится текст, то:
begin
    D_LIST_TB.Filtered:=false; // Сбрасываем фильтр
    D_LIST_TB.Filter:= 'DISCIP ' + ' LIKE ' + #39 +
        Edit1.Text + '%' + #39; // Определяем значение фильтра
    D_LIST_TB.Filtered:=true; // Устанавливаем фильтр
end else
    D_LIST_TB.Filtered:=false; // Сбрасываем фильтр
end;

```

2. Для компонента EditF (поле Быстрого поиска) в обработчик события onChange запишите:

```

procedure TMainForm.EditFChange(Sender: TObject);
begin
if Length(EditF.Text) > 0 then //Если вводится текст, то:
begin
    P_LIST_TB.Filtered:=false; // Сбрасываем фильтр
    P_LIST_TB.Filter:= 'PNAME ' + ' LIKE ' + #39
        + EditF.Text + '%' + #39; // Определяем значение фильтра
    P_LIST_TB.Filtered:=true; // Устанавливаем фильтр
end else
    P_LIST_TB.Filtered:=false; // Сбрасываем фильтр
    TransInfo;
end;

```

Для компонента FLT_P (кнопка **F**) в обработчик события onClick запишите:

```
procedure TMainForm.FLT_PClick(Sender: TObject);
var
  BM : TBookmark; // Объявление локальной переменной класса TBookmark
  temp : string;
begin
  BM := P_LIST_TB.GetBookmark; // Получаем позицию закладки

  temp := P_LIST_TB.PNAME.Value;
  if FLT_P.Down then // Если кнопка нажата, то:
  begin
    P_LIST_TB.Filtered := false; // Сбрасываем фильтр
    P_LIST_TB.Filter := 'PNAME = ' + QuotedStr(temp); // Определяем
                                                    // значение фильтра
    P_LIST_TB.Filtered := true; // Устанавливаем фильтр
  end else // В противном случае
  begin
    P_LIST_TB.Filtered := false; // Сбрасываем фильтр
    editF.OnChange(nil); // Возвращаем к тонкому фильтру
  end;

  P_LIST_TB.GotoBookmark(BM); // Переходим в позицию закладки
  P_LIST_TB.FreeBookmark(BM); // Очищаем закладку
  TransInfo;
end;
```

3. Для компонента Edit2 (поле Быстрого поиска) в обработчик события onChange запишите:

```
procedure TMainForm.Edit2Change(Sender: TObject);
begin
  if Length(Edit2.Text) > 0 then //Если вводится текст, то:
  begin
    G_LIST_TB.Filtered:=false; // Сбрасываем фильтр
    G_LIST_TB.Filter:= 'PGROUP ' + ' LIKE ' +
      quotedstr(Edit2.Text + '%');//Определяем значение фильтра
    G_LIST_TB.Filtered:=true; // Устанавливаем фильтр
  end else
  G_LIST_TB.Filtered:=false; // Сбрасываем фильтр
end;
```

Для компонента FLT_G (кнопка **F**) в обработчик события onClick запишите:

```

procedure TMainForm.FLT_GClick(Sender: TObject);
var
BM : TBookmark; // Объявление локальной переменной класса TBookmark
temp : string;
begin
    BM := G_LIST_TB.GetBookmark; // Получаем позицию закладки

    temp := G_LIST_TBPGROUP.Value;
    if FLT_G.Down then // Если кнопка нажата, то:
    begin
        G_LIST_TB.Filtered := false; // Сбрасываем фильтр
        G_LIST_TB.Filter := 'PGROUP = ' + QuotedStr(temp); // Определяем
                                                    // значение фильтра
        G_LIST_TB.Filtered := true; // Устанавливаем фильтр
    end else // В противном случае
    begin
        G_LIST_TB.Filtered := false; // Сбрасываем фильтр
        edit2.OnChange(nil); // Возвращаем к тонкому фильтру
    end;

    G_LIST_TB.GotoBookmark(BM); // Переходим в позицию закладки
    G_LIST_TB.FreeBookmark(BM); // Очищаем закладку
end;

```

4. Быстрый поиск по группам. В обработчик событий onChange компонента FEditGroup запишите программу:

```

procedure TMainForm.FEditGroupChange(Sender: TObject);
begin
    FEditDisc.Text := '';
    if Length(FEditGroup.Text) > 0 then // Если вводится текст, то:
    begin
        M_LIST_TB.Filtered:=false; // Сбрасываем фильтр
        M_LIST_TB.Filter:= 'PNAME = ' + QuotedStr(P_LIST_TB.PNAME.Value) +
        ' AND ' + 'PGROUP ' + ' LIKE ' + #39 + FEditGroup.Text + '%'
                                                    + #39; // Определяем значение фильтра
        M_LIST_TB.Filtered:=true; // Устанавливаем фильтр
    end else
    begin
        M_LIST_TB.Filtered:=false; // Сбрасываем фильтр
        RedFilter;
    end;
end;

```

5. Быстрый поиск по дисциплинам. В обработчик событий onChange компонента FEditDisc запишите программу:

```

procedure TMainForm.FEditDiscChange(Sender: TObject);
begin
    FEditGroup.Text := '';
    if Length(FEditDisc.Text) > 0 then //Если вводится текст, то:
    begin
        M_LIST_TB.Filtered:=false; // Сбрасываем фильтр
        M_LIST_TB.Filter:= 'DISCIP ' + ' LIKE ' + #39 + FEditDisc.Text +
            '%' + #39; // Определяем значение фильтра
        M_LIST_TB.Filtered:=true; // Устанавливаем фильтр
    end else
        M_LIST_TB.Filtered:=false; // Сбрасываем фильтр
    end;

```

Разработка формы «Редактирование академических часов»

1. Подключите к проекту новую форму, предварительно сохранив ее Unit в папке проекта под именем MyDlgED. Не забудьте затем в секции **implementation** Unit-а **Main** объявить вновь подключенный Unit :

```
uses DlgPas, MyDlgInp, MyDlg, MyDlgED;
```

Саму форму **Form1** переименуйте *Name* → MyEDForm, *Caption* → Редактирование академических часов, *Position* → poScreenCenter.

2. Нанесите на форму компоненты:

- **RbButton1**, Name → OKBtn, Caption → Выполнить,

ModalResult → mrOk, Cursor → crHandPoint.

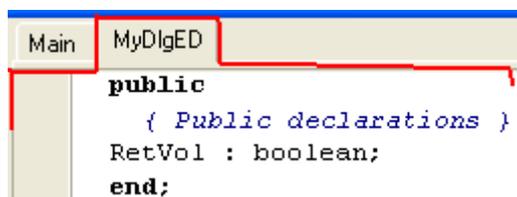
- **RbPanel1** : Align → alTop, Antialiased → False.

Далее на панель (RbPanel1) нанесите компоненты:

- Label: Name → NfoLab, Caption → NfoLab, Transparent → true;

- SpinEdit1: Name → LEC, Caption → Лекционные занятия, MaxValue → 100000;
- SpinEdit2: Name → PRACT, Caption → Практические занятия, MaxValue → 100000;
- SpinEdit3: Name → LAB, Caption → Лабораторные работы, MaxValue → 100000;
- SpinEdit4: Name → KURS, Caption → Курсовое проектир., MaxValue → 100000;
- SpinEdit5: Name → PKR, Caption → Проверка контр. работ, MaxValue → 100000;
- SpinEdit6: Name → KKR, Caption → Комплексные К.Р., MaxValue → 100000;
- SpinEdit7: Name → DIFZ, Caption → Проведение диф.зачета, MaxValue → 100000;
- SpinEdit8: Name → EXAM, Caption → Проведение экзамена, MaxValue → 100000;
- SpinEdit9: Name → DIPLOM, Caption → Рук.диплом.проектом, MaxValue → 100000;
- SpinEdit10: Name → ASPIR, Caption → Рук.аспирантами, MaxValue → 100000;
- SpinEdit11: Name → KONS, Caption → Консультации, MaxValue → 100000;
- SpinEdit12: Name → METHOD, Caption → Методическая работа, MaxValue → 100000.

3. Объявите логическую переменную **RetVol**:



```

Main MyDlgED
public
  { Public declarations }
  RetVol : boolean;
end;

```

4. В обработчике событий **FormActivate** формы MyEDForm присвойте переменной **RetVol** значение ЛОЖЬ.

```

procedure TMyEDForm.FormActivate(Sender: TObject);
begin
  RetVol := false;
end;

```

5. Для кнопки «**Выполнить**» запишите следующую процедуру:

```
procedure TMyEDForm.OKBtnClick(Sender: TObject);  
begin  
    RetVol := true;  
end;
```

Перейдите на главную форму **MainForm** в обработчик событий onClick компонента ADD_M (панель PN3) внесите следующую программу:

```
procedure TMainForm.ADD_MClick(Sender: TObject);  
begin  
    if M_LIST_TB.RecordCount > 0 then  
        begin  
            MyEDForm.NfoLab.Caption := 'Редактирование академич. часов,  
                преподаватель: ' + M_LIST_TBNAME.Value + ', группа/поток: '  
                + M_LIST_TBPGROUP.Value + ' дисциплина: ' + M_LIST_TBDISCIPLINE.Value;  
            MyEDForm.LEC.Value := M_LIST_TBLEC.AsInteger;  
            MyEDForm.PRACT.Value := M_LIST_TBPRACT.AsInteger;  
            MyEDForm.LAB.Value := M_LIST_TBLAB.AsInteger;  
            MyEDForm.KURS.Value := M_LIST_TBKURS.AsInteger;  
            MyEDForm.KONS.Value := M_LIST_TBKONS.AsInteger;  
            MyEDForm.PKR.Value := M_LIST_TBPKR.AsInteger;  
            MyEDForm.KKR.Value := M_LIST_TBKKR.AsInteger;  
            MyEDForm.DIFZ.Value := M_LIST_TBDIFZ.AsInteger;  
            MyEDForm.EXAM.Value := M_LIST_TBEXAM.AsInteger;  
            MyEDForm.DIPLOM.Value := M_LIST_TBDIPL.AsInteger;  
            MyEDForm.ASPIR.Value := M_LIST_TBASPIR.AsInteger;  
            MyEDForm.METHOD.Value := M_LIST_TBMETHOD.AsInteger;  
            MyEDForm.showModal;  
  
            if MyEDForm.RetVol = true then //Если флаг действительный, то:  
                begin  
                    M_LIST_TB.Edit;  
                    M_LIST_TBLEC.Value := MyEDForm.LEC.Value;  
                    M_LIST_TBPRACT.Value := MyEDForm.PRACT.Value;  
                    M_LIST_TBLAB.Value := MyEDForm.LAB.Value;  
                    M_LIST_TBKURS.Value := MyEDForm.KURS.Value;  
                    M_LIST_TBKONS.Value := MyEDForm.KONS.Value;  
                    M_LIST_TBPKR.Value := MyEDForm.PKR.Value;  
                    M_LIST_TBKKR.Value := MyEDForm.KKR.Value;  
                    M_LIST_TBDIFZ.Value := MyEDForm.DIFZ.Value;  
                    M_LIST_TBEXAM.Value := MyEDForm.EXAM.Value;  
                    M_LIST_TBDIPL.Value := MyEDForm.DIPLOM.Value;  
                    M_LIST_TBASPIR.Value := MyEDForm.ASPIR.Value;  
                    M_LIST_TBMETHOD.Value := MyEDForm.METHOD.Value;  
                    M_LIST_TB.Post;  
                end;  
            end;  
        end;  
end;
```

Для возможности удаления записей из M_LIST_Grid (панель PN3) в обработчик событий onClick компонента DEL_M внесите следующую программу:

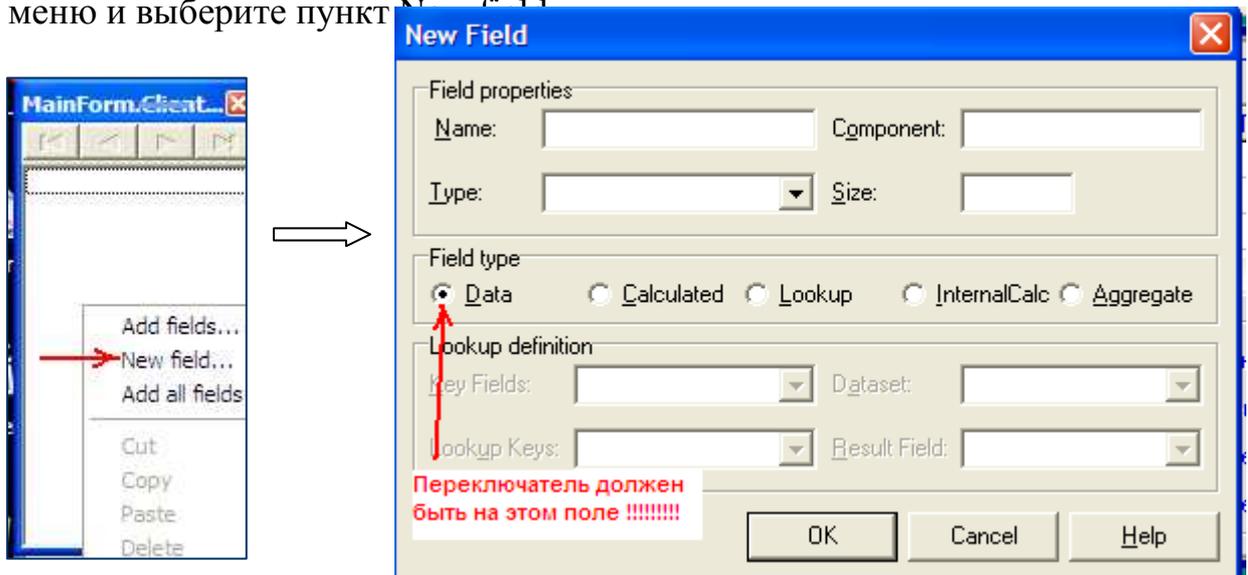
```
procedure TMainForm.DEL_MClick(Sender: TObject);  
begin  
  if M_LIST_TB.RecordCount <> 0 then  
    begin  
      MyDialog.NfoLab.Font.Color := clRed;  
      MyDialog.NfoLab.Caption := 'ЗАПИСЬ: БУДЕТ УДАЛЕНА !';  
      MyDialog.ShowModal;           // Вызываем диалоговое окно  
      if MyDialog.RetVol = true then //Если флаг действительный, то:  
        begin  
          M_LIST_TB.Delete; // Удаляем запись, если не начало таблицы  
          M_LIST_TB.Refresh;           // Обновляем DataSet  
        end;  
      end;  
    end;  
end;
```

6. Запустите приложение, попробуйте заполнить журнал (желательно при первом заполнении позвать преподавателя).

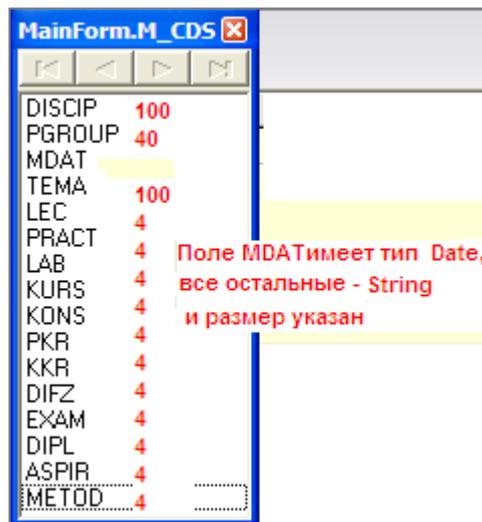
ФОРМИРОВАНИЕ ОТЧЕТОВ

При формировании отчетов связь между его составляющими осуществляется с помощью компонента **ClientDataSet**, который позволяет создать виртуальную связь.

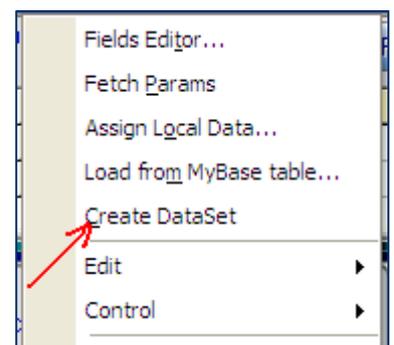
1. Нанесите на главную форму компонент **ClientDataSet1** и переименуйте его Name → **M_CDS**. Выполните двойной щелчок по компоненту **ClientDataSet** вызовите на открывшейся форме контекстное меню и выберите пункт **New Field**



Создайте структуру в соответствии с нижеследующим рисунком.



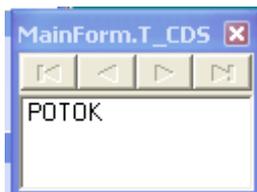
После заполнения формы вызовите контекстное меню для **ClientDataSet** (**M_CDS**) и нажмите пункт **CreateDataSet** (создается доменная структура).



У **ClientDataSet** (M_CDS) свойство **Active** должно быть **true**.

2. Нанесите на форму следующий компонент **ClientDataSet1** и переименуйте его Name → **T_CDS**. Выполните двойной щелчок по компоненту **T_CDS (ClientDataSet)**, на открывшейся форме вызовите контекстное меню и выберите пункт **New field**.

Создайте структуру в соответствии с нижеследующим рисунком. Поле поток имеет тип **String**, а размер поля **Size = 50**.

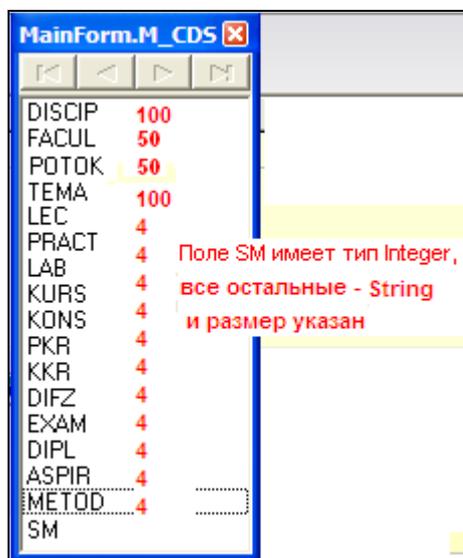


После заполнения формы вызовите контекстное меню для **ClientDataSet** (**T_CDS**) и нажмите пункт **CreateDataSet** (создается доменная структура).

У **ClientDataSet** (**T_CDS**) свойство **Active** должно быть **true**.

3. Нанесите на форму еще один (третий) компонент **ClientDataSet1** и переименуйте его Name → **U_CDS**. Выполните двойной щелчок по компоненту **U_CDS (ClientDataSet)**, на открывшейся форме вызовите контекстное меню и выберите пункт **New field**.

Создайте структуру в соответствии с нижеследующим рисунком.

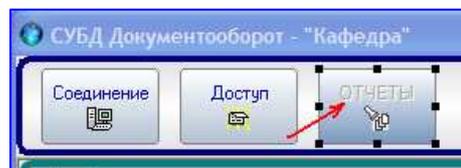


После заполнения формы вызовите контекстное меню для **ClientDataSet** (**U_CDS**) и нажмите пункт **CreateDataSet** (создается доменная структура).

У **ClientDataSet** (**U_CDS**) свойство **Active** должно быть **true**.

4. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll**.

5. Для доступности кнопок (они расположены на панелях SelPanel и RepPanel), вызывающих процедуры формирования отчетов, для кнопки



«ОТЧЕТЫ» (кнопка UREPBTN) запишите следующую процедуру:

```
procedure TMainForm.UrepBTNClick(Sender: TObject);  
begin  
    SelPanel.Enabled := UREPBTN.Down;  
    RepPanel.Enabled := UREPBTN.Down;  
    if UREPBTN.Down then  
        begin  
            PN2.Visible := false;  
            PN1.Visible := false;  
        end else  
        begin  
            PN1.Visible := true;  
            PN2.Visible := true;  
        end;  
end;
```

2. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**

3. Выбор периода (с какой даты по какую) для составления отчета предоставляется после нажатия кнопки «ВЫБРАТЬ» (кнопка FLTRBTN). Запишите в ее обработчик событий следующую программу:

```
procedure TMainForm.FLTRBTNClick(Sender: TObject);  
begin  
    if FLTRBTN.Down then  
        begin  
            BeginDTP.Enabled := false;  
            EndDTP.Enabled := false;  
            SelFilter;  
            SelRepBtn.Enabled := true;  
            SelPerBtn.Enabled := true;  
        end else  
        begin  
            M_LIST_TB.Filtered := false;  
            BeginDTP.Enabled := true;  
            EndDTP.Enabled := true;  
            SelRepBtn.Enabled := false;  
            SelPerBtn.Enabled := false;  
        end;  
end;
```

4. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**.

5. После выбора отчетного периода можно вызывать форму с отчетом за указанный период (Отчет по периодам). Создадим форму, отображающую отчет преподавателя о проделанной работе за конкретно выбранный период.

СВОДНЫЙ ОТЧЕТ ЗА ГОД

(кнопка Отчет№1, компонент AllRepBtn)

Создание формы с отчетом «ВЫПОЛНЕНИЕ ИНДИВИДУАЛЬНОГО ПЛАНА»

1. Подключите к проекту новую отчет-форму, (File → New → Other → Report) сохранив ее **Unit** (команда File → SaveAs) под именем **ALLRep** в папке проекта. Не забудьте затем в секциях **implementation** Unit-ов **Main** и **ALLRep** объявить вновь подключенные Unit-ы.

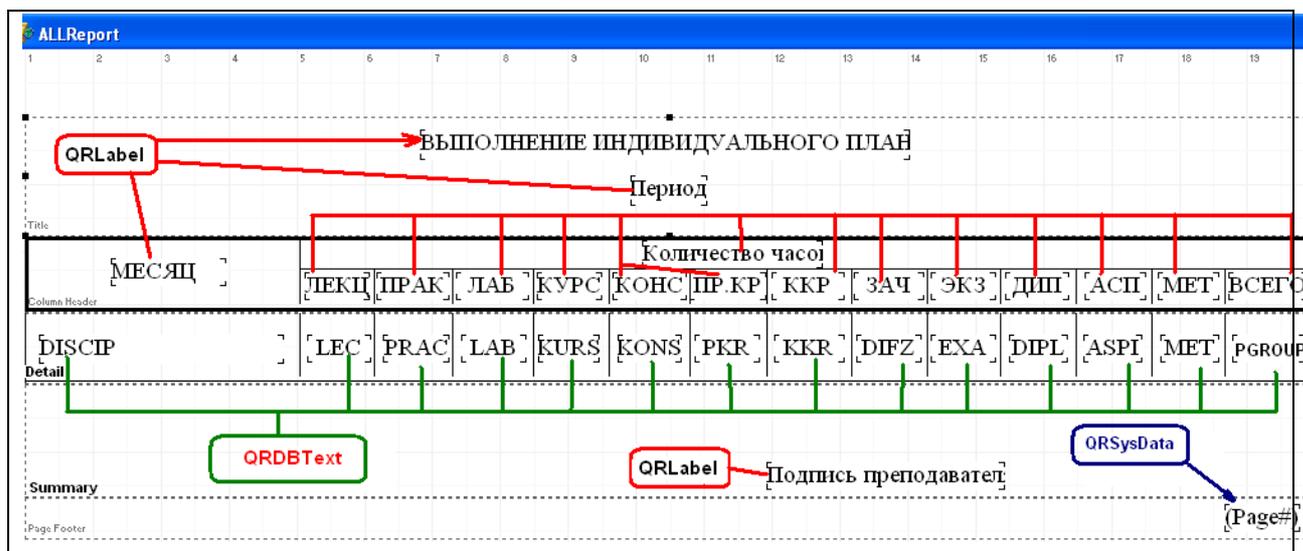
2. Саму форму (компонент **QuickReport1**) переименуйте *Name* → **ALLReport**, *DataSet* → **MainForm.M_CDS**.

3. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File** → **SaveAll**.

4. Установите на отчете следующие полосы

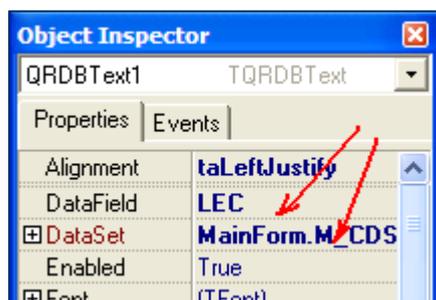
| | |
|-----------------|-----------------------|
| ☐ Bands | (TQuickRepBar |
| HasColumnHeader | True |
| HasDetail | True |
| HasPageFooter | True |
| HasPageHeader | False |
| HasSummary | True |
| HasTitle | True |
| Cursor | crDefault |
| ☐ DataSet | MainForm.M_CDS |

5. На полосах разместите компоненты со странички Qreport : QRLabel, QRDBText, QRSysData., QRShape.



У QRLabel «Период» значение свойства Name → PeriodLB. У компонента QRSysData1 для свойства Data → qrsPageNumber.

6. Компоненты QRDBText свяжите с соответствующими полями:



ОТЧЕТ ПО ДИСЦИПЛИНАМ

(кнопка Отчет№2, компонент SelPerBtn)

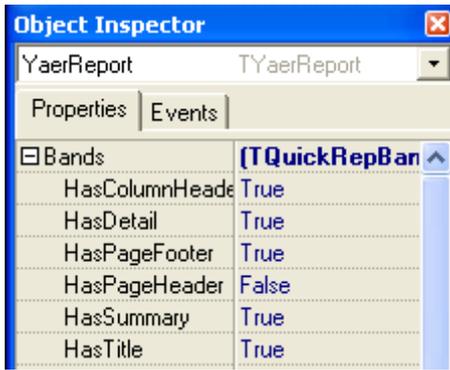
Создание формы с отчетом «УЧЕБНАЯ РАБОТА»

1. Подключите к проекту новую отчет-форму, (File → New → Other → Report) предварительно сохранив ее **Unit** под именем **YerRep** в папке проекта. Не забудьте затем в секциях **implementation** Unit-ов **Main** и **YerRep** объявить вновь подключенные Unit-ы.

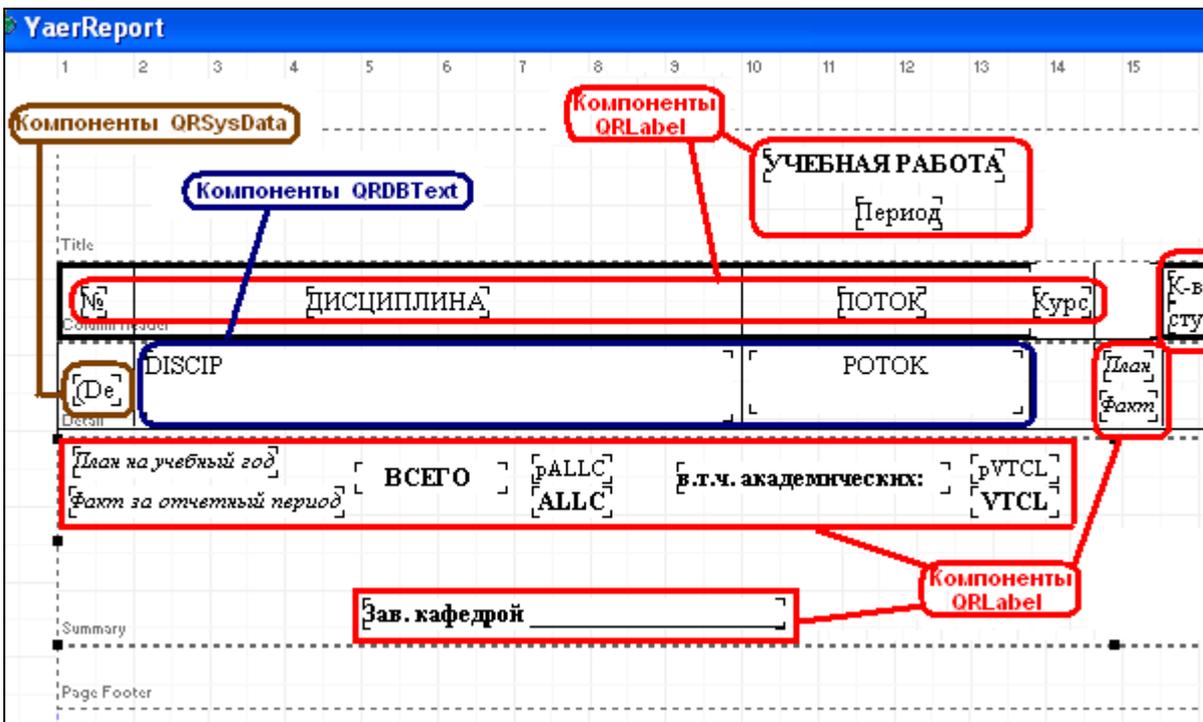
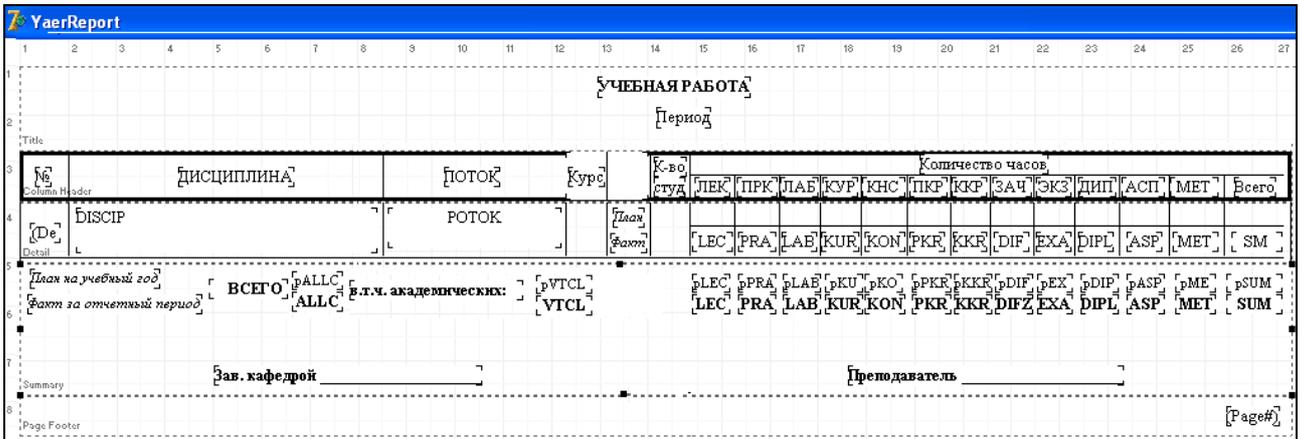
2. Саму форму (компонент **QuickReport1**) переименуйте **Name** → **YaeReport**, **DataSet** → **MainForm.U_CDS**.

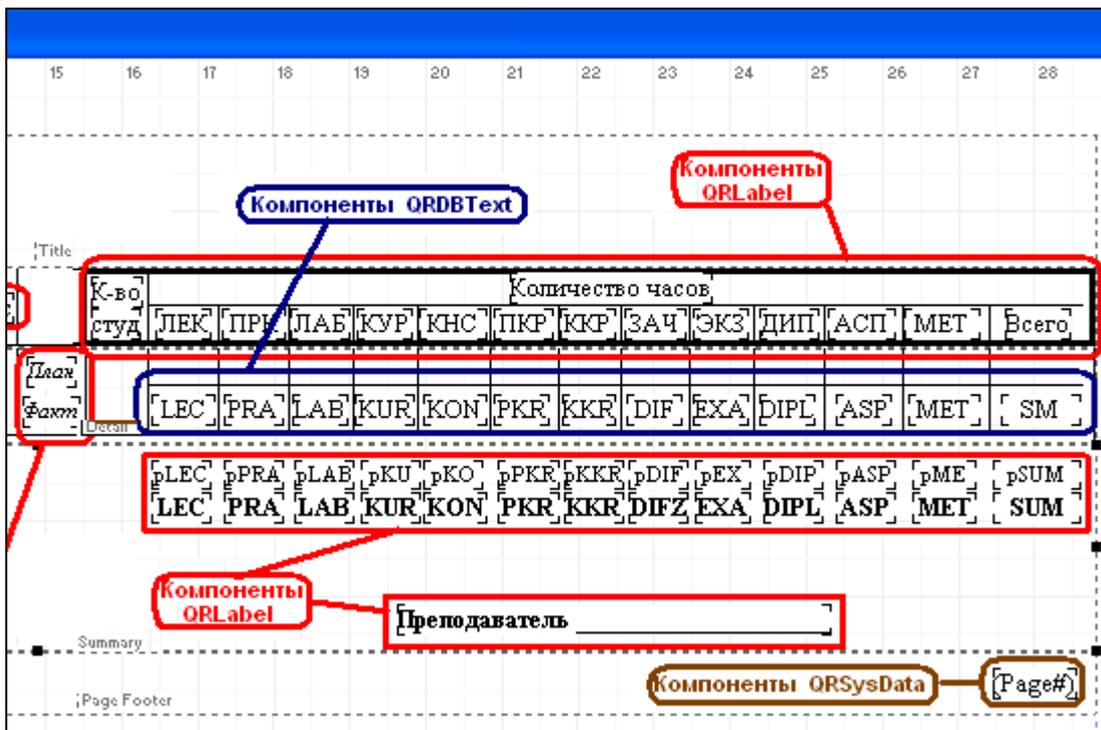
3. Скомпилируйте приложение, проверьте его работу и после его остановки досохраниите проект командой **File** → **SaveAll**.

4. Установите на отчете следующие полосы



5. На полосах разместите компоненты со странички QReport : QRLabel, QRDBText, QRSysData., QRShape.





6. Для компонентов QRLabel (на полосе **Summary**) с надписями на английском языке (LEC, PRA ...) значение свойства **Name** должно соответствовать названию полей в таблице компонента доступа к данным **U_CDS** (см. рис.), Для надписей pLEC, pPRA к названию в рис. добавляется префикс **p**, т.е. у QRLabel Caption → «pLEC», значит Name → pLEC; Caption → «pPRA», значит Name → pPRACT и т. д.



У QRLabel «Период» значение свойства Name → PeriodLB. Для компонентов QRLabel, у которых значение свойства Caption:

- «ALLC», Name → ALLCLOCK;
- «pALLC», Name → pALLCLOCK;

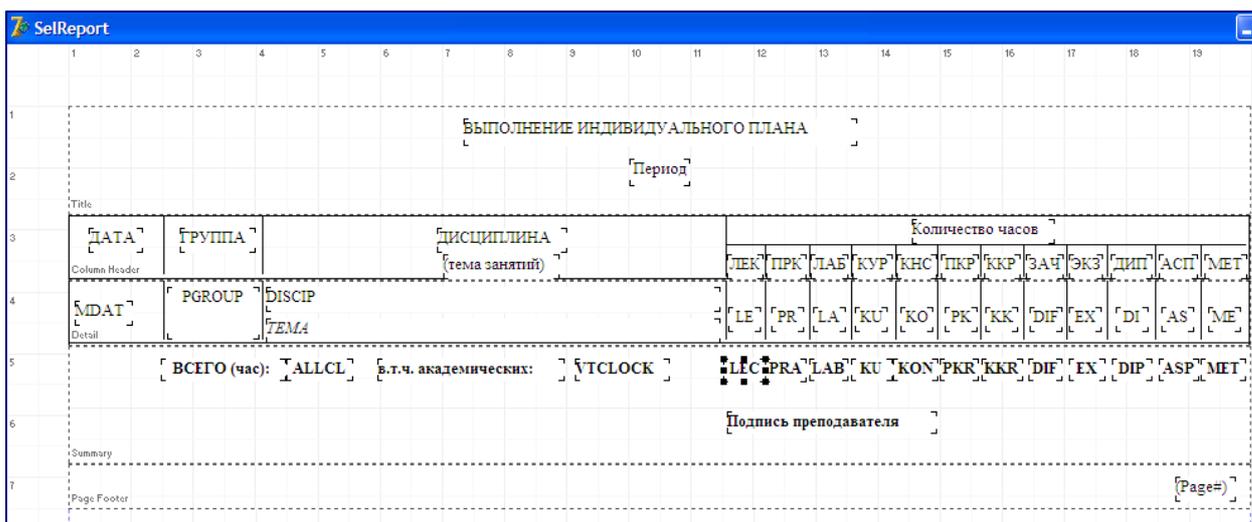
- «VTCL», Name → VTCLOCK;
- «pVTCL», Name → pVTCLOCK;
- 7. Компоненты QRDBText свяжите с соответствующими полями:



ОТЧЕТ ПО ПЕРИОДАМ

(кнопка Отчет№3, компонент SelRepBtn)

Создание формы с отчетом «ВЫПОЛНЕНИЕ ИНДИВИДУАЛЬНОГО ПЛАНА»

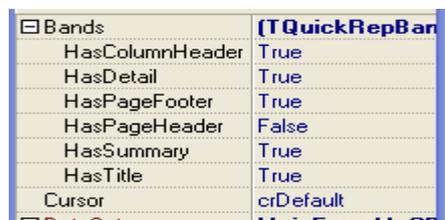


1. Подключите к проекту новую отчет-форму, (File → New → Other → Report) сохранив ее **Unit** (команда File → SaveAs) под именем **SelRep** в папке проекта. Не забудьте затем в секциях **implementation** Unit-ов **Main** и **SelRep** объявить вновь подключенные Unit-ы.

2. Саму форму (компонент **QuickReport1**) переименуйте *Name* → SelReport, *DataSet* → MainForm.M_CDS.

3. Скомпилируйте приложение, проверьте его работу и после его остановки сохраните проект командой **File → SaveAll**.

4. Установите на отчете следующие ПОЛОСЫ



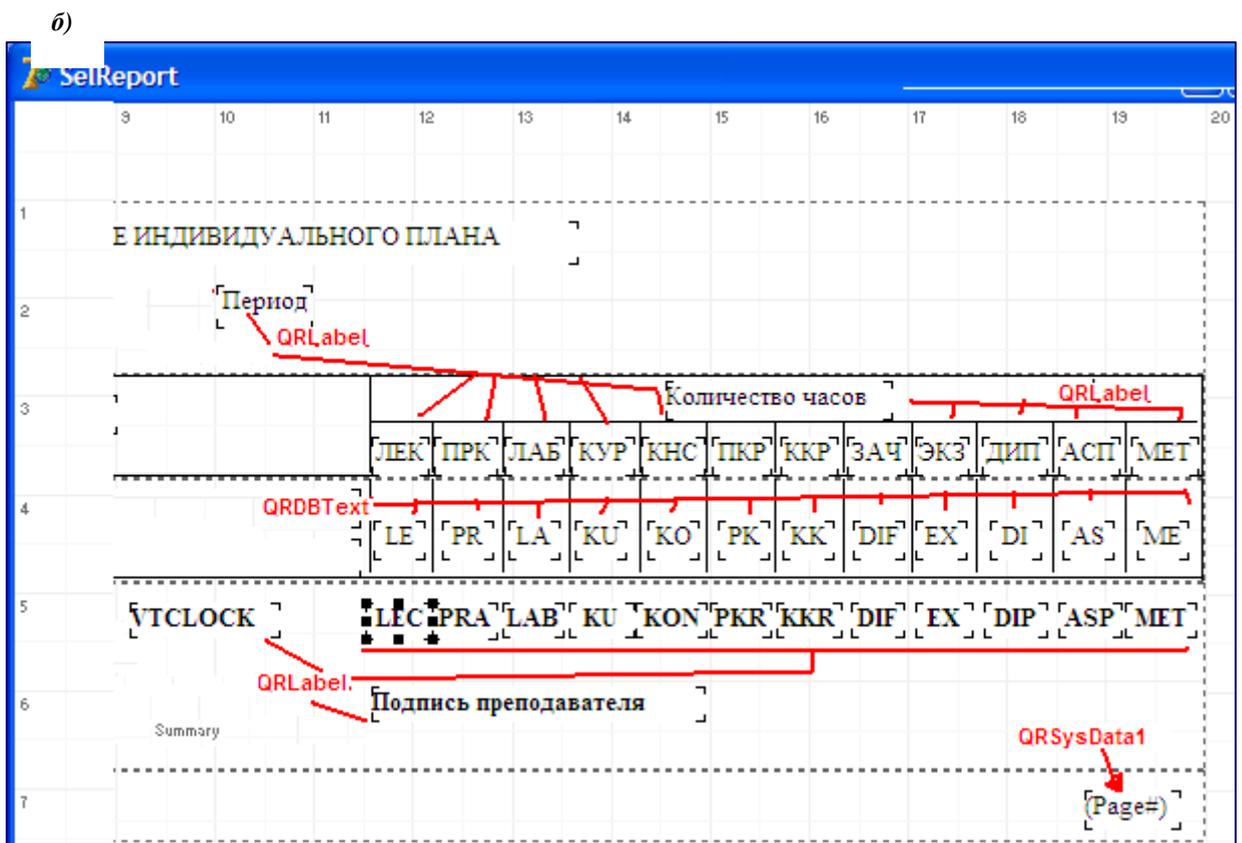
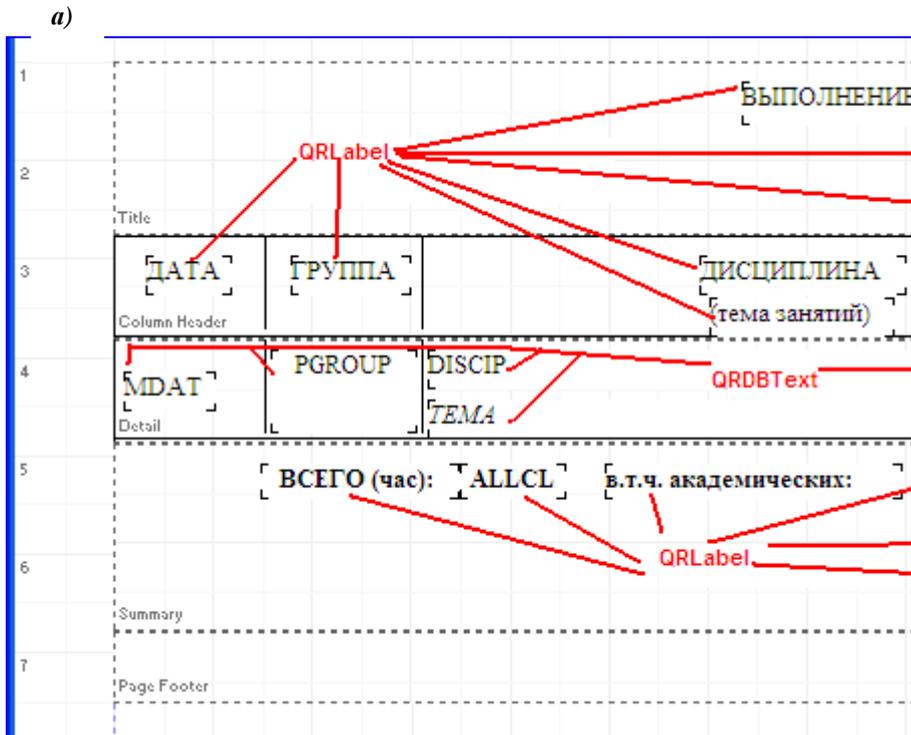


Рис. «Отчет»

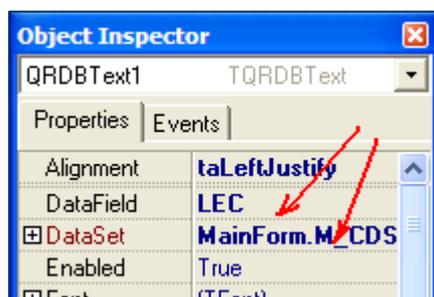
5. На полосах разместите компоненты со странички Qreport : QRLabel, QRDBText, QRSysData., QRShape в соответствии с рис. «Отчет (а) и б) »

6. Для компонентов QRLabel (на полосе **Summary**) с надписями на англ. языке (LEC, PRA ...) значение свойства **Name** должно соответствовать названию полей в таблице компонента доступа к данным **M_CDS** (см. рис.), т.е. у QRLabel Caption → «LEC» Name → LEC; Caption → «PRA», значит Name → PRACT и т. д.



У QRLabel «Период» Name → PeriodLB, у QRLabel с надписью Caption → ALLCL - Name → ALLCLOCK.

7. Компоненты QRDBText свяжите с соответствующими полями:



ПРОГРАММНЫЕ КОДЫ В ОБРАБОТЧИК СОБЫТИЙ OnClick ДЛЯ КОМПОНЕНТОВ: AllRepBtn (кнопка «Отчет №1»), SelPerBtn (кнопка «Отчет №2»), SelRepBtn (кнопка «Отчет №3»)

Для кнопки «Отчет №1» (компонент)) в обработчик события onClick

запишите:

```

procedure TMainForm.AllRepBtnClick(Sender: TObject);
var
  i: Integer;
  BN, EN: String;
  MSUM: Integer;
begin
  FEditGroup.Text := '';
  FEditDisc.Text := '';
  BN := IntToStr(BNSpinEdit.Value); EN := IntToStr(ENSpinEdit.Value);

  M_CDS.Close;
  M_CDS.CreateDataSet;
  M_CDS.Open;
  
```

```

{===== СЕНТЯБРЬ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.09.' + BN + ' AND ' +
'MDAT < ' + '01.10.' + BN + ' AND ' + 'PNAME = '
+ QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0; KKR := 0;
DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.Value;
PRACT := PRACT + M_LIST_TB.PRACT.Value;
LAB := LAB + M_LIST_TB.LAB.Value;
KURS := KURS + M_LIST_TB.KURS.Value;
KONS := KONS + M_LIST_TB.KONS.Value;
PKR := PKR + M_LIST_TB.PKR.Value;
KKR := KKR + M_LIST_TB.KKR.Value;
DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
EXAM := EXAM + M_LIST_TB.EXAM.Value;
DIPL := DIPL + M_LIST_TB.DIPL.Value;
ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
METOD := METOD + M_LIST_TB.METOD.Value;
MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
EXAM + DIPL + ASPIR + METOD;
M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'СЕНТЯБРЬ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

{===== ОКТЯБРЬ =====}

```

```

{===== ОКТЯБРЬ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.10.' + BN + ' AND ' +
'MDAT < ' + '01.11.' + BN + ' AND ' + 'PNAME = ' +
  QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METHOD := METHOD + M_LIST_TB.METHOD.Value;
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ + EXAM +
  DIPL + ASPIR + METHOD;
  M_LIST_TB.Next;
end;
M_CDS.Append;
M_CDS.DISCIP.Value := 'ОКТЯБРЬ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

{===== НОЯБРЬ =====}

```

```

{===== НОЯБРЬ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.11.' + BN + ' AND ' +
'MDAT < ' + '01.12.' + BN + ' AND ' + 'PNAME = ' +
  QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METHOD := METHOD + M_LIST_TB.METHOD.Value;
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ + EXAM +
  DIPL + ASPIR + METHOD;
  M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'НОЯБРЬ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

(===== ДЕКАБРЬ =====)
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.12.' + BN + ' AND ' +
'MDAT < ' + '01.01.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0; METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.Value;
PRACT := PRACT + M_LIST_TB.PRACT.Value;
LAB := LAB + M_LIST_TB.LAB.Value;
KURS := KURS + M_LIST_TB.KURS.Value;
KONS := KONS + M_LIST_TB.KONS.Value;
PKR := PKR + M_LIST_TB.PKR.Value;
KKR := KKR + M_LIST_TB.KKR.Value;
DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
EXAM := EXAM + M_LIST_TB.EXAM.Value;
DIPL := DIPL + M_LIST_TB.DIPL.Value;
ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
METOD := METOD + M_LIST_TB.METOD.Value;
MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ + EXAM +
DIPL + ASPIR + METOD;
M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'ДЕКАБРЬ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC);
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

(===== RHBAPb =====)
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.01.' + EN + ' AND ' + 'MDAT < ' +
'01.02.' + EN + ' AND ' + 'PNAME = ' + QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0; KKR := 0;
DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.Value;
PRACT := PRACT + M_LIST_TB.PRACT.Value;
LAB := LAB + M_LIST_TB.LAB.Value;
KURS := KURS + M_LIST_TB.KURS.Value;
KONS := KONS + M_LIST_TB.KONS.Value;
PKR := PKR + M_LIST_TB.PKR.Value;
KKR := KKR + M_LIST_TB.KKR.Value;
DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
EXAM := EXAM + M_LIST_TB.EXAM.Value;
DIPL := DIPL + M_LIST_TB.DIPL.Value;
ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
METHOD := METHOD + M_LIST_TB.METHOD.Value;
MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ + EXAM +
DIPL + ASPIR + METHOD;
M_LIST_TB.Next;
end;

M_CDS.Append;
M_CSDISCIP.Value := 'RHBAPb';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

{===== MTOFO: 3A 1 CEMECTP =====}
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.09.' + BN + ' AND ' +
'MDAT < ' + '01.02.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0; KKR := 0;
DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.Value;
PRACT := PRACT + M_LIST_TB.PRACT.Value;
LAB := LAB + M_LIST_TB.LAB.Value;
KURS := KURS + M_LIST_TB.KURS.Value;
KONS := KONS + M_LIST_TB.KONS.Value;
PKR := PKR + M_LIST_TB.PKR.Value;
KKR := KKR + M_LIST_TB.KKR.Value;
DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
EXAM := EXAM + M_LIST_TB.EXAM.Value;
DIPL := DIPL + M_LIST_TB.DIPL.Value;
ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
METOD := METOD + M_LIST_TB.METOD.Value;
MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
EXAM + DIPL + ASPIR + METOD;
M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'MTOFO: 3A 1 CEMECTP';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.GROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

    {===== ФЕРМАЛЬ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.02.' + EN + ' AND ' +
'MDAT < ' + '01.03.' + EN + ' AND ' + 'PNAME = ' +
  QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METOD := METOD + M_LIST_TB.METOD.Value;
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
  EXAM + DIPL + ASPIR + METOD;
  M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'ФЕРМАЛЬ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

    {===== MAPT =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.03.' + EN + ' AND ' +
'MDAT < ' + '01.04.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
    LEC := LEC + M_LIST_TB.LEC.Value;
    PRACT := PRACT + M_LIST_TB.PRACT.Value;
    LAB := LAB + M_LIST_TB.LAB.Value;
    KURS := KURS + M_LIST_TB.KURS.Value;
    KONS := KONS + M_LIST_TB.KONS.Value;
    PKR := PKR + M_LIST_TB.PKR.Value;
    KKR := KKR + M_LIST_TB.KKR.Value;
    DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
    EXAM := EXAM + M_LIST_TB.EXAM.Value;
    DIPL := DIPL + M_LIST_TB.DIPL.Value;
    ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
    METHOD := METHOD + M_LIST_TB.METHOD.Value;
    MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
        EXAM + DIPL + ASPIR + METHOD;
    M_LIST_TB.Next;
end;
M_CDS.Append;
M_CDS.DISCIP.Value := 'MAPT';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

{===== АНПЕЖБ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.04.' + EN + ' AND ' +
'MDAT < ' + '01.05.' + EN + ' AND ' + 'PNAME = ' +
  QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.Value;
PRACT := PRACT + M_LIST_TB.PRACT.Value;
LAB := LAB + M_LIST_TB.LAB.Value;
KURS := KURS + M_LIST_TB.KURS.Value;
KONS := KONS + M_LIST_TB.KONS.Value;
PKR := PKR + M_LIST_TB.PKR.Value;
KKR := KKR + M_LIST_TB.KKR.Value;
DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
EXAM := EXAM + M_LIST_TB.EXAM.Value;
DIPL := DIPL + M_LIST_TB.DIPL.Value;
ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
METOD := METOD + M_LIST_TB.METOD.Value;
MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ + EXAM +
DIPL + ASPIR + METOD;
M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'АНПЕЖБ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

      {===== МАЙ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.05.' + EN + ' AND ' +
'MDAT < ' + '01.06.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TB.PNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METOD := METOD + M_LIST_TB.METOD.Value;
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
  EXAM + DIPL + ASPIR + METOD;
  M_LIST_TB.Next;
end;
M_CDS.Append;
M_CDS.DISCIP.Value := 'МАЙ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.GROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

      {===== MOHB =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.06.' + EN + ' AND ' +
'MDAT < ' + '01.07.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METHOD := METHOD + M_LIST_TB.METHOD.Value;
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
  EXAM + DIPL + ASPIR + METHOD;
  M_LIST_TB.Next;
end;
M_CDS.Append;
M_CDS.DISCIP.Value := 'MOHB';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

      {===== МОЖЬ =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.07.' + EN + ' AND ' +
'MDAT < ' + '01.08.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METHOD := METHOD + M_LIST_TB.METHOD.Value;
  M_CDSPGROUP.Value := IntToStr(MSUM);
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
  EXAM + DIPL + ASPIR + METHOD;
  M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'МОЖЬ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC);
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.CDSPGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

    {===== ABFYCT =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.08.' + EN + ' AND ' +
'MDAT < ' + '01.09.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
    LEC := LEC + M_LIST_TB.LEC.Value;
    PRACT := PRACT + M_LIST_TB.PRACT.Value;
    LAB := LAB + M_LIST_TB.LAB.Value;
    KURS := KURS + M_LIST_TB.KURS.Value;
    KONS := KONS + M_LIST_TB.KONS.Value;
    PKR := PKR + M_LIST_TB.PKR.Value;
    KKR := KKR + M_LIST_TB.KKR.Value;
    DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
    EXAM := EXAM + M_LIST_TB.EXAM.Value;
    DIPL := DIPL + M_LIST_TB.DIPL.Value;
    ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
    METHOD := METHOD + M_LIST_TB.METHOD.Value;
    MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
    EXAM + DIPL + ASPIR + METHOD;
    M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIPLIN.Value := 'ABFYCT';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then M_CDS.METHOD.Value := IntToStr(METHOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

      {===== MTOFO 3A 2 CEMECTP =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.02.' + EN + ' AND ' +
'MDAT < ' + '01.09.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
  LEC := LEC + M_LIST_TB.LEC.Value;
  PRACT := PRACT + M_LIST_TB.PRACT.Value;
  LAB := LAB + M_LIST_TB.LAB.Value;
  KURS := KURS + M_LIST_TB.KURS.Value;
  KONS := KONS + M_LIST_TB.KONS.Value;
  PKR := PKR + M_LIST_TB.PKR.Value;
  KKR := KKR + M_LIST_TB.KKR.Value;
  DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
  EXAM := EXAM + M_LIST_TB.EXAM.Value;
  DIPL := DIPL + M_LIST_TB.DIPL.Value;
  ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
  METOD := METOD + M_LIST_TB.METOD.Value;
  MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR + DIFZ +
  EXAM + DIPL + ASPIR + METOD;
  M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'MTOFO: 3A 2 CEMECTP ';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

```

```

    {===== ИТОГО ЗА УЧЕБНЫЙ ГОД =====}
MSUM := 0;
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + '01.09.' + BN + ' AND ' +
'MDAT < ' + '01.09.' + EN + ' AND ' + 'PNAME = ' +
QuotedStr(P_LIST_TBPNAME.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
M_LIST_TB.First;
for i := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.Value;
PRACT := PRACT + M_LIST_TB.PRACT.Value;
LAB := LAB + M_LIST_TB.LAB.Value;
KURS := KURS + M_LIST_TB.KURS.Value;
KONS := KONS + M_LIST_TB.KONS.Value;
PKR := PKR + M_LIST_TB.PKR.Value;
KKR := KKR + M_LIST_TB.KKR.Value;
DIFZ := DIFZ + M_LIST_TB.DIFZ.Value;
EXAM := EXAM + M_LIST_TB.EXAM.Value;
DIPL := DIPL + M_LIST_TB.DIPL.Value;
ASPIR := ASPIR + M_LIST_TB.ASPIR.Value;
METOD := METOD + M_LIST_TB.METOD.Value;
MSUM := LEC + PRACT + LAB + KURS + KONS + PKR + KKR +
DIFZ + EXAM + DIPL + ASPIR + METOD;
M_LIST_TB.Next;
end;

M_CDS.Append;
M_CDS.DISCIP.Value := 'ИТОГО: ЗА ГОД';
if LEC > 0 then M_CDS.LEC.Value := IntToStr(LEC) ;
if PRACT > 0 then M_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then M_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then M_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then M_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then M_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then M_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then M_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then M_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then M_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then M_CDS.ASPIR.Value := IntToStr(ASPIR);
if METOD > 0 then M_CDS.METOD.Value := IntToStr(METOD);
if MSUM > 0 then M_CDS.PGROUP.Value := IntToStr(MSUM);
M_CDS.Post;

M_LIST_TB.First;
AllReport.PeriodLB.Caption := P_LIST_TBJOB.Value + ' ' +
P_LIST_TBPNAME.Value;
AllReport.Preview;
end;

```

Для кнопки «Отчет №2» (компонент SelPerBtn) в обработчик события

onClick запишите:

```
procedure TMainForm.SelPerBtnClick(Sender: TObject);
var
i, j, sm, sSUM, sLEC, sPRACT, sLAB, sKURS, sKONS,
sPKR, sKKR, sDIFZ, sEXAM, sDIPL, sASPIR, sMETOD: integer;
BG, EN: String;
pSUM, pSM: Integer;
begin
    FEditGroup.Text := '';
    FEditDisc.Text := '';

    sSum := 0; sLEC := 0; sPRACT := 0; sLAB := 0; sKONS := 0;
    sKURS := 0; sPKR:= 0; sKKR := 0; sDIFZ := 0; sEXAM := 0;
    sDIPL := 0; sASPIR := 0; sMETOD := 0;

    BG := DateToStr(BeginDTP.Date);
    EN := DateToStr(EndDTP.Date);
    {Создание главного представления выборки дисциплин по потокам}
    T_CDS.Close; // закрыть представление
    T_CDS.CreateDataSet; // создать структуру представления
    T_CDS.Open; // открыть представление
    if M_LIST_TB.RecordCount > 0 then // Если к-во записей в
        //таблице выборки > 0 то:

    begin
    M_LIST_TB.First; // Переходим к первой записи
        for i := 1 to M_LIST_TB.RecordCount do // В цикле от 1 до
            //конца таблицы

    begin
    T_CDS.Filtered := false; // Выключаем фильтр
        //в главном представлении
    T_CDS.Filter := 'ПОТОК = ' + QuotedStr(M_LIST_TB.PОТОК.Value); //Определяем
        //значение фильтра
    T_CDS.Filtered := true; // Включаем фильтр в главном представлении
    if T_CDS.RecordCount = 0 then // Если к-во записей = 0 (нет потока)
    begin
        T_CDS.Append; // Добавляем запись
        T_CDSPОТОК.Value := M_LIST_TB.PОТОК.Value; // Добавляем значение
            //нового потока
        T_CDS.Post; // Записываем данные
    end;
    T_CDS.Filtered := false; // Сбрасываем фильтр
    M_LIST_TB.Next; // Переходим к следующей записи таблицы выборки
    end;
    M_LIST_TB.First;
    end;
    {=====}
```

```

U_CDS.Close;      // закрыть представление
U_CDS.CreateDataSet; // создать структуру представления
U_CDS.Open;      // открыть представление
T_CDS.First;

for i := 1 to T_CDS.RecordCount do
begin
M_LIST_TB.Filtered := false;
M_LIST_TB.Filter := 'MDAT >= ' + BG + ' AND ' + 'MDAT <='
+ EN + ' AND ' + 'POTOK = ' + QuotedStr(T_CDSPOTOK.Value);
M_LIST_TB.Filtered := true;
LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0;
KKR := 0; DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METHOD := 0;

for j := 1 to M_LIST_TB.RecordCount do
begin
LEC := LEC + M_LIST_TB.LEC.AsInteger;
PRACT := PRACT + M_LIST_TB.PRACT.AsInteger;
LAB := LAB + M_LIST_TB.LAB.AsInteger;
KURS := KURS + M_LIST_TB.KURS.AsInteger;
KONS := KONS + M_LIST_TB.KONS.AsInteger;
PKR := PKR + M_LIST_TB.PKR.AsInteger;
KKR := KKR + M_LIST_TB.KKR.AsInteger;
DIFZ := DIFZ + M_LIST_TB.DIFZ.AsInteger;
EXAM := EXAM + M_LIST_TB.EXAM.AsInteger;
DIPL := DIPL + M_LIST_TB.DIPL.AsInteger;
ASPIR := ASPIR + M_LIST_TB.ASPIR.AsInteger;
METHOD := METHOD + M_LIST_TB.METHOD.AsInteger;
M_LIST_TB.Next;
end;

sm := LEC + PRACT + LAB + KURS + KONS + PKR + KKR +
      DIFZ + EXAM + DIPL + ASPIR + METHOD;

U_CDS.Append;
U_CDSPOTOK.Value := T_CDSPOTOK.Value;
U_CDSDISCIP.Value := M_LIST_TB.DISCIP.Value;
if LEC > 0 then U_CDS.LEC.Value := IntToStr(LEC);
if PRACT > 0 then U_CDS.PRACT.Value := IntToStr(PRACT);
if LAB > 0 then U_CDS.LAB.Value := IntToStr(LAB);
if KURS > 0 then U_CDS.KURS.Value := IntToStr(KURS);
if KONS > 0 then U_CDS.KONS.Value := IntToStr(KONS);
if PKR > 0 then U_CDS.PKR.Value := IntToStr(PKR);
if KKR > 0 then U_CDS.KKR.Value := IntToStr(KKR);
if DIFZ > 0 then U_CDS.DIFZ.Value := IntToStr(DIFZ);
if EXAM > 0 then U_CDS.EXAM.Value := IntToStr(EXAM);
if DIPL > 0 then U_CDS.DIPL.Value := IntToStr(DIPL);
if ASPIR > 0 then U_CDS.ASPIR.Value := IntToStr(ASPIR);
if METHOD > 0 then U_CDS.METHOD.Value := IntToStr(METHOD);

```

```

U_CDSSM.Value := sm;
U_CDS.Post;
sSUM := sSum + sm;
sLEC := sLEC + LEC; sPRACT := sPRACT + PRACT; sLAB := sLAB + LAB;
sKURS := sKURS + KURS; sKONS := sKONS + KONS; sPKR:= sPKR + PKR;
sKKR := sKKR + KKR; sDIFZ := sDIFZ + DIFZ; sEXAM := sEXAM + EXAM;
sDIPL := sDIPL + DIPL; sASPIR := sASPIR + ASPIR;
sMETOD := sMETOD + METHOD;

T_CDS.Next;
end;
U_CDS.IndexFieldNames := 'DISCIP';

YaerReport.SUM.Caption := IntToStr(sSum);
YaerReport.LEC.Caption := IntToStr(sLEC);
YaerReport.PRACT.Caption := IntToStr(sPRACT);
YaerReport.LAB.Caption := IntToStr(sLAB);
YaerReport.KURS.Caption := IntToStr(sKURS);
YaerReport.KONS.Caption := IntToStr(sKONS);
YaerReport.PKR.Caption := IntToStr(sPKR);
YaerReport.KKR.Caption := IntToStr(sKKR);
YaerReport.DIFZ.Caption := IntToStr(sDIFZ);
YaerReport.EXAM.Caption := IntToStr(sEXAM);
YaerReport.DIPL.Caption := IntToStr(sDIPL);
YaerReport.ASPIR.Caption := IntToStr(sASPIR);
YaerReport.METOD.Caption := IntToStr(sMETOD);
YaerReport.ALLCLOCK.Caption := IntToStr(sSUM);
YaerReport.VTCLOCK.Caption := IntToStr(sSUM - sMETOD);
YaerReport.pLEC.Caption := P_LIST_TBLEK.AsString;
YaerReport.pPRACT.Caption := P_LIST_TBPRACT.AsString;
YaerReport.pLAB.Caption := P_LIST_TBLAB.AsString;
YaerReport.pKURS.Caption := P_LIST_TBKURS.AsString;
YaerReport.pKONS.Caption := P_LIST_TBKONS.AsString;
YaerReport.pPKR.Caption := P_LIST_TBPKR.AsString;
YaerReport.pKKR.Caption := P_LIST_TBKKR.AsString;
YaerReport.pDIFZ.Caption := P_LIST_TBDIFZ.AsString;
YaerReport.pEXAM.Caption := P_LIST_TBEXAM.AsString;
YaerReport.pDIPL.Caption := P_LIST_TBDIPL.AsString;
YaerReport.pASPIR.Caption := P_LIST_TBASPIR.AsString;
YaerReport.pMETOD.Caption := P_LIST_TBMETOD.AsString;

pSUM := P_LIST_TBLEK.Value + P_LIST_TBPRACT.Value + P_LIST_TBLAB.Value +
P_LIST_TBKURS.Value + P_LIST_TBKONS.Value + P_LIST_TBPKR.Value +
P_LIST_TBKKR.Value + P_LIST_TBDIFZ.Value + P_LIST_TBEXAM.Value +
P_LIST_TBDIPL.Value + P_LIST_TBASPIR.Value + P_LIST_TBASPIR.Value +
P_LIST_TBMETOD.Value;
pSM := pSUM - P_LIST_TBMETOD.Value;
YaerReport.pSUM.Caption := IntToStr(pSUM);
YaerReport.pALLCLOCK.Caption := IntToStr(pSUM);
YaerReport.pVTCLOCK.Caption := IntToStr(pSM);
YaerReport.PeriodLB.Caption := P_LIST_TBJOB.Value + ' ' +
P_LIST_TBPNAM.Value + ' ИЕРМОД: c ' + DateToStr(BeginDTP.Date) +
' no ' + DateToStr(EndDTP.Date);

YaerReport.preview;
end;

```

Для кнопки «Отчет №3» (компонент SelRepBtn) создайте следующую процедуру:

```
procedure TMainForm.SelRepBtnClick(Sender: TObject);
begin
FEditGroup.Text := ''; FEditDisc.Text := '';
  if P_LIST_TB.RecordCount <> 0 then SelReport.PeriodLB.Caption :=
P_LIST_TBJOB.Value + ' ' + P_LIST_TBPNAME.Value + 'ПЕРИОД: с ' +
DateToStr(BeginDTP.Date) + ' по ' + DateToStr(EndDTP.Date);
  SelFilter;
  MSELECT;
  if LEC > 0 then SelReport.LEC.Caption := IntToStr(LEC) else
  SelReport.LEC.Caption := '';
  if PRACT > 0 then SelReport.PRACT.Caption := IntToStr(PRACT) else
  SelReport.PRACT.Caption := '';
  if LAB > 0 then SelReport.LAB.Caption := IntToStr(LAB) else
  SelReport.LAB.Caption := '';
  If KURS > 0 then SelReport.KURS.Caption := IntToStr(KURS) else
  SelReport.KURS.Caption := '';
  if KONS > 0 then SelReport.KONS.Caption := IntToStr(KONS) else
  SelReport.KONS.Caption := '';
  if PKR > 0 then SelReport.PKR.Caption := IntToStr(PKR) else
  SelReport.PKR.Caption := '';

  if KKR > 0 then SelReport.KKR.Caption := IntToStr(KKR) else
  SelReport.KKR.Caption := '';
  if DIFZ > 0 then SelReport.DIFZ.Caption := IntToStr(DIFZ) else
  SelReport.DIFZ.Caption := '';
  if EXAM > 0 then SelReport.EXAM.Caption := IntToStr(EXAM) else
  SelReport.EXAM.Caption := '';
  if DIPL > 0 then SelReport.DIPL.Caption := IntToStr(DIPL) else
  SelReport.DIPL.Caption := '';
  if ASPIR > 0 then SelReport.ASPIR.Caption := IntToStr(ASPIR) else
  SelReport.ASPIR.Caption := '';
  if METHOD > 0 then SelReport.METHOD.Caption := IntToStr(METHOD) else
  SelReport.METHOD.Caption := '';
  SelReport.ALLCLOCK.Caption := IntToStr(LEC + PRACT + LAB + KURS +
KONS + PKR + KKR + DIFZ + EXAM + DIPL + ASPIR + METHOD);
  SelReport.VTCLOCK.Caption := IntToStr(LEC + PRACT + LAB + KURS +
KONS + PKR + KKR + DIFZ + EXAM + DIPL + ASPIR);
  SelReport.preview;
end;
```

В выше представленной программе имеется вызов личного метода *MSELECT* и вызов формы с отчетом *SelReport . preview*.

Объявите метод *MSELECT* в разделе *Private*

```
private
{ Private declarations }
procedure Link;
procedure RedFilter;
procedure TransInfo;
procedure SelFilter;
procedure MSELECT;
```

Запишите процедуру метода **MSELECT** в секцию личных методов.!!!

```
{Выборка по условиям DATE - DATE}
procedure TMainForm.MSELECT;
var
i:integer;
begin
  LEC := 0; PRACT := 0; LAB := 0; KURS := 0; KONS := 0; PKR := 0; KKR := 0;
  DIFZ := 0; EXAM := 0; DIPL := 0; ASPIR := 0;  METOD := 0;
  M_CDS.Close;
  M_CDS.CreateDataSet;
  M_CDS.Open;
  M_LIST_TB.First;
  for i := 1 to M_LIST_TB.RecordCount do
  begin
    M_CDS.Append;
    M_CSDISCIP.Value := M_LIST_TBDISCIP.Value;
    M_CDSPGROUP.Value := M_LIST_TBPGROUP.Value;
    M_CDSMDAT.Value := M_LIST_TBMDAT.Value;
    M_CDSTEMA.Value := M_LIST_TBTEMA.Value;

    if M_LIST_TBLEC.Value > 0 then M_CDSLEC.Value := M_LIST_TBLEC.AsString;
    if M_LIST_TBPRACT.Value > 0 then M_CDSPRACT.Value := M_LIST_TBPRACT.AsString;
    if M_LIST_TBLAB.Value > 0 then M_CDSLAB.Value := M_LIST_TBLAB.AsString;
    if M_LIST_TBKURS.Value > 0 then M_CDSKURS.Value := M_LIST_TBKURS.AsString;
    if M_LIST_TBKONS.Value > 0 then M_CDSKONS.Value := M_LIST_TBKONS.AsString;
    if M_LIST_TBPKR.Value > 0 then M_CDSPKR.Value := M_LIST_TBPKR.AsString;
    if M_LIST_TBKKR.Value > 0 then M_CDSKKR.Value := M_LIST_TBKKR.AsString;
    if M_LIST_TBDIFZ.Value > 0 then M_CSDIFZ.Value := M_LIST_TBDIFZ.AsString;
    if M_LIST_TBEXAM.Value > 0 then M_CDSEXAM.Value := M_LIST_TBEXAM.AsString;
    if M_LIST_TBDIPL.Value > 0 then M_CSDIPL.Value := M_LIST_TBDIPL.AsString;
    if M_LIST_TBASPIR.Value > 0 then M_CDSASPIR.Value := M_LIST_TBASPIR.AsString;
    if M_LIST_TBMETOD.Value > 0 then M_CDSMETOD.Value := M_LIST_TBMETOD.AsString;
    M_CDS.Post;
    LEC := LEC + M_LIST_TBLEC.Value;
    PRACT := PRACT + M_LIST_TBPRACT.Value;
    LAB := LAB + M_LIST_TBLAB.Value;
    KURS := KURS + M_LIST_TBKURS.Value;
    KONS := KONS + M_LIST_TBKONS.Value;
    PKR := PKR + M_LIST_TBPKR.Value;
    KKR := KKR + M_LIST_TBKKR.Value;
    DIFZ := DIFZ + M_LIST_TBDIFZ.Value;
    EXAM := EXAM + M_LIST_TBEXAM.Value;
    DIPL := DIPL + M_LIST_TBDIPL.Value;
    ASPIR := ASPIR + M_LIST_TBASPIR.Value;
    METOD := METOD + M_LIST_TBMETOD.Value;
    M_LIST_TB.Next;
  end;
  M_LIST_TB.First;
  M_CDS.First;
end;
```

Скомпилируйте СОЗДАННЫЙ (готовый) проект, проверьте его работу.