УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите теоретические сведения к лабораторной работе, выполните практическое задание.

Результаты работы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 06.03.2023.

Требования к отчету:

Отчет предоставляется преподавателю в электронном варианте и должен содержать:

- **название работы, постановку цели, вывод;**
- **ответы на контрольные вопросы, указанные преподавателем.**

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лабораторная работа № 6 (продолжение)

Тема: Разработка пользовательского интерфейса

Цель работы: проектирование пользовательского интерфейса.

При разработке интерфейса приложения следует руководствоваться эвристическими правилами проектирования **Якоба Нильсена (Jacob Nielsen):**

1. Видимость состояния системы (правило обратной связи)

Компьютерная программа всегда должна информировать пользователя о состоянии своей работы с помощью соответствующих средств. Пользователь обязательно должен видеть, к чему привело любое его действие: ввод данных, нажатие кнопки и т. п.

2. Время оповещения

Промежуток времени, в течение которого пользователь получает информацию о реакции на его действие или о событии, должен быть минимальным.

3. Равенство между системой и реальным миром

Компьютерная программа должна использовать понятия, которые уже знакомы пользователю по реальному миру, к которым он привык.

4. Свобода действий пользователя

Пользователь должен иметь контроль над программой и возможность изменить текущее состояние программы. Очень часто пользователь дает различные команды по ошибке и у него должен быть «аварийный выход» из этой ситуации, четко обозначенный в программе. Чаще всего такой «выход» реализуется в виде кнопки **Cancel (Отмена**).

5. Последовательность и стандарты

Принцип последовательности означает использование одних и тех же средств для выражения схожих образов и выполнения действий, имеющих одинаковую природу.

6. Предупреждение ошибок

Применительно к теме проектирования интерфейсов компьютерных программ принцип предупреждения ошибок означает следующее: «Дизайн, который предупреждает возникновение проблем, лучше, чем самое хорошее сообщение об ошибке».

7. Понимание лучше, чем запоминание

При разработке интерфейса рекомендуется делать все объекты, функции, действия видимыми и легко доступными пользователю. В хорошем интерфейсе инструкции по использованию системы всегда видимы или легко доступны для вызова в любое время, когда это требуется. Это может быть реализовано как в виде продуманной организации элементов интерфейса, так и непосредственно в виде подсказок пользователю. Хороший и очень известный пример — анимированная надпись, которая появляется на панели задач Windows при попадании курсора на кнопку **Пуск**: «Начните работу с нажатия этой кнопки».

8. Гибкость и эффективность использования

При проектировании интерфейса требуется, чтобы интерфейс был одинаково удобен и для новичков, и для опытных пользователей. Для решения этой проблемы прибегают к простому приему: функции, которые ускоряют работу, оформлены так, что они не видны начинающим, но легко доступны продвинутым пользователям.

9. Отсутствие малополезных элементов программ

Не нужно загромождать интерфейс программы элементами, которые в данном случае являются неуместными и малополезными. Дело в том, что каждый элемент, будь то кнопка или текстовая подпись, обязательно отвлекает часть внимания пользователя

10. Распознавание и исправление ошибок

Хорошие сообщения об ошибках — это сообщения, которые объясняют, в чем состоит проблема и, самое главное, как ее исправить.

2. Разработка пользовательского интерфейса

2.1. Разработка эффективных форм

Формы – это строительные блоки интерфейса пользователя. Заголовок формы традиционно используется для вывода информации о двух вещах: названии программы и названии документа, с которым в данный момент работает пользователь. Более удобным является порядок расположения информации в заголовке окна, когда первым идет название открытого документа, а после него — название программы.

Для максимального ускорения процесса ввода данных рекомендуется:

1)назначать клавиатурные эквиваленты команд;

2)располагать элементы формы согласовано с задачами пользователя;

3) выполнять добавление и редактирование записей в одной и той же форме.

Еще одна важная часть разработки форм – создание содержательных и эффективных меню.

Не рекомендуется использовать в своей программе нестандартные элементы интерфейса.

Элементы управления на форме приложения располагаются на равном расстоянии между ними.

Tab Order – это порядок, в котором экранный курсор перемещается по элементам управления в форме при нажатии клавиши <Tab> на клавиатуре

компьютера. По окончании проектирования формы рекомендуется проверять TabOrder и, при необходимости, корректировать его.

При разработке приложения рекомендуется применять шрифты такими, какими они определены по умолчанию. Что касается начертания шрифтов, то принято использовать всего два начертания: нормальное и полужирное (последнее – для выделения важной информации, заголовков и т. п.).

Элемент управления **Список (ListBox)** один из самых популярных во всей палитре компонентов для создания интерфейса (рисунок 40). Он позволяет легко просматривать большие объемы информации и осуществлять выделение нужных строк. Однако у него есть недостаток: отсутствие горизонтальной линейки прокрутки. Из-за этого слишком длинные строки обрезаются на границе элемента, что особенно неприятно, если таким образом становится недоступной какая-либо важная информация.

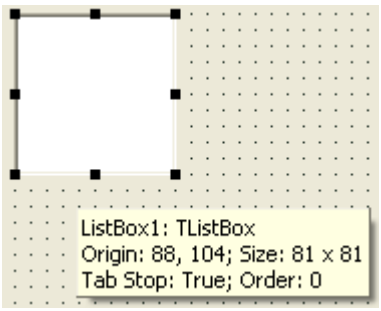
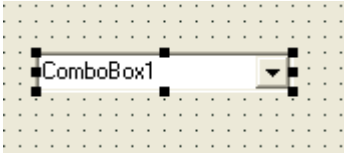


Рисунок 1 - Элемент управления Список

ComboBox практически не отличается от **ListBox**, за исключением дополнительной функции – он позволяет вводить информацию в маленьком поле ввода сверху. Есть несколько типов **ComboBox**, но наиболее популярен спадающий вниз (drop-down combobox), который можно видеть внизу окна диалога выбора файла. Элементы списка задаются в Редакторе строк (рисунок 2).



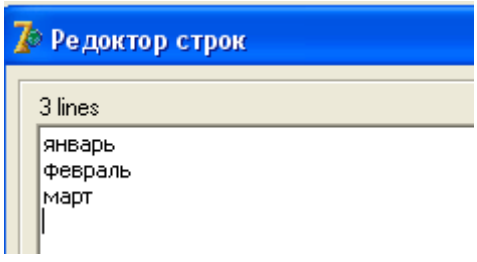


Рисунок 2 - Задание элементов списка ComboBox

Вводимый текст может приравниваться к верхнему регистру, если в свойстве CharCase выбрать ecUpperCase, к нижнему – если выбрать ecLowerCase (рисунок 3).

Рисунок 3 - Варианты ComboBox с различными регистрами

Компонент может работать в трех режимах, определяемых значением свойства Style:

Значение	Механизм работы списка
csDropDown	В присоединенном поле можно указывать значения, отсутствующие в списке. Текст, введенный пользователем, доступен через свойство Текст. Список раскрывающийся
csDropDownList	Допустим только выбор значений, имеющихся в списке. Список раскрывающийся
CsSimple	Отличается от стиля csDropDown только тем, что список не является раскрывающимся

- Максимальное число элементов, одновременно отображаемых в видимой части списка, задается в свойстве DropDownCount. Чтобы раскрыть список из программы, свойству DroppenDown (Раскрыт) нужно присвоить значение True.

- Единственный метод, связанный с выделением данных, это процедура Select All, которая выделяет весь текст, введенный пользователем.

- Основные события для поля со списком:

Название	Условия генерации
OnCange	Пользователь изменил текст в присоединенном поле
OnDropDown	Список раскрывается. Это событие нужно обрабатывать, если содержимое списка может

	меняться во время работы программы

Флажки и **переключатели** используются только для выбора из группы предложенных вариантов. Разница состоит в том, что флажки используются для выбора одновременно нескольких вариантов из группы, а переключатели позволяют сделать выбор в пользу только одного варианта.

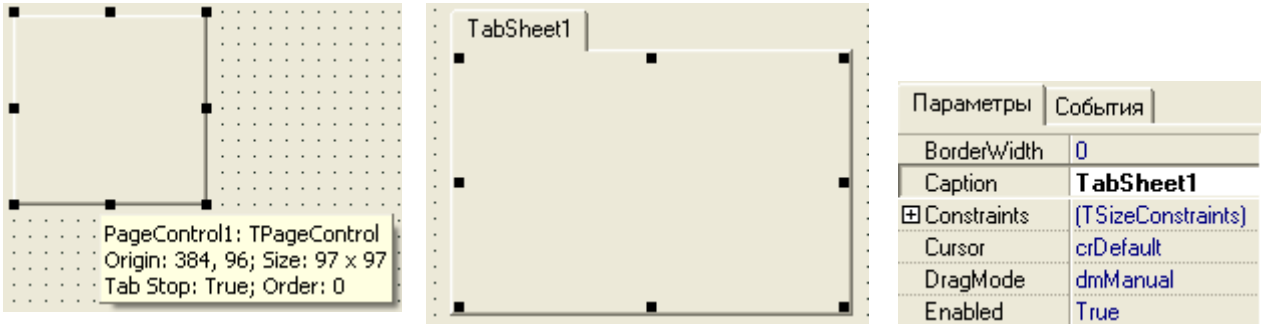


Рисунок 4 - Компонент PageControl

Вкладки широко используются при проектировании интерфейсов современных программ. Этот элемент интерфейса очень эффективен, т.к. позволяет логически группировать большое количество информации, но количество вкладок не должно быть большим. В Delphi для создания вкладок используется компонент PageControl (рисунок 4).

Создание новой страницы **TabSheet** производится нажатием правой кнопки мыши на **PageControl1**, а затем вводом названия каждой закладки в поле *Caption* (рисунок 5).

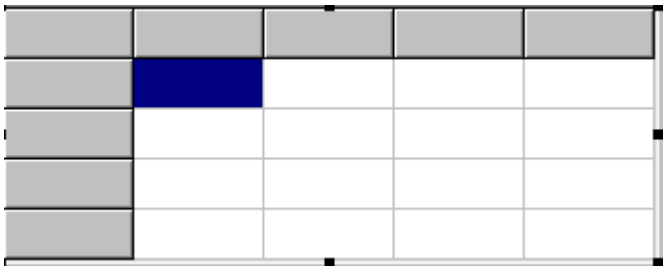


Рисунок 5 - Таблица строк

Компонент **Таблица строк (StringGrid)** служит для представления текстовых данных в виде таблицы. Доступ к каждому элементу таблицы происходит через свойство **Cells**.

Для размещения компонента на форме нужно щелкнуть на пиктограмме  панели **Additional** (Дополнительно) и затем щелчком поместить на форме.

Основное свойство таблицы строк – это двумерный масссив **Cells**, позволяющий обращаться к содержимому ячеек и изменять их содержимое. Первое измерение – номер строки, второе – номер столбца. Нумерация строк и столбцов осуществляется с нуля.

Число строк задается в свойстве **ColCount**, число столбцов – в свойстве **RowCount**. Нумерация строк и столбцов осуществляется с нуля.

Таблица 3 - Свойства, предназначенные для оформления таблицы строк

BorderStyle	Стиль вычерчивания границ ячеек. Значение этого свойства можно комбинировать со значением свойства Ctrl3D для получения ячеек оригинального вида
Ctrl3D	Таблица представляется в трехмерном виде
ColWidths	Массив, хранящий ширину каждого столбца (в пикселах)
DefaultDrawing	Если значение свойства – true, производится автоматическое заполнение содержимого каждой ячейки. В противном случае для таблицы необходимо определить обработчик события OnDrawCell, чтобы запрограммировать процесс отрисовки ячеек
DefaultRowHeight	Начальная высота строки по умолчанию.
FixedColor	Цвет области строк и столбцов, служащих заголовком таблицы
GridHeight	Высота всей таблицы (в пикселах).
GridLineWidth	Ширина (в пикселах) линий, разделяющих ячейки таблицы
GridWidth	Ширина всей таблицы (в пикселах)
Options	Позволяет задавать различные режимы работы таблицы
RowHeights	Массив, хранящий высоту каждой строки (в пикселах)

ScrollBars	Наличие полос прокрутки
VisibleColCount	Число видимых в таблице столбцов (без области заголовка)

2.2. Диалоговые окна и элементы управления

Окно – элемент собственного субэкранного пространства, находящийся в произвольном месте «над» основным экраном. Несколько окон, одновременно располагающихся на экране, могут перекрываться, находясь «выше» или «ниже» относительно друг друга.

Хотя наиболее естественным для оконного интерфейса является графический режим, основные его элементы применимы и в текстовом режиме, где он применяется в равной степени.

При разработке этих элементов интерфейса рекомендуется использовать **золотое сечение**. По этому правилу, стороны диалоговых окон и элементов управления соотносятся как 1,618:1.

В дизайне интерфейса следует группировать элементы в программе (кнопки на панелях инструментов, пункты меню, закладки, опции на этих закладках и т. п.), применяя принцип «кошелька Миллера» (не более семи элементов в группе). Экран программы рекомендуется разбивать на ясно очерченные блоки элементов, может быть, даже с заголовком для каждого блока.

Пример 5. Разработать приложение для анализа статей областного бюджета. Интерфейс приложения разработать в соответствии с рекомендациями пункта 2.

Форма приложения (рисунок 6) содержит компоненты: **Panel**, **PageControl, Chart**, **StringGrid**, **Button**, **ListBox**, **Label**. Первая страница **Статьи бюджета** служит для ввода наименований и размеров статей бюджета в млн руб. Вторая страница **Диаграмма статей бюджета** предназначена для изображения статей бюджета в виде диаграммы (рисунок 7). На третьей странице **Анализ статей бюджета** выводятся результаты анализа статей

бюджета: три наиболее и наименее затратные статьи бюджета (рисунок 8).

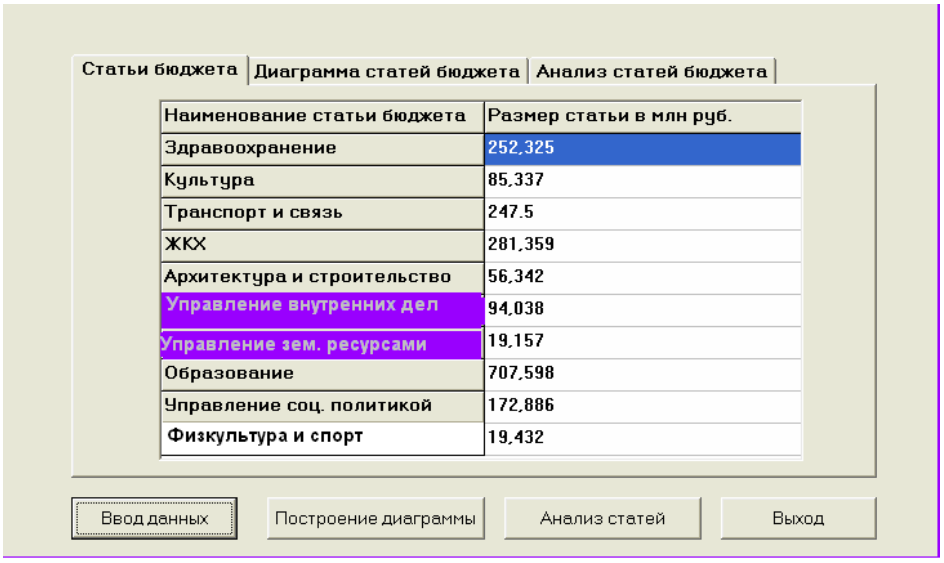


Рисунок 6 - Приложение **Статьи бюджета**

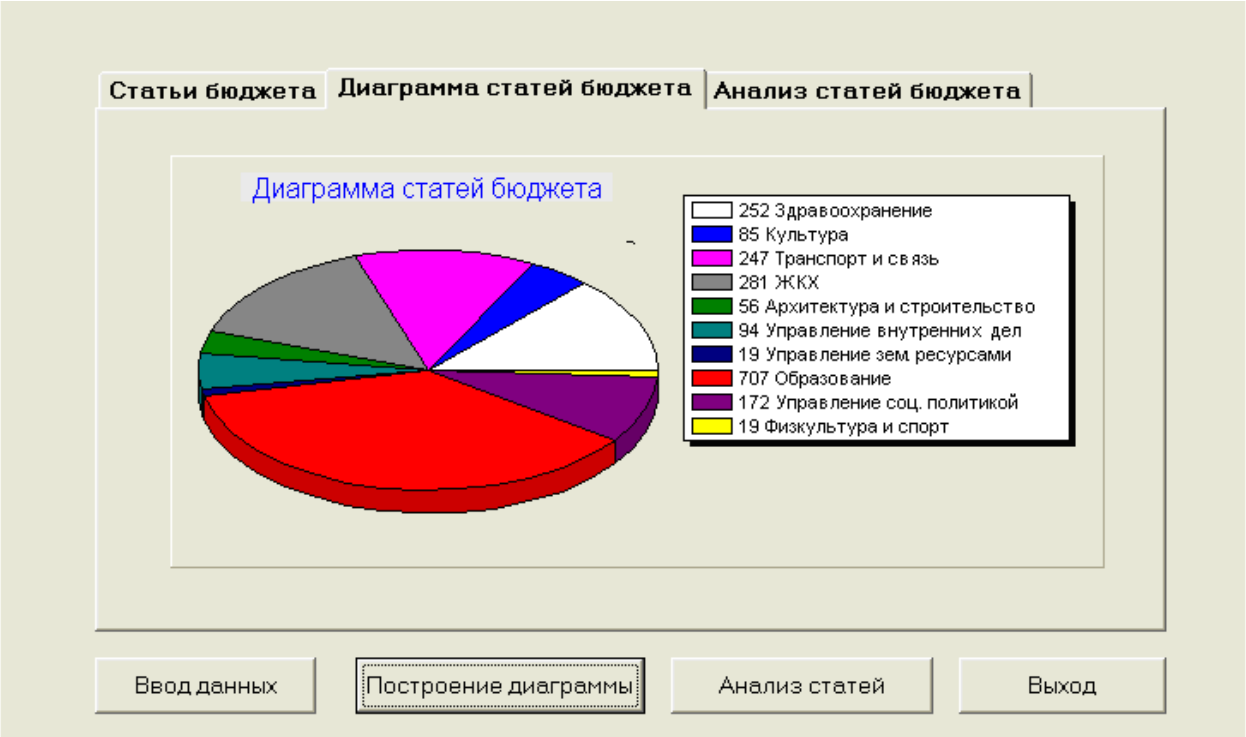


Рисунок 7 - Вкладка **Диаграмма статей бюджета**

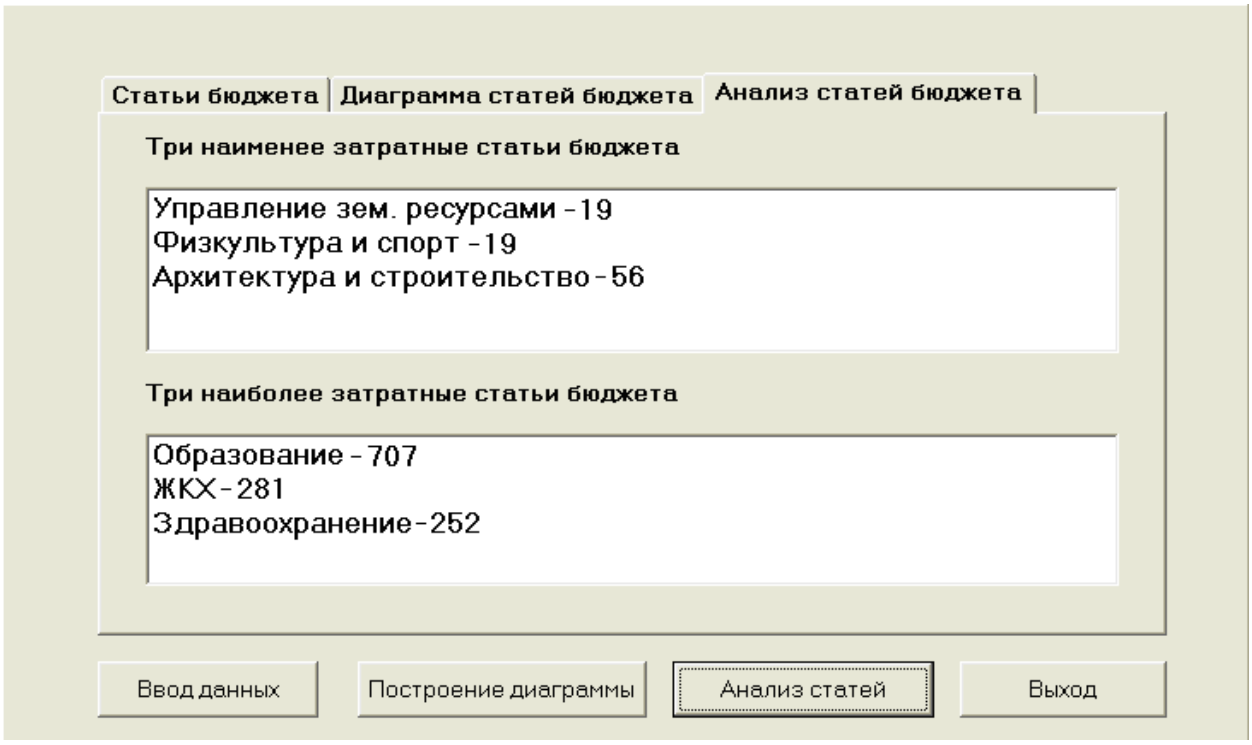


Рисунок 8 - Вкладка **Анализ статей бюджета**

Листинг программы

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, StdCtrls, ComCtrls, ExtCtrls, TeEngine, Series, TeeProcs,
  Chart;
type
  TForm1 = class(TForm)
    Panel1: TPanel;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    StringGrid1: TStringGrid;
    Chart1: TChart;
    Series1: TPieSeries;
    Button4: TButton;
    ListBox1: TListBox;
    ListBox2: TListBox;
    Label1: TLabel;
    Label2: TLabel;
    procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
  procedure Button3Click(Sender: TObject);
private
```

```
{ Private declarations }
 public
  { Public declarations }
 end;
var
 Form1: TForm1;
 type bud=record
 n:string;
 r:integer;
end;
var
   b:array[1..10] of bud;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var i,c:integer;
   s:string;
begin
stringgrid1.defaultcolwidth:=250;//ширина ячейки таблицы
 stringgrid1.defaultrowheight:=24; //высота ячейки
 stringgrid1.colcount:=2;//число столбцов
stringgrid1.Cells[0,0]:='Наименование статьи бюджета';
 stringgrid1.Cells[1,0]:='Размер статьи в млн.руб.';
 for i:=1 to 10 do
  begin
```

```
stringgrid1.RowCount:=i+1;
    b[i].n:=inputbox('Ввод статей бюджета','Введите название статьи
бюджета','');
    stringgrid1.Cells[0,i]:=b[i].n;
    s:=inputbox('Ввод статей бюджета','Введите ее размер','');
    stringgrid1.Cells[1,i]:=s;
    val(s,b[i].r,c);
  end;
end;
procedure TForm1.Button2Click(Sender: TObject);
  var
  i:integer;
  s:Tcolor;
begin
 for i:=1 to 10 do
  begin
   case i of
     1: s:=clwhite;
     2: s:=clblue;
     3: s:=clfuchsia;
     4: s:=clgray;
     5: s:=clgreen;
     6: s:=clteal;
     7: s:=clnavy;
     8: s:=clred;
     9: s:=clpurple;
     10: s:=clyellow;
   end;
form1.series1.AddPie(b[i].r,b[i].n,s);
 end;
```

```
end;
procedure TForm1.Button3Click(Sender: TObject);
var k,i,j,buf:integer;
   mm:string;
begin
 for i:=10 downto 2 do
   begin
    for j:=2 to i do
      if b[j-1].r>b[j].r then
        begin
          buf:=b[j-1].r;
          b[j-1].r:=b[j].r;
          b[j].r:=buf;
          mm:=b[j-1].n;
          b[j-1].n:=b[j].n;
          b[j].n:=mm;
        end;
   end;
        for i:=1 to 3 do listbox1.items.Add(b[i].n+'-'+inttostr(b[i].r));
         for i:=10 downto 8 do listbox2.items.Add(b[i].n+'-'+inttostr(b[i].r));
end;
end.
```

Варианты заданий

1. Разработать приложение для анализа успеваемости студенческой группы. На первой странице организовать ввод фамилий студентов и экзаменационных оценок, расчет среднего балла сдачи каждого экзамена группой. На второй странице в виде гистограммы вывести средние баллы по каждому экзамену. На третьей странице вывести наименьший и наибольший средние баллы.

2. Разработать приложение для анализа продаж товаров в фирме за квартал. На первой странице организовать ввод наименований товаров и сумм продаж. На второй странице в виде гистограммы показать суммы продаж товаров. На третьей странице вывести по две наименьшие и наибольшие суммы продаж.

3. Разработать приложение для анализа продаж товаров в фирме. На первой странице организовать ввод наименований товаров и сумм продаж. На второй странице в виде гистограммы показать суммы продаж товаров. На третьей странице вывести наименьшую и наибольшую суммы продаж.

4. Разработать приложение для определения студентов, владеющих языками программирования. На первой странице организовать ввод фамилий студентов и владение языком программирования C или Pascal. На второй странице в виде гистограммы вывести числа студентов, владеющих отдельно C и Pascal. На третьей странице вывести фамилии студентов, владеющих двумя языками программирования.

5. Разработать приложение для анализа статей бюджета фирмы. На первой странице организовать ввод значений статей бюджета. На второй странице в виде гистограммы вывести значения статей бюджета. На третьей странице вывести пять наиболее затратных статей бюджета.

6. Разработать приложение для анализа статей бюджета города. На первой странице организовать ввод значений статей бюджета. На второй странице в виде гистограммы вывести значения статей бюджета. На третьей странице вывести четыре наименее затратных статей бюджета.

7. Разработать приложение для определения студентов, умеющих работать с графическими пакетами. На первой странице организовать ввод фамилий студентов и владение AutoCAD (А) или КОМПАС-ГРАФИК (К). На второй странице в виде гистограммы вывести числа студентов, владеющих отдельно А и К. На третьей странице вывести фамилии студентов, умеющих работать с А и К.

8. Разработать приложение для определения студентов, умеющих работать с графическими пакетами. На первой странице организовать ввод фамилий студентов и владение CorelDraw (С) или Photoshop (Р). На второй странице в виде гистограммы вывести числа студентов, владеющих отдельно С и Р. На третьей странице вывести фамилии студентов, умеющих работать с Photoshop.

9. Разработать приложение для анализа успеваемости студенческой группы. На первой странице организовать ввод фамилий студентов и экзаменационных оценок, расчет среднего балла сдачи каждого экзамена группой. На второй странице в виде гистограммы вывести средние баллы по каждому экзамену. На третьей странице вывести название дисциплины с наибольшим средним баллом.

10. Разработать приложение для анализа успеваемости студенческой группы. На первой странице организовать ввод фамилий студентов и экзаменационных оценок, расчет среднего балла сдачи каждого экзамена группой. На второй странице в виде гистограммы вывести средние баллы по каждому экзамену. На третьей странице вывести названия двух дисциплин с наибольшими средними баллами.

11. Разработать приложение для анализа успеваемости студенческой группы. На первой странице организовать ввод фамилий студентов и экзаменационных оценок, расчет среднего балла сдачи каждого экзамена группой. На второй странице в виде гистограммы вывести средние баллы по каждому экзамену. На третьей странице вывести названия дисциплин с наибольшим и наименьшим средними баллами.

12. Разработать приложение для анализа продаж товаров в фирме. На первой странице организовать ввод наименований товаров и сумм продаж. На второй странице в виде гистограммы показать суммы продаж товаров. На третьей странице вывести наименования трех товаров, сумма продаж которых наибольшая.

13. Разработать приложение для анализа продаж товаров в фирме. На первой странице организовать ввод наименований товаров и сумм продаж. На второй странице в виде гистограммы показать суммы продаж товаров. На третьей странице вывести наименования пяти товаров, сумма продаж которых наименьшая.

14. Разработать приложение. На первой странице организовать ввод десяти крупнейших рек мира с указанием их длины в км. На второй странице изобразить данные в виде гистограммы. На третьей странице вывести названия рек, длины которых более трех тысяч км.

15. Разработать приложение. На первой странице организовать ввод десяти столиц государств мира с указанием числа жителей в млн. На второй странице изобразить данные в виде гистограммы. На третьей странице вывести название столиц, имеющих население более 8 млн. жителей.