Содержание

Задание на разработку программного комплекса

[Функциональная структура приложения](#_Toc295286327)

Спецификация программного комплекса

[Структурная диаграмма программного комплекса](#_Toc295286329)

Данные для тестирования

[Инструкция пользователя](#_Toc295286331)

Приложение

# Задание на разработку программного комплекса

Разработать информационную подсистему отдела кадров.

Информация, обрабатываемая в подсистеме, должна храниться в текстовом или типизированном файлах.

Данные, которые должны быть отражены в подсистеме: фамилия, месяц и год приема на работу, образование, специальность, пол, отношение к воинской службе, год рождения.

Сведения о сотрудниках пополняются по мере приема на работу.

В системе должны решаться следующие задачи:

1. Создание файла.
2. Дополнение файла.
3. Корректировка данных в файле.
4. Формирование ответов на запросы пользователя:
   1. Вывести в виде таблицы фамилии, образование, отношение к воинской службе сотрудников младше 30 лет;
   2. Определить средний возраст работающих мужчин и женщин;
   3. Вывести на экран фамилии сотрудников, принятых в текущем году, в порядке убывания их возраста;
   4. Построить графики изменения уровня образования от возраста;
   5. Построить круговую диаграмму, отражающую возрастное соотношение работающих: выделить группы до 30 лет, от 31 до 50 лет, старше 51 года;
   6. Построить столбиковую диаграмму, отражающую долю сотрудников с высшим, средним специальным, средним, неполным средним, начальным образованием. Столбики диаграммы расположить в порядке возрастания.

**Функциональная структура приложения**

Программный комплекс предназначен для работы с базой данных в виде типизированного файла с расширением txt. Комплекс осуществляет чтение, запись и создание базы.

Помимо этого программный комплекс предоставляет возможность создавать, удалять и редактировать записи в БД. А также осуществляет выборку данных на основе фиксированных правил отбора и построение графиков и диаграмм.

# Спецификация программного комплекса

Основной задачей является разработка информационной подсистемы отдела кадров. Поскольку для реализации требуется дружественный интерфейс то было принято решение вести разработку в среде программирования Delphi 7. Это улучшает потребительские качества программного продукта, а также позволит создать графический интерфейс для операционной системы Windows.

Единственное требование для данного программного пакета это наличие ОС семейства Windows.

Интерфейс программы состоит из основной формы и вспомогательных форм, таких как форма редактирования записи, форма отображения результатов запроса, форма построения графиков, форма выбора базы данных. Попасть на эти формы пользователь может при помощи использования кнопок на основной форме.

В качестве входных данных программа использует типизированный файл базы данных, а также сведения о работнике такие как:

* Фамилия – текстовое поле с максимальной длинной 100 символов;
* Дата приема на работу – поле типа TDate;
* Образование – текстовое поле длинной 100 символов;
* Специальность - текстовое поле длинной 100 символов;
* Пол – поле типа byte;
* Отношение к воинской службе - текстовое поле длинной 100 символов;
* Год рождения – поле типа TDate.

В качестве выходных данных программы выступает файл базы данных. А также графическое отображение результатов запросов пользователя. Из графической информации пользователь может получить следующие типы объектов:

* Таблица с данными;
* График;
* Текстовая информация.

Поскольку вся информация проверяется при вводе и пользователь выбирает только то, что может использоваться в данном поле, то ловится всего одна ошибка связанная с попыткой редактирования пустой строки таблицы данных.

# Структурная диаграмма программного комплекса

Схема построения программного средства и основных классов:

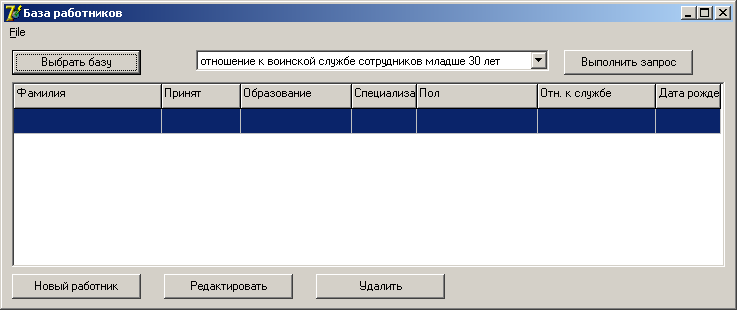


# Данные для тестирования

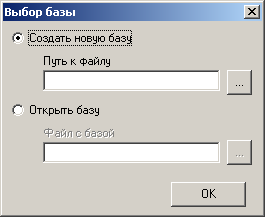
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Фамилия | Дата рождения | Принят на работу | Пол | Образование | Специальность | Отношение к воинской службе |
| Петров | 03.12.1956 | 12.12.2001 | муж | Высшее | Инженер | В запасе |
| Иванов | 15.01.1990 | 11.12.2009 | муж | Среднее | Техник | В запасе |
| Федорова | 30.04.1964 | 04.11.2004 | жен | Высшее | Бухгалтер |  |
| Сидоров | 24.07.1992 | 0707.2010 | муж | Начальное | Курьер | Призывник |

# Инструкция пользователя

При запуске программы пользователь попадает но основное окно программы:



Если необходимо использовать уже созданную базу то необходимо использовать кнопку «Выбрать базу». После этого пользователь попадет на окно выбора создания или открытия существующей базы.



Для того что бы выбрать базу необходимо нажать на кнопку «…» и в открывшемся окне задать имя или выбрать уже существующий файл базы.

**Внимание:** Если выбран режим создания БД и указывается существующий файл, то он будет перезаписан.

При этом даже если не загружать базу есть возможность создавать новые записи. По завершению работы и закрытию программы пользователю будет предложено сохранить внесенные изменения.

Управлять записями можно используя три кнопки внизу экрана. При помощи этих кнопок можно создавать. Удалять или редактировать записи.

Существует возможность выполнения определенных запросов. Для выбора нужного запроса необходимо воспользоваться выпадающим списком в верхней части экрана, после чего нажать на кнопку «Выполнить запрос». Как только программа осуществит выборку на экран будет выведена форма с результатами запроса.

график диаграмма типизированный файл

# Приложение. Исходные тексты программных модулей

unit Unit1;

interface

uses

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

Dialogs, StdCtrls, Menus, Grids, DBGrids, DB, DBTables;

type

TWorker=record

FIO: string[100];

StartWork : TDate;

Edication : String[100];

Spec : String[100];

sex: byte;

Armi: String[100];

BirthDay: TDate;

end;

type

TForm1 = class(TForm)

MainMenu1: TMainMenu;

File1: TMenuItem;

Exit1: TMenuItem;

Button1: TButton;

Button2: TButton;

Button3: TButton;

StringGrid1: TStringGrid;

Button4: TButton;

Button5: TButton;

ComboBox1: TComboBox;

procedure Exit1Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure ReadData(fileName:string; new:bool);

procedure Button2Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure TableRowToWorker(index:integer; table:TStringGrid);

procedure AddRowToTable(row:TWorker; table:TStringGrid);

procedure EditTableRow(row:TWorker; table:TStringGrid);

procedure Button5Click(Sender: TObject);

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

procedure SetBase();

procedure WriteData(fileName:string; table:TStringGrid);

procedure Button3Click(Sender: TObject);

procedure FormShow(Sender: TObject);

private

{ Private declarations }

Worker:TWorker;

Base:String;

tableDS:TDataSet;

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

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

{$R \*.dfm}

procedure TForm1.Exit1Click(Sender: TObject);

begin

Close;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

SetBase();

end;

//выводит форму выбора файла с базой данных

procedure TForm1.SetBase();

var

i:integer;

begin

if Form2.ShowModal = mrOk then

begin

if Form2.RadioButton1.Checked then

begin

Base := Form2.Edit1.Text;

ReadData(Base, true);

end

else

begin

Base := Form2.Edit3.Text;

ReadData(Base, false);

end;

end;

end;

//переводит объект типа TWorker в строку таблицы

//row - новая строка типа TWorker

//table - таблица, в которую добавляется строка

procedure TForm1.AddRowToTable(row:TWorker; table:TStringGrid);

begin

table.Cells[0, table.RowCount - 1] := row.FIO;

table.Cells[1, table.RowCount - 1] := DateToStr(row.StartWork);

table.Cells[2, table.RowCount - 1] := row.Edication;

table.Cells[3, table.RowCount - 1] := row.Spec;

if row.sex = 0 then

table.Cells[4, table.RowCount - 1] := 'муж'

else

table.Cells[4, table.RowCount - 1] := 'жен';

table.Cells[5, table.RowCount - 1] := row.Armi;

table.Cells[6, table.RowCount - 1] := DateToStr(row.BirthDay);

table.RowCount := table.RowCount + 1;

end;

//Вносит изменения в отредактированную строку

//row - отредактированные данные типа TWorker

//table - таблица, в которую вносятся изменения

procedure TForm1.EditTableRow(row:TWorker; table:TStringGrid);

var

i:integer;

begin

i := table.Selection.Top;

table.Cells[0, i] := row.FIO;

table.Cells[1, i] := DateToStr(row.StartWork);

table.Cells[2, i] := row.Edication;

table.Cells[3, i] := row.Spec;

if row.sex = 0 then

table.Cells[4, i] := 'муж'

else

table.Cells[4, i] := 'жен';

table.Cells[5, i] := row.Armi;

table.Cells[6, i] := DateToStr(row.BirthDay);

end;

//Преобразует строку таблицы в TWorker

//index - номер строки

//table - таблица, в которой находятся данные

procedure TForm1.TableRowToWorker(index:integer; table:TStringGrid );

var

i:integer;

begin

i := index;

Worker.FIO := table.Cells[0, i];

Worker.StartWork := StrToDate(table.Cells[1, i]);

Worker.Edication := table.Cells[2, i];

Worker.Spec := table.Cells[3, i];

if table.Cells[4, i] = 'муж' then

Worker.sex := 0

else

Worker.sex := 1;

Worker.Armi := table.Cells[5, i];

Worker.BirthDay := StrToDate(table.Cells[6, i]);

end;

//читает данные из файла в таблицу

// fileName - имя файла и путь к нему

//new - показывает отрывается существующая база или создается новая

procedure TForm1.ReadData(fileName:string; new:bool);

var

F:File of TWorker;

size:integer;

begin

AssignFile(F,fileName);

if new then

begin

Rewrite(F);

end

else

begin

Reset(F);

Seek(F,0);

while (not EOF(F)) do

begin

Read(F,Worker);

AddRowToTable(Worker,StringGrid1);

end;

end;

CloseFile(F);

end;

//событие при нажатии кнопки создания новой записи

procedure TForm1.Button2Click(Sender: TObject);

begin

Form3.editRecord := false;

if Form3.ShowModal = mrOk then

begin

Worker := Form3.Worker;

AddRowToTable(Worker,StringGrid1);

//Worker.StartWork := Form3.Edit1;

end;

end;

//событие при нажатии редактирования

procedure TForm1.Button4Click(Sender: TObject);

begin

try

Form3.editRecord := true;

TableRowToWorker(StringGrid1.Selection.Top,StringGrid1);

Form3.Worker := Worker;

if Form3.ShowModal = mrOk then

begin

Worker := Form3.Worker;

EditTableRow(Worker, StringGrid1);

end;

except

MessageDlg('При попытки редактирования произошла ошибка',mtError,[mbOK],0);

end;

end;

//Функция удаления строки из TStringGrid

//RowNumber - номер строки

//Grid - таблица из которой происходит удаление

procedure GridDeleteRow(RowNumber: Integer; Grid: TstringGrid);

var

i: Integer;

begin

Grid.Row := RowNumber;

if (Grid.Row = Grid.RowCount - 1) then

{ On the last row}

Grid.RowCount := Grid.RowCount - 1

else

begin

{ Not the last row}

for i := RowNumber to Grid.RowCount - 1 do

Grid.Rows[i] := Grid.Rows[i + 1];

Grid.RowCount := Grid.RowCount - 1;

end;

end;

//Удаление строк

procedure TForm1.Button5Click(Sender: TObject);

var

i,n,start:integer;

begin

n := StringGrid1.Selection.Bottom - StringGrid1.Selection.Top + 1;

start:= StringGrid1.Selection.Top;

if StringGrid1.Selection.Bottom < StringGrid1.Selection.Top then

start:= StringGrid1.Selection.Bottom;

for i := 1 to n do

begin

GridDeleteRow(start + i - 1, StringGrid1);

end;

end;

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

var

res:integer;

begin

if Base = '' then

begin

res := MessageDlg('Сохранить записи?',mtInformation,[mbYes, mbNo],0);

if res = mrYes then

SetBase()

else

exit;

end;

WriteData(Base, StringGrid1);

end;

//запись данных в файл базы

//fileName - имя и путь к файлу базы

//table - Таблица с данными

procedure TForm1.WriteData(fileName:string; table:TStringGrid);

var

F:File of TWorker;

i:integer;

begin

AssignFile(F,fileName);

Reset(F);

Seek(F,0);

for i := 1 to table.RowCount-2 do

begin

TableRowToWorker(i,StringGrid1);

Write(F,Worker);

end;

CloseFile(F);

end;

//Функция быстрой сортировки

//А - массив для сортировки

//iLo - начало массива

//iHi - конец массива

procedure QuickSort(var A: array of TWorker; iLo, iHi: Integer);

var

Lo, Hi: Integer;

Pivot :TDate;

T: TWorker;

begin

Lo := iLo;

Hi := iHi;

Pivot := A[(Lo + Hi) div 2].BirthDay;

repeat

while A[Lo].BirthDay < Pivot do Inc(Lo) ;

while A[Hi].BirthDay > Pivot do Dec(Hi) ;

if Lo <= Hi then

begin

T := A[Lo];

A[Lo] := A[Hi];

A[Hi] := T;

Inc(Lo) ;

Dec(Hi) ;

end;

until Lo > Hi;

if Hi > iLo then QuickSort(A, iLo, Hi) ;

if Lo < iHi then QuickSort(A, Lo, iHi) ;

end;

//Выполняет запросы

procedure TForm1.Button3Click(Sender: TObject);

var

i:integer;

Workers:array of TWorker;

y1,m1,d1,y2,m2,d2:Word;

MAge, WAge : real;

MStch, WStch : integer;

begin

if ComboBox1.ItemIndex = 0 then

begin

SetLength(Workers, StringGrid1.RowCount - 2);

DecodeDate(Now,y2,m2,d2);

for i := 1 to StringGrid1.RowCount - 2 do

begin

TableRowToWorker(i,StringGrid1);

DecodeDate(Worker.BirthDay,y1,m1,d1);

if (abs(y2 - y1) <= 30) and (Worker.sex = 0) then

Workers[i-1] := Worker;

end;

Form4.FillTable(Workers);

Form4.ShowModal;

end

else

if ComboBox1.ItemIndex = 1 then

begin

DecodeDate(Now,y2,m2,d2);

MStch := 0;

WStch := 0;

for i := 1 to StringGrid1.RowCount - 2 do

begin

TableRowToWorker(i,StringGrid1);

DecodeDate(Worker.BirthDay,y1,m1,d1);

if (Worker.sex = 0) then

begin

MAge := MAge + y2 - y1;

MStch := MStch +1;

end

else

begin

WAge := WAge + y2 - y1;

WStch := WStch +1;

end;

end;

MAge := MAge / MStch;

WAge := WAge / WStch;

Form5.Label3.Caption := FloatToStrF(MAge,ffGeneral, 8, 2);

Form5.Label4.Caption := FloatToStrF(WAge,ffGeneral, 8, 2);

Form5.ShowModal;

end

else

if ComboBox1.ItemIndex = 2 then

begin

SetLength(Workers, StringGrid1.RowCount - 1);

DecodeDate(Now,y2,m2,d2);

for i := 1 to StringGrid1.RowCount - 2 do

begin

TableRowToWorker(i,StringGrid1);

DecodeDate(Worker.StartWork,y1,m1,d1);

if y2 = y1 then

Workers[i] := Worker;

end;

if (Length(Workers) > 3) then

QuickSort(Workers,1, Length(Workers))

else

if Workers[1].BirthDay < Workers[1].BirthDay then

begin

worker := Workers[1];

Workers[1] := Workers[2];

Workers[2] := worker;

end;

Form6.FillTable(Workers);

Form6.ShowModal;

end

else

if ComboBox1.ItemIndex = 3 then

begin

SetLength(Workers, StringGrid1.RowCount - 2);

for i := 1 to StringGrid1.RowCount - 2 do

begin

TableRowToWorker(i,StringGrid1);

Workers[i-1] := Worker;

end;

Form7.FillTable(Workers);

Form7.ShowModal;

end

else

if ComboBox1.ItemIndex = 4 then

begin

SetLength(Workers, StringGrid1.RowCount - 2);

for i := 1 to StringGrid1.RowCount - 2 do

begin

TableRowToWorker(i,StringGrid1);

Workers[i-1] := Worker;

end;

Form7.FillKrug(Workers);

Form7.ShowModal;

end

else

if ComboBox1.ItemIndex = 5 then

begin

SetLength(Workers, StringGrid1.RowCount - 2);

for i := 1 to StringGrid1.RowCount - 2 do

begin

TableRowToWorker(i,StringGrid1);

Workers[i-1] := Worker;

end;

Form7.FillStolb(Workers);

Form7.ShowModal;

end;

end;

procedure TForm1.FormShow(Sender: TObject);

begin

StringGrid1.Cells[0,0] := 'Фамилия';

StringGrid1.Cells[1,0] := 'Принят';

StringGrid1.Cells[2,0] := 'Образование';

StringGrid1.Cells[3,0] := 'Специализация';

StringGrid1.Cells[4,0] := 'Пол';

StringGrid1.Cells[5,0] := 'Отн. к службе';

StringGrid1.Cells[6,0] := 'Дата рождения';

end;

end.

unit Unit2;

interface

uses

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

Dialogs, StdCtrls;

type

TForm2 = class(TForm)

RadioButton1: TRadioButton;

RadioButton2: TRadioButton;

Edit1: TEdit;

Label1: TLabel;

Label3: TLabel;

Edit3: TEdit;

Button1: TButton;

Button2: TButton;

Button3: TButton;

OpenDialog1: TOpenDialog;

SaveDialog1: TSaveDialog;

procedure RadioButton1Click(Sender: TObject);

procedure RadioButton2Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

{$R \*.dfm}

procedure TForm2.RadioButton1Click(Sender: TObject);

begin

Label1.Enabled := true;

Edit1.Enabled := true;

Edit3.Enabled := false;

Label3.Enabled := false;

Button2.Enabled := true;

Button3.Enabled := false;

end;

procedure TForm2.RadioButton2Click(Sender: TObject);

begin

Label1.Enabled := false;

Edit1.Enabled := false;

Button2.Enabled := false;

Edit3.Enabled := true;

Label3.Enabled := true;

Button3.Enabled := true;

end;

procedure TForm2.Button2Click(Sender: TObject);

begin

if SaveDialog1.Execute then

begin

Edit1.Text := SaveDialog1.FileName+'.txt';

end

end;

procedure TForm2.Button3Click(Sender: TObject);

begin

if OpenDialog1.Execute then

begin

Edit3.Text := OpenDialog1.FileName;

end;

end;

procedure TForm2.Button1Click(Sender: TObject);

begin

ModalResult := mrOk;

//Close;

end;

end.

unit Unit3;

interface

uses

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

Dialogs, StdCtrls, ComCtrls, Unit1;

type

TForm3 = class(TForm)

Label1: TLabel;

Edit1: TEdit;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

ComboBox1: TComboBox;

Edit2: TEdit;

ComboBox2: TComboBox;

Edit3: TEdit;

DateTimePicker1: TDateTimePicker;

DateTimePicker2: TDateTimePicker;

Button1: TButton;

Button2: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure ClearFields();

procedure FormShow(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

Worker:TWorker;

editRecord:bool;

end;

var

Form3: TForm3;

implementation

{$R \*.dfm}

procedure TForm3.ClearFields();

begin

Edit1.Text := '';

DateTimePicker2.Date := Now;

ComboBox1.ItemIndex := 0;

Edit2.Text := '';

ComboBox2.ItemIndex := 0;

Edit3.Text := '';

DateTimePicker1.Date := Now;

end;

procedure TForm3.Button1Click(Sender: TObject);

begin

Worker.FIO := Edit1.Text;

Worker.StartWork := DateTimePicker2.Date;

Worker.Edication := ComboBox1.Text;

Worker.Spec := Edit2.Text;

Worker.sex := ComboBox2.ItemIndex;

Worker.Armi := Edit3.Text;

Worker.BirthDay := DateTimePicker1.Date;

ModalResult := mrOK;

end;

procedure TForm3.Button2Click(Sender: TObject);

begin

ModalResult := mrCancel;

end;

procedure TForm3.FormShow(Sender: TObject);

begin

if not editRecord then

ClearFields

else

begin

Edit1.Text := Worker.FIO;

DateTimePicker2.Date := Worker.StartWork;

ComboBox1.Text := Worker.Edication;

Edit2.Text := Worker.Spec;

ComboBox2.ItemIndex := Worker.sex;

Edit3.Text := Worker.Armi;

DateTimePicker1.Date := Worker.BirthDay;

end;

end;

end.

unit Unit4;

interface

uses

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

Dialogs, Grids, Unit1;

type

TForm4 = class(TForm)

StringGrid1: TStringGrid;

private

{ Private declarations }

public

{ Public declarations }

procedure FillTable(Workers:array of TWorker);

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

procedure TForm4.FillTable(Workers:array of TWorker);

var

i,n:integer;

begin

n:= Length(Workers);

StringGrid1.Cells[0,0] := 'Фамилия';

StringGrid1.Cells[1,0] := 'Образование';

StringGrid1.Cells[2,0] := 'Отношение к службе';

StringGrid1.RowCount := n+1;

for i:=1 to n do

begin

StringGrid1.Cells[0,i] := Workers[i-1].FIO;

StringGrid1.Cells[1,i] := Workers[i-1].Edication;

StringGrid1.Cells[2,i] := Workers[i-1].Armi;

end;

end;

end.

unit Unit6;

interface

uses

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

Dialogs, Grids, Unit1;

type

TForm6 = class(TForm)

StringGrid1: TStringGrid;

private

{ Private declarations }

public

procedure FillTable(Workers:array of TWorker);

{ Public declarations }

end;

var

Form6: TForm6;

implementation

{$R \*.dfm}

procedure TForm6.FillTable(Workers:array of TWorker);

var

i,n:integer;

begin

n:= Length(Workers);

StringGrid1.Cells[0,0] := 'Фамилия';

StringGrid1.Cells[1,0] := 'Дата рожд.';

StringGrid1.RowCount := n;

for i:=1 to n-1 do

begin

StringGrid1.Cells[0,i] := Workers[i].FIO;

StringGrid1.Cells[1,i] := DateToStr(Workers[i].BirthDay);

end;

end;

end.

unit Unit7;

interface

uses

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

Dialogs, TeEngine, TeeFunci, Series, ExtCtrls, TeeProcs, Chart, Unit1;

type

TForm7 = class(TForm)

Chart1: TChart;

Series1: TLineSeries;

TeeFunction1: TMultiplyTeeFunction;

Chart2: TChart;

Series3: TPieSeries;

Chart3: TChart;

Series2: TBarSeries;

private

{ Private declarations }

public

{ Public declarations }

procedure FillTable(Workers:array of TWorker);

procedure FillKrug(Workers:array of TWorker);

procedure FillStolb(Workers:array of TWorker);

end;

var

Form7: TForm7;

implementation

{$R \*.dfm}

procedure TForm7.FillTable(Workers:array of TWorker);

var

i,n:integer;

y1,m1,d1,y2,m2,d2:Word;

yars:array[0..60] of integer;

begin

//Chart1.SeriesList.Clear;

//Chart1.SeriesList.Add(Series1);

Chart1.Visible := true;

Chart2.Visible := false;

Chart3.Visible := false;

Caption := 'Образование';

Series1.Clear;

n:= Length(Workers);

DecodeDate(Now,y2,m2,d2);

for i:=1 to 60 do

yars[i-1] := 0;

for i:=1 to n do

begin

DecodeDate(Workers[i-1].BirthDay,y1,m1,d1);

if (Workers[i-1].Edication = 'Высшее') then

yars[y2-y1] := yars[y2-y1] + 5

else

if (Workers[i-1].Edication = 'Cредне специальное') then

yars[y2-y1] := yars[y2-y1] + 4

else

if (Workers[i-1].Edication = 'Cреднее') then

yars[y2-y1] := yars[y2-y1] + 3

else

if (Workers[i-1].Edication = 'Неполное среднее') then

yars[y2-y1] := yars[y2-y1] + 2

else

yars[y2-y1] := yars[y2-y1] + 1;

end;

for i:=1 to 60 do

Series1.AddXY(i,yars[i-1]);

end;

procedure TForm7.FillKrug(Workers:array of TWorker);

var

i,n : integer;

y1,m1,d1,y2,m2,d2 : Word;

stch1,stch2,stch3 : integer;

begin

//Chart1.SeriesList.Clear;

//Chart1.SeriesList.Clear;

//Chart1.SeriesList.Add(Series2);

Chart1.Visible := false;

Chart2.Visible := true;

Chart3.Visible := false;

Caption := 'Возрастное соотношение';

Series3.Clear;

stch1 := 0;

stch2 := 0;

stch3 := 0;

n:= Length(Workers);

DecodeDate(Now,y2,m2,d2);

for i:=1 to n do

begin

DecodeDate(Workers[i-1].BirthDay,y1,m1,d1);

if (y2-y1 <= 30) then

Inc(stch1)

else

if (y2-y1 <= 50) then

Inc(stch2)

else

Inc(stch3);

end;

//Series2.Add(50);

Series3.Add(stch1);

Series3.Add(stch2);

Series3.Add(stch3);

end;

procedure TForm7.FillStolb(Workers:array of TWorker);

var

i,n : integer;

stch1,stch2,stch3,stch4,stch5 : integer;

begin

Chart1.Visible := false;

Chart2.Visible := false;

Chart3.Visible := true;

Caption := 'Образование сотрудников';

Series2.Clear;

stch1 := 0;

stch2 := 0;

stch3 := 0;

stch4 := 0;

stch5 := 0;

n:= Length(Workers);

for i:=1 to n do

begin

if (Workers[i-1].Edication = 'Высшее') then

Inc(stch1)

else

if (Workers[i-1].Edication = 'Cредне специальное') then

Inc(stch2)

else

if (Workers[i-1].Edication = 'Cреднее') then

Inc(stch3)

else

if (Workers[i-1].Edication = 'Неполное среднее') then

Inc(stch4)

else

Inc(stch5);

end;

Series2.Add(stch1);

Series2.Add(stch2);

Series2.Add(stch3);

Series2.Add(stch4);

Series2.Add(stch5);

end;

end.