**Задание:**

**Записи**

Сведения о товаре в магазине:

-наименование;

-сорт;

-цена за ед.;

-дата получения;

Вывести сведения о товаре в магазине, полученного в прошлом месяце первого сорта в алфавитном порядке наименований.

**Код Delphi 10:**

**Main.pas**

**unit** Main;

**interface**

**uses**

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.Grids, Vcl.StdCtrls;

**type**

TProduct = **Record**

Title: String[15];

Sort: String[1];

Price: String[5];

Date: String[10];

**end**;

TBaseProd = **array of** TProduct;

TMainForm = **class**(TForm)

SG: TStringGrid;

MainMenu: TMainMenu;

PopupMenu: TPopupMenu;

FileM: TMenuItem;

About: TMenuItem;

Openfile: TMenuItem;

Savetofile: TMenuItem;

N1: TMenuItem;

**Exit**: TMenuItem;

Developer: TMenuItem;

Info: TMenuItem;

Open: TOpenDialog;

Save: TSaveDialog;

Add: TButton;

Edit: TButton;

Delete: TButton;

Show: TButton;

Label1: TLabel;

**procedure** InfoClick(Sender: TObject);

**procedure** DeveloperClick(Sender: TObject);

**procedure** FormCloseQuery(Sender: TObject; **var** CanClose: Boolean);

**procedure** ExitClick(Sender: TObject);

**procedure** AddClick(Sender: TObject);

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

**procedure** EditClick(Sender: TObject);

**procedure** DeleteClick(Sender: TObject);

**procedure** ShowClick(Sender: TObject);

**procedure** SavetofileClick(Sender: TObject);

**procedure** OpenfileClick(Sender: TObject);

**public**

ProdList: TBaseProd;

NewArr: TBaseProd;

**end**;

**var**

MainForm: TMainForm;

**implementation**

{$R \*.dfm}

**uses** Product, Sorting;

**procedure** TMainForm.AddClick(Sender: TObject);

**begin**

Visible := false;

**with** AddF **as** TAddF **do**

**begin**

PrName.Clear;

PrSort.Clear;

PrPrice.Clear;

Day.Clear;

Month.Clear;

Year.Clear;

Submit.Enabled := false;

Caption := 'Add product';

Submit.Caption := 'Add';

ShowModal;

**end**;

**end**;

**procedure** TMainForm.EditClick(Sender: TObject);

**begin**

**if** (SG.Row <> 0) **then**

**begin**

Visible := false;

**with** AddF **as** TAddF **do**

**begin**

Caption := 'Edit product';

PrName.Text := ProdList[SG.Row - 1].Title;

PrSort.Text := ProdList[SG.Row - 1].Sort;

PrPrice.Text := ProdList[SG.Row - 1].Price;

Day.Text := Copy(ProdList[SG.Row - 1].Date, 1, 2);

Month.Text := Copy(ProdList[SG.Row - 1].Date, 4, 2);

Year.Text := Copy(ProdList[SG.Row - 1].Date, 7, 4);

Submit.Caption := 'Edit';

ShowModal;

**end**;

**end**;

**end**;

**procedure** TMainForm.DeleteClick(Sender: TObject);

**var**

ButtonSelected, i: ShortInt;

**begin**

**if** (SG.Row <> 0) **then**

**begin**

ButtonSelected := MessageDlg('Are you sure you want to delete this product?', mtConfirmation, [mbYes,mbNo], 0);

**if** ButtonSelected = mrYes **then**

**begin**

**for** i := (SG.Row - 1) **to** High(ProdList) - 1 **do**

**begin**

ProdList[i] := ProdList[i + 1];

**with** SG **as** TStringGrid **do**

**begin**

Cells[0, i + 1] := ProdList[i].Title;

Cells[1, i + 1] := ProdList[i].Sort;

Cells[2, i + 1] := ProdList[i].Price;

Cells[3, i + 1] := ProdList[i].Date;

**end**;

**end**;

SetLength(ProdList, Length(ProdList) - 1);

SG.RowCount := Length(ProdList) + 1;

**end**;

**if** SG.RowCount = 1 **then**

**begin**

Delete.Enabled := false;

Edit.Enabled := false;

Savetofile.Enabled := false;

Show.Enabled := false;

**end**;

**end**;

**end**;

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

**begin**

Savetofile.Enabled := false;

Edit.Enabled := false;

Delete.Enabled := false;

Show.Enabled := false;

**with** SG **as** TStringGrid **do**

**begin**

RowCount := 1;

Cells[0, 0] := 'Product title';

Cells[1, 0] := 'Sort';

Cells[2, 0] := 'Price';

Cells[3, 0] := 'Date of delivery';

**end**;

**end**;

**procedure** TMainForm.FormCloseQuery(Sender: TObject; **var** CanClose: Boolean);

**var**

ButtonSelected: Byte;

**begin**

ButtonSelected := MessageDlg('Are you sure you want to exit?', mtConfirmation, [mbYes,mbNo], 0);

**if** ButtonSelected <> mrYes **then**

CanClose := False;

**end**;

**procedure** TMainForm.ExitClick(Sender: TObject);

**begin**

Close;

**end**;

**procedure** TMainForm.DeveloperClick(Sender: TObject);

**begin**

MessageDlg('This program is developed by Maksim Gladkiy.', mtInformation, [mbOk], 0);

**end**;

**procedure** TMainForm.InfoClick(Sender: TObject);

**begin**

MessageDlg('This is a prototype of shop.' + #10#13 + 'You can add products, edit their information and delete them.' + #10#13 + 'Program find''s products of first sort, which were brought last month, and sort them in alphabet order.', mtInformation, [mbOk], 0);

**end**;

**procedure** TMainForm.OpenfileClick(Sender: TObject);

**var**

InputFile: **File of** TProduct;

i, j: Byte;

Temp: SmallInt;

**begin**

**if** Open.Execute **then**

**begin**

**try**

AssignFile(InputFile, Open.FileName);

Reset(InputFile);

i := 0;

**repeat**

SetLength(ProdList, i + 1);

Read(InputFile, ProdList[i]);

Inc(i);

**until** EoF(InputFile);

SG.RowCount := i + 1;

**with** SG **as** TStringGrid **do**

**for** i := 0 **to** High(ProdList) **do**

**begin**

Cells[0, i + 1] := ProdList[i].Title;

Cells[1, i + 1] := ProdList[i].Sort;

Cells[2, i + 1] := ProdList[i].Price;

Cells[3, i + 1] := ProdList[i].Date;

**end**;

CloseFile(InputFile);

Savetofile.Enabled := true;

Edit.Enabled := true;

Delete.Enabled := true;

**if** (SG.RowCount > 1) **then**

Show.Enabled := true

**else**

Show.Enabled := false;

**except**

MessageDlg('Check entered data. Try again.', mtError, [mbRetry], 0);

CloseFile(InputFile);

**end**;

**end**;

**end**;

**procedure** TMainForm.SavetofileClick(Sender: TObject);

**var**

OutputFile: **File of** TProduct;

MyFile: String;

ButtonSelected, i: Byte;

**begin**

**if** Save.Execute **then**

**begin**

MyFile := Save.FileName;

AssignFile(OutputFile, MyFile);

Rewrite(OutputFile);

**for** i := 0 **to** High(ProdList) **do**

Write(OutputFile, ProdList[i]);

CloseFile(OutputFile);

**end**;

**end**;

**procedure** TMainForm.ShowClick(Sender: TObject);

**begin**

Visible := false;

SortF.ShowModal;

**end**;

**end**.

**Product.pas**

**unit** Product;

**interface**

**uses**

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Menus;

**type**

TAddF = **class**(TForm)

Submit: TButton;

AddMenu: TMainMenu;

Help: TMenuItem;

Cancel: TButton;

PrName: TEdit;

Label1: TLabel;

Label2: TLabel;

PrSort: TEdit;

Label3: TLabel;

PrPrice: TEdit;

Label4: TLabel;

Day: TEdit;

Label5: TLabel;

Month: TEdit;

Label6: TLabel;

Year: TEdit;

**procedure** SubmitClick(Sender: TObject);

**procedure** HelpClick(Sender: TObject);

**procedure** CancelClick(Sender: TObject);

**procedure** DayKeyPress(Sender: TObject; **var** Key: Char);

**procedure** DayKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**procedure** MonthKeyPress(Sender: TObject; **var** Key: Char);

**procedure** MonthKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**procedure** YearKeyPress(Sender: TObject; **var** Key: Char);

**procedure** YearKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**procedure** PrPriceKeyPress(Sender: TObject; **var** Key: Char);

**procedure** PrPriceKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**procedure** PrSortKeyPress(Sender: TObject; **var** Key: Char);

**procedure** PrNameKeyPress(Sender: TObject; **var** Key: Char);

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

**procedure** PrChange(Sender: TObject);

**private**

{ Private declarations }

**public**

{ Public declarations }

**end**;

**var**

AddF: TAddF;

**implementation**

{$R \*.dfm}

**uses** Main, Sorting;

**procedure** TAddF.SubmitClick(Sender: TObject);

**begin**

**if** Caption = 'Add product' **then**

**with** MainForm **as** TMainForm **do**

**begin**

SetLength(ProdList, Length(ProdList) + 1);

ProdList[High(ProdList)].Title := PrName.Text;

ProdList[High(ProdList)].Sort := PrSort.Text;

ProdList[High(ProdList)].Price := PrPrice.Text;

ProdList[High(ProdList)].Date := Day.Text + '.' + Month.Text + '.' + Year.Text;

SG.RowCount := SG.RowCount + 1;

SG.Cells[0, Length(ProdList)] := ProdList[High(ProdList)].Title;

SG.Cells[1, Length(ProdList)] := ProdList[High(ProdList)].Sort;

SG.Cells[2, Length(ProdList)] := ProdList[High(ProdList)].Price;

SG.Cells[3, Length(ProdList)] := ProdList[High(ProdList)].Date;

Savetofile.Enabled := true;

**end**

**else**

**with** MainForm **as** TMainForm **do**

**begin**

ProdList[SG.Row - 1].Title := PrName.Text;

ProdList[SG.Row - 1].Sort := PrSort.Text;

ProdList[SG.Row - 1].Price := PrPrice.Text;

ProdList[SG.Row - 1].Date := Day.Text + '.' + Month.Text + '.' + Year.Text;

SG.RowCount := SG.RowCount + 1;

SG.Cells[0, SG.Row] := ProdList[SG.Row - 1].Title;

SG.Cells[1, SG.Row] := ProdList[SG.Row - 1].Sort;

SG.Cells[2, SG.Row] := ProdList[SG.Row - 1].Price;

SG.Cells[3, SG.Row] := ProdList[SG.Row - 1].Date;

**end**;

MainForm.Edit.Enabled := true;

MainForm.Delete.Enabled := true;

MainForm.Visible := true;

**end**;

**procedure** TAddF.HelpClick(Sender: TObject);

**begin**

MessageDlg('Here you can add products to the list.' + #10#13 + 'Name length is [1..15].' + #10#13 + 'Sort is [1..4].' + #10#13 + 'Price is [1..2000].' + #10#13 + 'Date: Day [1..31], Month [1..12], Year [2014..2019].', mtInformation, [mbOk], 0);

**end**;

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

**begin**

MainForm.Visible := true;

**end**;

**procedure** TAddF.CancelClick(Sender: TObject);

**begin**

MainForm.Visible := true;

**end**;

**procedure** TAddF.PrChange(Sender: TObject);

**begin**

**if** (Length(PrName.Text) > 0)

**and** (Length(PrSort.Text) = 1)

**and** (Length(PrPrice.Text) > 0)

**and** (Length(Day.Text) = 2)

**and** (Length(Month.Text) = 2)

**and** (Length(Year.Text) = 4)

**then**

Submit.Enabled := true

**else**

Submit.Enabled := false;

**end**;

**procedure** TAddF.PrNameKeyPress(Sender: TObject; **var** Key: Char);

**var**

Letters: **set of** Char;

**begin**

Letters := ['A'..'z', '.', '-', ' ', #8];

**with** PrName **as** TEdit **do**

**begin**

**if not**(Key **in** Letters) **then**

Key := #0;

**if** (Length(Text) <> SelStart) **then**

Key := #0;

**if** (Length(Text) = 15) **and** (Key <> #8)**then**

Key := #0

**end**;

**end**;

**procedure** TAddF.PrSortKeyPress(Sender: TObject; **var** Key: Char);

**var**

Numerals: **set of** Char;

**begin**

Numerals := ['1'..'4', #8];

**with** PrSort **as** TEdit **do**

**begin**

**if not**(Key **in** Numerals) **then**

Key := #0;

**if** (Length(Text) <> SelStart) **then**

Key := #0;

**if** (Length(Text) = 1) **and** (Key <> #8)**then**

Key := #0;

**end**;

**end**;

**procedure** TAddF.PrPriceKeyPress(Sender: TObject; **var** Key: Char);

**var**

Numerals: **set of** Char;

**begin**

Numerals := ['0'..'9', #8];

**with** PrPrice **as** TEdit **do**

**begin**

**if not**(Key **in** Numerals) **then**

Key := #0;

**if** (Length(Text) <> SelStart) **then**

Key := #0;

**if** (Length(Text) = 0) **and** (Key = '0')**then**

Key := #0;

**if** (Length(Text) = 4) **and** (Key <> #8)**then**

Key := #0;

**end**;

**end**;

**procedure** TAddF.PrPriceKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**begin**

**with** PrPrice **as** Tedit **do**

**begin**

**if** (Length(Text) = 4) **and** ((StrToInt(Text) > 2000) **or** (StrToInt(Text) < 1)) **then**

Text := '';

**end**;

**end**;

**procedure** TAddF.DayKeyPress(Sender: TObject; **var** Key: Char);

**var**

Numerals: **set of** Char;

**begin**

Numerals := ['0'..'9', #8];

**with** Day **as** TEdit **do**

**begin**

**if not**(Key **in** Numerals) **then**

Key := #0;

**if** (Length(Text) <> SelStart) **then**

Key := #0;

**if** (Length(Text) = 2) **and** (Key <> #8)**then**

Key := #0;

**end**;

**end**;

**procedure** TAddF.DayKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**begin**

**with** Day **as** Tedit **do**

**begin**

**if** (Length(Text) = 2) **and** ((StrToInt(Text) > 31) **or** (StrToInt(Text) = 0)) **then**

Text := '';

**end**;

**end**;

**procedure** TAddF.MonthKeyPress(Sender: TObject; **var** Key: Char);

**var**

Numerals: **set of** Char;

**begin**

Numerals := ['0'..'9', #8];

**with** Month **as** TEdit **do**

**begin**

**if not**(Key **in** Numerals) **then**

Key := #0;

**if** (Length(Text) <> SelStart) **then**

Key := #0;

**if** (Length(Text) = 2) **and** (Key <> #8)**then**

Key := #0;

**end**;

**end**;

**procedure** TAddF.MonthKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**begin**

**with** Month **as** Tedit **do**

**begin**

**if** (Length(Text) = 2) **and** ((StrToInt(Text) > 12) **or** (StrToInt(Text) = 0)) **then**

Text := '';

**end**;

**end**;

**procedure** TAddF.YearKeyPress(Sender: TObject; **var** Key: Char);

**var**

Numerals: **set of** Char;

**begin**

Numerals := ['0'..'9', #8];

**with** Year **as** TEdit **do**

**begin**

**if not**(Key **in** Numerals) **then**

Key := #0;

**if** (Length(Text) <> SelStart) **then**

Key := #0;

**if** (Length(Text) = 4) **and** (Key <> #8)**then**

Key := #0;

**end**;

**end**;

**procedure** TAddF.YearKeyUp(Sender: TObject; **var** Key: Word; Shift: TShiftState);

**begin**

**with** Year **as** Tedit **do**

**begin**

**if** (Length(Text) = 4) **and** ((StrToInt(Text) > 2019) **or** (StrToInt(Text) < 2014)) **then**

Text := '';

**end**;

**end**;

**end**.

**Sorting.pas**

**unit** Sorting;

**interface**

**uses**

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.Grids, Vcl.StdCtrls;

**type**

TSortF = **class**(TForm)

SGR: TStringGrid;

SortMenu: TMainMenu;

Help: TMenuItem;

Close: TButton;

Save: TSaveDialog;

SavetoFileS: TMenuItem;

**procedure** HelpClick(Sender: TObject);

**procedure** CloseClick(Sender: TObject);

**procedure** FormShow(Sender: TObject);

**procedure** SavetoFileSClick(Sender: TObject);

**private**

**public**

**end**;

**var**

SortF: TSortF;

**implementation**

{$R \*.dfm}

**uses** Main, Product;

**procedure** TSortF.CloseClick(Sender: TObject);

**begin**

MainForm.Visible := true;

**end**;

**procedure** TSortF.FormShow(Sender: TObject);

**var**

i, j: Byte;

Temp: TProduct;

**begin**

**with** SGR **as** TStringGrid **do**

**begin**

RowCount := 10;

Cells[0, 0] := 'Product title';

Cells[1, 0] := 'Sort';

Cells[2, 0] := 'Price';

Cells[3, 0] := 'Date of delivery';

**end**;

**With** MainForm **as** TMainForm **do**

**begin**

j := 0;

SetLength(NewArr, High(ProdList) + 1);

**for** i := 0 **to** High(ProdList) **do**

**if** (ProdList[i].Date[4] = '0')**and**(ProdList[i].Date[5] = '1')**and**(ProdList[i].Date[7] = '2')**and**(ProdList[i].Date[8] = '0')**and**(ProdList[i].Date[9] = '1')**and**(ProdList[i].Date[10] = '9')**and**(ProdList[i].Sort = '1') **then**

**begin**

NewArr[j] := ProdList[i];

inc(j);

**end**;

**for** i := 1 **to** j - 1 **do**

**if** NewArr[i - 1].Title[1] > NewArr[i].Title[1] **then**

**begin**

Temp := NewArr[i];

NewArr[i] := NewArr[i - 1];

NewArr[i - 1] := Temp;

**end**;

**end**;

**with** SGR **as** TStringGrid **do**

**begin**

**for** i := 0 **to** j **do**

**begin**

Cells[0, i + 1] := MainForm.NewArr[i].Title;

Cells[1, i + 1] := MainForm.NewArr[i].Sort;

Cells[2, i + 1] := MainForm.NewArr[i].Price;

Cells[3, i + 1] := MainForm.NewArr[i].Date;

**end**;

**end**;

**end**;

**procedure** TSortF.HelpClick(Sender: TObject);

**begin**

MessageDlg('This page shows products of 1 sort, which were brought last month, in alphabet order.', mtInformation, [mbOk], 0);

**end**;

**procedure** TSortF.SavetoFileSClick(Sender: TObject);

**var**

OutputFile: **File of** TProduct;

MyFile: String;

ButtonSelected, i: Byte;

**begin**

**if** Save.Execute **then**

**begin**

MyFile := Save.FileName;

AssignFile(OutputFile, MyFile);

Rewrite(OutputFile);

**for** i := 0 **to** High(MainForm.NewArr) **do**

Write(OutputFile, MainForm.NewArr[i]);

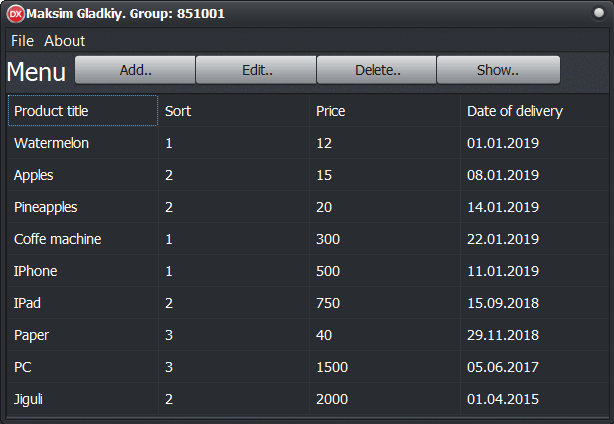
CloseFile(OutputFile);

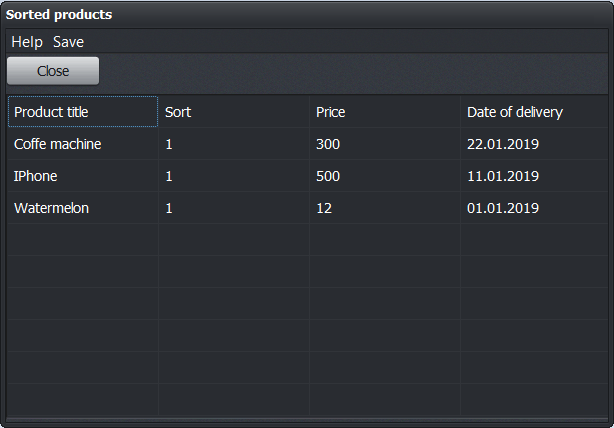
**end**;

**end**;

**end**.

**Скриншоты:**

****

****

**Блок-схема:**







