УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №4.1

по предмету «Основы алгоритмизации и программирования»

Вариант 13

Выполнил:

Машевский Д.В

Гр. 351003

Проверил:

Данилова Г. В.

Минск 2023

**Задание:**

Сведения об ассортименте игрушек в магазине:

* название игрушки;
* цена;
* количество;
* возрастные данные, например 2-5, т.е. от 2 до 5 лет.

Вывести названия игрушек, которые подходят детям от 1 до 3 лет; стоимость самой дорогой игрушки и ее наименование; название игрушки, которая по стоимости не превышает "х" руб. и подходит ребенку в возрасте от "а" до "б" лет. Значения “x” и “a” вводятся с клавиатуры.

**Код программы Delphi:**

**unit Unit1Laba41;**

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

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

Vcl.ExtCtrls;

Const

MAXPENNY = 99;

MAX\_AGE = 150;

Type

TCost = Record

Ruble: 0 .. 9999;

Penny: 0 .. 99;

End;

TGame = Record

Name: String[25];

Sort: String[15];

Cost: TCost;

Number: Integer;

MinAge: Integer;

MaxAge: Integer;

End;

TBorderArr = Array [0 .. 1] Of Byte;

TArr = Array of TGame;

TForm1 = class(TForm)

MainMenuForm: TMainMenu;

NFile: TMenuItem;

NInstruction: TMenuItem;

NDeveloper: TMenuItem;

NOpen: TMenuItem;

NSave: TMenuItem;

LabelNameForm: TLabel;

BtnAdd: TButton;

BtnDelete: TButton;

BtnInfo13: TButton;

ShopWidow: TStringGrid;

SaveFileDlg: TSaveDialog;

OpenFileDlg: TOpenDialog;

BtnUserSearch: TButton;

EditYourMinAge: TEdit;

EditYourMaxAge: TEdit;

Label1FormUserRecAge: TLabel;

LabelFromAge: TLabel;

LabelToAge: TLabel;

ShapeMinAge: TShape;

ShapeMaxAge: TShape;

EditYourCost: TEdit;

Label1: TLabel;

ShapeYourCost: TShape;

procedure FormCreate(Sender: TObject);

procedure BtnAddClick(Sender: TObject);

procedure BtnDeleteClick(Sender: TObject);

procedure FormClick(Sender: TObject);

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

procedure BtnInfo13Click(Sender: TObject);

procedure NSaveClick(Sender: TObject);

Procedure CheckEnter(Sender: TObject; var Key: Char);

procedure NOpenClick(Sender: TObject);

procedure NDeveloperClick(Sender: TObject);

procedure NInstructionClick(Sender: TObject);

procedure ShopWidowDblClick(Sender: TObject);

procedure ShopWidowDrawCell(Sender: TObject; ACol, ARow: Integer;

Rect: TRect; State: TGridDrawState);

procedure BtnUserSearchClick(Sender: TObject);

procedure ShopWidowKeyPress(Sender: TObject; var Key: Char);

procedure FormKeyPress(Sender: TObject; var Key: Char);

procedure EditYourMinAgeChange(Sender: TObject);

procedure EditYourMaxAgeChange(Sender: TObject);

procedure EditYourMinAgeKeyPress(Sender: TObject; var Key: Char);

Function InputBorders(): Boolean;

procedure EditYourMaxAgeKeyPress(Sender: TObject; var Key: Char);

procedure EditYourMinAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditYourMaxAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditYourCostChange(Sender: TObject);

Procedure CheckIsFull(Sender: TObject);

procedure EditYourCostKeyPress(Sender: TObject; var Key: Char);

private

{ Private declarations }

Saved: Boolean;

IsFull: Boolean;

public

{ Public declarations }

Procedure UpdateGrid();

Procedure CheckEdit(Var IsCorrect, IsFull: Boolean);

end;

var

form1: TForm1;

Games: TArr;

implementation

{$R \*.dfm}

uses Unit1, Unit4Laba41, Unit5, Unit5Laba41, Unit6Laba41;

Function CostToStr(Cost: TCost): String;

Begin

CostToStr := IntToStr(Cost.Ruble) + 'р. ' + IntToStr(Cost.Penny) + 'коп.';

End;

Procedure TForm1.UpdateGrid();

Var

I, J: Integer;

Begin

For I := 1 To ShopWidow.RowCount Do

For J := 0 To ShopWidow.ColCount Do

ShopWidow.Cells[J, I] := '';

If (Length(Games) <> 0) Then

Begin

BtnDelete.Enabled := True;

BtnInfo13.Enabled := True;

NSave.Enabled := True;

Saved := False;

For I := 1 To Length(Games) Do

Begin

ShopWidow.Cells[0, I] := Games[I - 1].Name;

ShopWidow.Cells[1, I] := Games[I - 1].Sort;

ShopWidow.Cells[2, I] := IntToStr(Games[I - 1].Number);

ShopWidow.Cells[3, I] := CostToStr(Games[I - 1].Cost);

ShopWidow.Cells[4, I] := IntToStr(Games[I - 1].MinAge);

ShopWidow.Cells[5, I] := IntToStr(Games[I - 1].MaxAge);

End;

End

Else

Begin

BtnDelete.Enabled := False;

BtnInfo13.Enabled := False;

NSave.Enabled := False;

Saved := True;

BtnUserSearch.Enabled := False;

End;

End;

Function CheckOnSimilar(NewGame: TGame): Boolean;

Var

I: Integer;

IsSimilar: Boolean;

Begin

IsSimilar := False;

For I := Low(Games) To High(Games) Do

If (NewGame.Name = Games[I].Name) And (NewGame.Sort = Games[I].Sort) And

(NewGame.Number = Games[I].Number) And

(NewGame.Cost.Ruble = Games[I].Cost.Ruble) And

(NewGame.Cost.Penny = Games[I].Cost.Penny) And

(NewGame.MinAge = Games[I].MinAge) and

(NewGame.MaxAge = Games[I].MaxAge) Then

IsSimilar := True;

CheckOnSimilar := IsSimilar;

End;

Function CreateGame(Dlg: TForm4): TGame;

Var

Game: TGame;

Begin

Game.Name := Dlg.EditName.Text;

Game.Sort := Dlg.EditSort.Text;

Game.Number := StrToInt(Dlg.EditNumber.Text);

Game.Cost.Ruble := StrToInt(Dlg.EditRuble.Text);

Game.Cost.Penny := StrToInt(Dlg.EditPenny.Text);

Game.MinAge := StrToInt(Dlg.EditMinAge.Text);

Game.MaxAge := StrToInt(Dlg.EditMaxAge.Text);

CreateGame := Game;

End;

Procedure TForm1.CheckEnter(Sender: TObject; var Key: Char);

var

chars: set of Char;

editText: String;

begin

editText := (Sender as TEdit).Text;

if (Length(editText) = 0) then

begin

chars := ['0' .. '9', #8, #31];

end

else

chars := ['0' .. '9', #8, #31];

if not(Key in chars) then

Key := #0;

end;

Procedure TForm1.FormClick(Sender: TObject);

Begin

form1.FocusControl(nil);

End;

procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

Var

DlgResult: Integer;

Begin

If (Not Saved) then

Begin

DlgResult := Application.MessageBox

('Имеются несохранённые данные. Сохранить их перед выходом ' +

'из программы?', 'Сохранение', MB\_YESNO);

If (DlgResult = IDYES) then

NSaveClick(Sender)

Else If (DlgResult = IDNO) then

Saved := False;

End;

CanClose := MessageBox(Handle, 'Вы действительно хотите выйти?', 'Внимание',

MB\_OKCANCEL) = mrOk;

End;

Function TForm1.InputBorders(): Boolean;

var

Borders: TBorderArr;

IsGood: Boolean;

Begin

Borders[0] := StrToInt(EditYourMinAge.Text);

Borders[1] := StrToInt(EditYourMaxAge.Text);

IsGood := True;

If (Borders[0] > Borders[1]) Then

Begin

BtnUserSearch.Enabled := False;

IsGood := False;

End;

InputBorders := IsGood;

End;

procedure TForm1.FormCreate(Sender: TObject);

Var

I: Integer;

begin

Saved := True;

BtnDelete.Enabled := False;

BtnInfo13.Enabled := False;

BtnUserSearch.Enabled := False;

MainMenuForm.Items[0].Items[1].Enabled := False;

ShopWidow.Cells[0, 0] := 'Название';

ShopWidow.Cells[1, 0] := 'Назначение';

ShopWidow.Cells[2, 0] := 'Инв-ый номер';

ShopWidow.Cells[3, 0] := 'Цена за ед.';

ShopWidow.Cells[4, 0] := 'Мин. возраст';

ShopWidow.Cells[5, 0] := 'Макс. возраст';

for I := 0 to 5 do

ShopWidow.ColWidths[I] := 125;

end;

Procedure TForm1.FormKeyPress(Sender: TObject; var Key: Char);

Begin

If Key = #13 Then

BtnAdd.Click();

End;

procedure TForm1.NDeveloperClick(Sender: TObject);

CONST

FIRST\_MESSAGE =

'Машевский Даниил Витальевич, группа 351003, Лабораторная № 4.1'#10'Сведения об

ассортименте игрушек в магазине:'#10'- Название игрушки;'#10'- Цена;'#10'-

Количество;'#10'- Возрастные данные, например 2-5, т.е. от 2 до 5 лет.;';

SECOND\_MESSAGE =

'Вывести названия игрушек, которые подходят детям от 1 до 3 лет; стоимость

самой дорогой игрушки и ее наименование; название игрушки, которая по стоим.

не превышает "х" руб. и подходит ребенку в возр. от "а" до "б" лет. Значения

“x” и “a” вводятся с клав.';

Begin

Application.MessageBox(FIRST\_MESSAGE + SECOND\_MESSAGE, 'О разработчике');

end;

Procedure TForm1.NInstructionClick(Sender: TObject);

Const

FIRST\_MESSAGE = '1. Для добавления информации о приборе нажмите на кнопку '

+ '"Добавить игрушку" и заполните все поля. В случае ошибки ' +

'вы увидите соответствующее сообщение.' + #13#10;

SECOND\_MESSAGE =

'2. Для удаления информации о игрушке выделите строку/строки, ' +

'которые необходимо удалить, и нажмите на кнопку "Удалить игрушку". ' +

#13#10;

THIRD\_MESSAGE =

'3. Для вывода информации о рекомендуемых игрушках от 1 до 3 лет, и

наименование и стоимость самой дорогой игрушки '

+ 'нажмите на кнопку "Справка 1-3".' + #13#10;

FOURTH\_MESSAGE =

'4. Для редактирования информации о игрушке дважды щёлкните по нужной строке '

+ 'и исправьте необходимую информацию. ' + #13#10 + 'Горячие клавиши:' +

#13#10 + 'Главное окно: Enter - Добавить игрушку; Backspace - Удалить прибор'

+ #13#10 +

'Окно добавления/редактирования информации: Enter - кнопка "Ок"';

Begin

Application.MessageBox(FIRST\_MESSAGE + SECOND\_MESSAGE + THIRD\_MESSAGE +

FOURTH\_MESSAGE, 'Инструкция');

End;

procedure TForm1.NOpenClick(Sender: TObject);

Var

ReadFile: File Of TGame;

Game: TGame;

DlgResult: Integer;

Begin

If (Not Saved) then

Begin

DlgResult := Application.MessageBox

('Имеются несохранённые данные. Сохранить их перед открытием ' +

'нового файла?', 'Сохранение', MB\_YESNOCANCEL + MB\_ICONINFORMATION);

If (DlgResult = IDYES) then

NSaveClick(Sender)

Else If (DlgResult = IDNO) then

Saved := True;

End;

If (Saved) and (OpenFileDlg.Execute) then

Begin

Try

AssignFile(ReadFile, OpenFileDlg.FileName);

Reset(ReadFile);

Games := nil;

ShopWidow.RowCount := 1;

While Not(EOF(ReadFile)) do

Begin

Read(ReadFile, Game);

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

Games[Length(Games) - 1] := Game;

ShopWidow.RowCount := ShopWidow.RowCount + 1;

End;

Except

Application.MessageBox('Ошибка при извлечении данных! Измените ' +

'параметры или содержимое файла! ', 'Ошибка!',

MB\_ICONERROR);

End;

CloseFile(ReadFile);

UpdateGrid();

NSave.Enabled := False;

Saved := True;

MainMenuForm.Items[0].Items[1].Enabled := False;

if (Length(EditYourCost.Text) <> 0) and

(Length(EditYourMinAge.Text) <> 0) and

(Length(EditYourMinAge.Text) <> 0) then

BtnUserSearch.Enabled := True;

End

End;

procedure TForm1.NSaveClick(Sender: TObject);

Var

SaveFile: File of TGame;

I: Integer;

Begin

If (SaveFileDlg.Execute and FileExists(SaveFileDlg.FileName)) then

Begin

AssignFile(SaveFile, SaveFileDlg.FileName);

Try

Rewrite(SaveFile);

Except

Application.MessageBox

('Отказано в доступе! Измените параметры файла!', 'Ошибка!',

MB\_ICONERROR);

End;

For I := 0 to High(Games) do

Write(SaveFile, Games[I]);

CloseFile(SaveFile);

Saved := True;

Application.MessageBox('Данные успешно записаны в файл!', 'Сохранение',

MB\_ICONINFORMATION);

End;

End;

Procedure EditInformation(Dlg: TForm4; Game: TGame);

Begin

Dlg.EditName.Text := Game.Name;

Dlg.EditSort.Text := Game.Sort;

Dlg.EditNumber.Text := IntToStr(Game.Number);

Dlg.EditRuble.Text := IntToStr(Game.Cost.Ruble);

Dlg.EditPenny.Text := IntToStr(Game.Cost.Penny);

Dlg.EditMinAge.Text := IntToStr(Game.MinAge);

Dlg.EditMaxAge.Text := IntToStr(Game.MaxAge);

End;

Procedure TForm1.ShopWidowDblClick(Sender: TObject);

Var

Dlg: TForm4;

Begin

If ShopWidow.Row <> 0 Then

Try

Dlg := TForm4.Create(Self);

Dlg.Caption := 'Редактирование информации';

EditInformation(Dlg, Games[ShopWidow.Row - 1]);

Dlg.ShowModal();

If (Dlg.ModalResult = mrOk) then

Begin

If Not((Games[ShopWidow.Row - 1].Name = Dlg.EditName.Text) and

(Games[ShopWidow.Row - 1].Sort = Dlg.EditSort.Text) and

(Games[ShopWidow.Row - 1].Number = StrToInt

(Dlg.EditNumber.Text)) and

(Games[ShopWidow.Row - 1].Cost.Ruble = StrToInt

(Dlg.EditRuble.Text)) and

(Games[ShopWidow.Row - 1].Cost.Penny = StrToInt

(Dlg.EditPenny.Text)) and

(Games[ShopWidow.Row - 1].MinAge = StrToInt

(Dlg.EditMinAge.Text)) and

(Games[ShopWidow.Row - 1].MaxAge = StrToInt

(Dlg.EditMaxAge.Text))) Then

Begin

Games[ShopWidow.Row - 1].Name := Dlg.EditName.Text;

Games[ShopWidow.Row - 1].Sort := Dlg.EditSort.Text;

Games[ShopWidow.Row - 1].Number :=

StrToInt(Dlg.EditNumber.Text);

Games[ShopWidow.Row - 1].Cost.Ruble :=

StrToInt(Dlg.EditRuble.Text);

Games[ShopWidow.Row - 1].Cost.Penny :=

StrToInt(Dlg.EditPenny.Text);

Games[ShopWidow.Row - 1].MinAge :=

StrToInt(Dlg.EditMinAge.Text);

Games[ShopWidow.Row - 1].MaxAge :=

StrToInt(Dlg.EditMaxAge.Text);

UpdateGrid();

End;

End;

Finally

Dlg.Free();

End;

End;

procedure TForm1.ShopWidowDrawCell(Sender: TObject; ACol, ARow: Integer;

Rect: TRect; State: TGridDrawState);

begin

if (ARow in [0]) then

begin

TStringGrid(Sender).Canvas.Brush.Color := clYellow;

TStringGrid(Sender).Canvas.FillRect(Rect);

TStringGrid(Sender).Canvas.Font.Color := clGreen;

TStringGrid(Sender).Canvas.TextOut(Rect.Left + 1, Rect.Top + 1,

TStringGrid(Sender).Cells[ACol, ARow]);

end;

end;

Procedure TForm1.ShopWidowKeyPress(Sender: TObject; var Key: Char);

Begin

If (Key = #8) And (BtnDelete.Enabled = True) Then

BtnDelete.Click();

If Key = #13 Then

BtnAdd.Click();

End;

Procedure TForm1.BtnAddClick(Sender: TObject);

Var

Dlg: TForm4;

NewProduct: TGame;

Begin

Try

Dlg := TForm4.Create(Self);

Dlg.Caption := 'Добавление информации';

Dlg.ShowModal();

If (Dlg.ModalResult = mrOk) Then

Begin

NewProduct := CreateGame(Dlg);

If Not CheckOnSimilar(NewProduct) Then

Begin

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

Games[Length(Games) - 1] := NewProduct;

ShopWidow.RowCount := ShopWidow.RowCount + 1;

UpdateGrid();

Saved := False;

End

Else

Application.MessageBox('Данный продукт уже добавлен!', 'Ошибка',

MB\_ICONINFORMATION);

End;

Finally

Dlg.Free();

MainMenuForm.Items[0].Items[1].Enabled := True;

End;

End;

Function Swap(var Toy1, Toy2: TGame): TGame;

var

Temp: TGame;

begin

Temp := Toy1;

Toy1 := Toy2;

Toy2 := Temp;

Swap := Temp;

end;

function SortToysByAgeRestriction(): TArr;

var

I, J: Integer;

SortedProducts: TArr;

begin

SortedProducts := Games;

for I := 0 to Length(SortedProducts) - 2 do

for J := 0 to Length(SortedProducts) - 2 - I do

if (SortedProducts[J].MinAge > SortedProducts[J + 1].MinAge) or

((SortedProducts[J].MinAge = SortedProducts[J + 1].MinAge) and

(SortedProducts[J].MaxAge > SortedProducts[J + 1].MaxAge)) then

Swap(SortedProducts[J], SortedProducts[J + 1]);

SortToysByAgeRestriction := SortedProducts;

end;

Procedure TForm1.BtnDeleteClick(Sender: TObject);

Var

I, J, DlgResult, StartI, EndI: Integer;

Begin

If ShopWidow.Selection.Top = 0 Then

Application.MessageBox('Выделите необходимый товар!', 'Ошибка',

MB\_ICONERROR)

Else

Begin

DlgResult := Application.MessageBox

('Вы уверены, что хотите удалить товар?', 'Удаление товара',

MB\_YESNO + MB\_ICONQUESTION);

If DlgResult = IDYES Then

Begin

If (ShopWidow.Selection.Top <> ShopWidow.Selection.Bottom) then

Begin

StartI := ShopWidow.Selection.Top - 1;

EndI := ShopWidow.Selection.Bottom - 1;

For I := StartI To EndI Do

Begin

For J := StartI to (High(Games) - 1) do

Games[J] := Games[J + 1];

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

ShopWidow.RowCount := ShopWidow.RowCount - 1;

End;

UpdateGrid();

End

Else

Begin

If (ShopWidow.Selection.Top = ShopWidow.RowCount - 1) Then

Begin

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

UpdateGrid();

End

Else

Begin

For I := (ShopWidow.Selection.Top - 1)

To (High(Games) - 1) Do

Games[I] := Games[I + 1];

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

End;

ShopWidow.RowCount := ShopWidow.RowCount - 1;

UpdateGrid();

End;

End

End;

End;

procedure TForm1.BtnInfo13Click(Sender: TObject);

var

Dlg: TForm5;

SortedToys: TArr;

I, Row: Integer;

MaxCost: Integer;

Name: String;

begin

Row := 1;

MaxCost := 0;

SortedToys := SortToysByAgeRestriction();

if Length(SortedToys) <> 0 then

begin

try

Dlg := TForm5.Create(Self);

Dlg.ShopInfo.ColCount := Length(SortedToys) + 1;

Dlg.ShopInfo.FixedRows := 1;

for I := 0 to Length(SortedToys) - 1 do

begin

if (SortedToys[I].MinAge >= 1) and (SortedToys[I].MaxAge <= 3)

then

begin

Dlg.ShopInfo.Cells[Row, 0] := SortedToys[I].Name;

Inc(Row);

end

else

Dlg.ShopInfo.ColCount := Dlg.ShopInfo.ColCount - 1;

if (SortedToys[I].Cost.Ruble \* 100 + SortedToys[I].Cost.Penny) >

MaxCost

then

begin

MaxCost := SortedToys[I].Cost.Ruble \* 100 + SortedToys[I]

.Cost.Penny;

Name := SortedToys[I].Name;

end;

end;

Dlg.LabelNameMaxCostGame.Caption := Name;

Dlg.LabelMaxCostGameCost.Caption := IntToStr(MaxCost div 100) +

'руб.' + IntToStr(MaxCost mod 100) + 'коп.';

Dlg.ShowModal();

finally

Dlg.Free();

end;

end

else

Application.MessageBox

('Среди данных игрушек нет игр, рекомендуемых детям от 1 до 3 лет',

'Сведения отсутствуют', MB\_ICONINFORMATION);

end;

procedure TForm1.BtnUserSearchClick(Sender: TObject);

var

Dlg: TForm6;

SortedToys: TArr;

I, Row: Integer;

IsCorrect, IsFull, IsGood: Boolean;

begin

IsGood := InputBorders;

InputBorders();

CheckEdit(IsCorrect, IsFull);

if IsGood = False then

Begin

ShowMessage

('Ошибка. Первое значение границы не может быть больше второго.');

EditYourMinAge.Text := '';

EditYourMaxAge.Text := '';

End

Else

Begin

if IsFull And IsCorrect then

Begin

SortedToys := SortToysByAgeRestriction();

if Length(SortedToys) <> 0 then

begin

try

Dlg := TForm6.Create(Self);

Dlg.ShopForChildren.RowCount := Length(SortedToys) + 1;

Dlg.ShopForChildren.FixedRows := 1;

Row := 1;

for I := 0 to Length(SortedToys) - 1 do

begin

if (SortedToys[I].MinAge >=

StrToInt(EditYourMinAge.Text)) and

(SortedToys[I].MaxAge <= StrToInt(EditYourMaxAge.Text)

) and (SortedToys[I].Cost.Ruble <=

StrToInt(EditYourCost.Text)) then

begin

Dlg.ShopForChildren.Cells[0, Row] :=

SortedToys[I].Name;

Dlg.ShopForChildren.Cells[1, Row] :=

IntToStr(SortedToys[I].MinAge);

Dlg.ShopForChildren.Cells[2, Row] :=

IntToStr(SortedToys[I].MaxAge);

Dlg.ShopForChildren.Cells[3, Row] :=

IntToStr(SortedToys[I].Cost.Ruble);

Inc(Row);

end

Else

Dlg.ShopForChildren.RowCount :=

Dlg.ShopForChildren.RowCount - 1;

end;

Dlg.ShowModal();

finally

Dlg.Free();

end;

end;

End;

End;

end;

Function CheckZeroStr(Str: String): Boolean;

Var

I, Len: Integer;

Fl: Boolean;

Begin

Fl := False;

Len := Length(Str) - 1;

For I := 1 to Len do

Begin

If (Str[I] = '0') and (I <> High(Str)) then

Begin

If (Str[I + 1] <> '') then

Fl := True;

End;

End;

CheckZeroStr := Fl;

End;

Procedure TForm1.CheckEdit(Var IsCorrect, IsFull: Boolean);

Begin

If (ShapeMinAge.Pen.Color = clGreen) and

(ShapeMaxAge.Pen.Color = clGreen) Then

IsCorrect := True

Else

IsCorrect := False;

End;

Procedure TForm1.EditYourCostChange(Sender: TObject);

Var

SortedToys: TArr;

Begin

SortedToys := SortToysByAgeRestriction();

If Length(EditYourCost.Text) = 0 Then

Begin

IsFull := False;

ShapeYourCost.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeYourCost.Pen.Color := clGreen;

End;

CheckIsFull(Sender);

End;

procedure TForm1.EditYourCostKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditYourCost.Text)) Then

EditYourCost.Text := '';

If TryStrToInt(EditYourCost.Text + Key, Value) Then

If (Value > MAX\_RUBLE) Then

Key := #0;

if EditYourCost.Text = '0' then

Begin

EditYourCost.Text := '';

End;

if (Key = '0') and (Length(EditYourCost.Text) = 0) then

Key := #0;

end;

Procedure TForm1.CheckIsFull(Sender: TObject);

Var

SortedToys: TArr;

Begin

SortedToys := SortToysByAgeRestriction();

If (Length(EditYourCost.Text) <> 0) And (Length(EditYourMinAge.Text) <> 0)

And (Length(EditYourMaxAge.Text) <> 0) and (Length(SortedToys) <> 0) Then

Begin

BtnUserSearch.Enabled := True;

End

Else

BtnUserSearch.Enabled := False;

End;

Procedure TForm1.EditYourMaxAgeChange(Sender: TObject);

Var

SortedToys: TArr;

Begin

SortedToys := SortToysByAgeRestriction();

If Length(EditYourMaxAge.Text) = 0 Then

Begin

IsFull := False;

ShapeMaxAge.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeMaxAge.Pen.Color := clGreen;

End;

CheckIsFull(Sender);

End;

procedure TForm1.EditYourMaxAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

If (Key = VK\_INSERT) and (SsShift in Shift) then

Key := 0

end;

procedure TForm1.EditYourMaxAgeKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditYourMaxAge.Text)) Then

EditYourMaxAge.Text := '';

If TryStrToInt(EditYourMaxAge.Text + Key, Value) Then

If (Value > MAX\_AGE) Then

Key := #0;

if EditYourMaxAge.Text = '0' then

Begin

EditYourMaxAge.Text := '';

End;

end;

procedure TForm1.EditYourMinAgeChange(Sender: TObject);

Begin

BtnUserSearch.Enabled := False;

If Length(EditYourMinAge.Text) = 0 Then

Begin

IsFull := False;

ShapeMinAge.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeMinAge.Pen.Color := clGreen;

End;

CheckIsFull(Sender);

End;

procedure TForm1.EditYourMinAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

If (Key = VK\_INSERT) and (SsShift in Shift) then

Key := 0

end;

procedure TForm1.EditYourMinAgeKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditYourMinAge.Text)) Then

EditYourMinAge.Text := '';

If TryStrToInt(EditYourMinAge.Text + Key, Value) Then

If (Value > MAX\_AGE) Then

Key := #0;

if EditYourMinAge.Text = '0' then

Begin

EditYourMinAge.Text := '';

End;

end;

end.

**unit Unit4Laba41;**

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

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

Const

MAX\_PENNY = 99;

MAX\_RUBLE = 9999;

MAX\_AGE = 99;

type

TForm4 = class(TForm)

EditName: TEdit;

EditSort: TEdit;

EditRuble: TEdit;

EditPenny: TEdit;

EditMinAge: TEdit;

EditNumber: TEdit;

ShapeName: TShape;

ShapeSort: TShape;

ShapeNumber: TShape;

ShapeRuble: TShape;

ShapePenny: TShape;

ShapeMinAge: TShape;

EditMaxAge: TEdit;

ShapeMaxAge: TShape;

BtnOk: TButton;

ButtonCancel: TButton;

LabelNameGame: TLabel;

LabelPurposeGame: TLabel;

LabelNumberGame: TLabel;

LabelCostGame: TLabel;

LabelFromRuble: TLabel;

LabelToPenny: TLabel;

LabelRecommendedAge: TLabel;

LabelFromMinAge: TLabel;

LabelToMaxAge: TLabel;

LabelYears: TLabel;

procedure BtnOkClick(Sender: TObject);

procedure EditNameChange(Sender: TObject);

procedure EditNameKeyPress(Sender: TObject; var Key: Char);

procedure EditPennyChange(Sender: TObject);

procedure EditPennyKeyPress(Sender: TObject; var Key: Char);

procedure EditRubleChange(Sender: TObject);

procedure EditRubleKeyPress(Sender: TObject; var Key: Char);

procedure EditSortChange(Sender: TObject);

procedure EditNumberChange(Sender: TObject);

procedure EditNumberKeyPress(Sender: TObject; var Key: Char);

procedure EditSortKeyPress(Sender: TObject; var Key: Char);

procedure EditMinAgeChange(Sender: TObject);

procedure EditMaxAgeChange(Sender: TObject);

procedure ButtonCancelClick(Sender: TObject);

Procedure CheckEnter(Sender: TObject; var Key: Char);

procedure EditMaxAgeKeyPress(Sender: TObject; var Key: Char);

procedure EditMinAgeKeyPress(Sender: TObject; var Key: Char);

Procedure CheckShiftInsert(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditNameKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditSortKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditNumberKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditRubleKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditPennyKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditMinAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

procedure EditMaxAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

private

{ Private declarations }

IsFull: Boolean;

public

{ Public declarations }

Procedure CheckEdit(Var IsCorrect, IsFull: Boolean);

end;

Const

RussianLetters =

'абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ';

Var

FormInput: TForm4;

implementation

{$R \*.dfm}

procedure TForm4.BtnOkClick(Sender: TObject);

Var

IsCorrect: Boolean;

Begin

CheckEdit(IsCorrect, IsFull);

If IsFull And IsCorrect then

ModalResult := mrOk

Else If IsFull Then

Begin

Application.MessageBox('Введите корректные данные!', 'Ошибка!',

MB\_ICONERROR);

End

Else

Application.MessageBox('Заполните все поля!!', 'Ошибка!', MB\_ICONERROR);

End;

procedure TForm4.ButtonCancelClick(Sender: TObject);

begin

close

end;

Procedure TForm4.CheckEdit(Var IsCorrect, IsFull: Boolean);

Begin

If (ShapeName.Pen.Color = clGreen) And (ShapeSort.Pen.Color = clGreen) And

(ShapeNumber.Pen.Color = clGreen) and (ShapeRuble.Pen.Color = clGreen) And

(ShapePenny.Pen.Color = clGreen) And (ShapeMinAge.Pen.Color = clGreen) and

(ShapeMaxAge.Pen.Color = clGreen) Then

IsCorrect := True

Else

IsCorrect := False;

If (EditName.Text = '') Or (EditSort.Text = '') or (EditNumber.Text = '') Or

(EditRuble.Text = '') Or (EditPenny.Text = '') Or (EditMaxAge.Text = '')

or (EditMinAge.Text = '') Then

IsFull := False

Else

IsFull := True;

if (Length(EditMinAge.Text) <> 0) and (Length(EditMaxAge.Text) <> 0) then

begin

if StrToInt(EditMinAge.Text) > StrToInt(EditMaxAge.Text) then

begin

ShapeMinAge.Pen.Color := clRed;

ShapeMaxAge.Pen.Color := clRed;

IsFull := False;

end;

End;

End;

procedure TForm4.EditMaxAgeChange(Sender: TObject);

Begin

If Length(EditMaxAge.Text) = 0 Then

Begin

IsFull := False;

ShapeMaxAge.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeMaxAge.Pen.Color := clGreen;

End

end;

Procedure TForm4.CheckEnter(Sender: TObject; var Key: Char);

var

chars: set of Char;

editText: String;

begin

editText := (Sender as TEdit).Text;

if (Length(editText) = 0) then

begin

chars := ['0' .. '9', #8, #31];

end

else

chars := ['0' .. '9', #8, #31];

if not(Key in chars) then

Key := #0;

end;

Function CheckZeroStr(Str: String): Boolean;

Var

I, Len: Integer;

Fl: Boolean;

Begin

Fl := False;

Len := Length(Str) - 1;

For I := 1 to Len do

Begin

If (Str[I] = '0') and (I <> High(Str)) then

Begin

If (Str[I + 1] <> '') then

Fl := True;

End;

End;

CheckZeroStr := Fl;

End;

procedure TForm4.EditMaxAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

procedure TForm4.EditMaxAgeKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

Begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditMaxAge.Text)) Then

EditMaxAge.Text := '';

If TryStrToInt(EditMaxAge.Text + Key, Value) Then

If (Value > MAX\_AGE) Then

Key := #0;

if EditMaxAge.Text = '0' then

Begin

EditMaxAge.Text := '';

End;

end;

Procedure TForm4.EditMinAgeChange(Sender: TObject);

Begin

If Length(EditMinAge.Text) = 0 Then

Begin

IsFull := False;

ShapeMinAge.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeMinAge.Pen.Color := clGreen;

End;

end;

procedure TForm4.EditMinAgeKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

procedure TForm4.EditMinAgeKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

Begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditMinAge.Text)) Then

EditMinAge.Text := '';

If TryStrToInt(EditMinAge.Text + Key, Value) Then

If (Value > MAX\_AGE) Then

Key := #0;

if EditMinAge.Text = '0' then

Begin

EditMinAge.Text := '';

End;

end;

Procedure TForm4.EditNameChange(Sender: TObject);

Begin

If (Length(EditName.Text) = 0) then

Begin

IsFull := False;

ShapeName.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeName.Pen.Color := clGreen;

End;

End;

Procedure TForm4.CheckShiftInsert(Sender: TObject; var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_INSERT) and (SsShift in Shift) then

Key := 0

End;

procedure TForm4.EditNameKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

Procedure TForm4.EditNameKeyPress(Sender: TObject; var Key: Char);

Begin

If Key = #13 Then

BtnOk.Click();

If (Key = '-') or (Key = #190) or (Key = #191) Then

Key := #0;

If Not((Pos(Key, RussianLetters) <> 0) or CharInSet(Key,

[#08, #45, #32, #46])) Then

Key := #0;

End;

Procedure TForm4.EditNumberChange(Sender: TObject);

Begin

If (Length(EditNumber.Text) = 0) Then

Begin

IsFull := False;

ShapeNumber.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeNumber.Pen.Color := clGreen;

End;

End;

procedure TForm4.EditNumberKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

Procedure TForm4.EditNumberKeyPress(Sender: TObject; var Key: Char);

Var

chars: set of Char;

editText: String;

begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditNumber.Text)) Then

EditNumber.Text := '';

if EditNumber.Text = '0' then

Begin

EditNumber.Text := '';

End;

End;

Procedure TForm4.EditPennyChange(Sender: TObject);

Begin

If Length(EditPenny.Text) = 0 Then

Begin

IsFull := False;

ShapePenny.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapePenny.Pen.Color := clGreen;

End

End;

procedure TForm4.EditPennyKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

Procedure TForm4.EditPennyKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

Begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditPenny.Text)) Then

EditPenny.Text := '';

If TryStrToInt(EditPenny.Text + Key, Value) Then

If (Value > MAX\_PENNY) Then

Key := #0;

End;

Procedure TForm4.EditRubleChange(Sender: TObject);

Begin

If Length(EditRuble.Text) = 0 Then

Begin

IsFull := False;

ShapeRuble.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeRuble.Pen.Color := clGreen;

End

End;

procedure TForm4.EditRubleKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

Procedure TForm4.EditRubleKeyPress(Sender: TObject; var Key: Char);

Var

Value: Integer;

Begin

CheckEnter(Sender, Key);

If Key = #8 then

If (CheckZeroStr(EditRuble.Text)) Then

EditRuble.Text := '';

If TryStrToInt(EditRuble.Text + Key, Value) Then

If (Value > MAX\_RUBLE) Then

Key := #0;

if EditRuble.Text = '0' then

Begin

EditRuble.Text := '';

End;

End;

Procedure TForm4.EditSortChange(Sender: TObject);

Begin

If (Length(EditSort.Text) = 0) Then

Begin

IsFull := False;

ShapeSort.Pen.Color := clRed;

End

Else

Begin

IsFull := True;

ShapeSort.Pen.Color := clGreen;

End;

End;

procedure TForm4.EditSortKeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

begin

CheckShiftInsert(Sender, Key, Shift);

end;

Procedure TForm4.EditSortKeyPress(Sender: TObject; var Key: Char);

Begin

If Key = #13 Then

BtnOk.Click();

If (Key = '-') or (Key = #190) Then

Key := #0;

If Not((Pos(Key, RussianLetters) <> 0) or CharInSet(Key,

[#08, #45, #32, #46])) Then

Key := #0;

End;

end.

**unit Unit6Laba41;**

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Samples.Spin,

Vcl.Grids;

Type

TForm6 = class(TForm)

ShopForChildren: TStringGrid;

BtnClose: TButton;

LabelYourAgeGame: TLabel;

procedure BtnCloseClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form6: TForm6;

implementation

{$R \*.dfm}

uses Unit1Laba41, Unit4Laba41, Unit5Laba41;

procedure TForm6.BtnCloseClick(Sender: TObject);

begin

Close

end;

function CostToStr(Cost: TCost): String;

begin

CostToStr := IntToStr(Cost.Ruble) + 'р. ' + IntToStr(Cost.Penny) + 'коп.';

end;

procedure TForm6.FormCreate(Sender: TObject);

Var

I: Integer;

begin

ShopForChildren.Cells[0, 0] := 'Название';

ShopForChildren.Cells[1, 0] := 'Мин. возраст';

ShopForChildren.Cells[2, 0] := 'Макс. возраст';

ShopForChildren.Cells[3, 0] := 'Цена (руб.)';

for I := 0 to 3 do

ShopForChildren.ColWidths[I] := 80;

end;

end.

**unit Unit5Laba41;**

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

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

Vcl.Samples.Spin;

type

TForm5 = class(TForm)

LabelNameForm13: TLabel;

ShopInfo: TStringGrid;

BtnClose: TButton;

LabelSeparate: TLabel;

LabelMaxCostGame: TLabel;

LabelNameMaxCostGame: TLabel;

LabelMaxCostGameCost: TLabel;

procedure FormShow(Sender: TObject);

procedure BtnCloseClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

implementation

{$R \*.dfm}

procedure TForm5.BtnCloseClick(Sender: TObject);

begin

close

end;

procedure TForm5.FormShow(Sender: TObject);

Var

I: Integer;

begin

ShopInfo.Cells[0, 0] := 'Название';

ShopInfo.ColWidths[0] := 103;

end;

end.

**Код программы Java:**

import java.io.\*;

import java.util.Scanner;

class gameShop {

private Game[] games;

gameShop() {

games = new Game[0];

}

public void addGame(String name, String sort, String number, Cost cost, String minAge, String maxAge) {

games = changeCountOfGames(games, games.length + 1);

games[games.length - 1] = new Game(name, sort, number, cost, minAge, maxAge);

}

public void deleteProduct(int index){

if (index != games.length - 1){

for (int i = index; i < games.length - 1; i++){

games[i] = games[i + 1];

}

}

games = changeCountOfGames(games, games.length - 1);

}

public void changeGame(int index, Game newGame){

games[index] = newGame;

}

public Game[] sortGames(){

Game[] sortedGames = new Game[0];

for (Game product : games) {

if ((Integer.parseInt(product.getMinAge()) = 1) && (Integer.parseInt(product.getMinAge()) < 1) && (Integer.parseInt(product.getMaxAge()) < 3) && (Integer.parseInt(product.getMaxAge()) = 3))) {

sortedGames = changeCountOfGames(sortedGames, sortedGames.length + 1);

sortedGames[sortedGames.length - 1] = product;

}

}

for (int i = 0; i < sortedGames.length - 1; i++) {

for (int j = 0; j < sortedGames.length - i - 1; j++) {

if (Integer.parseInt(sortedGames[j].getMinAge()) > Integer.parseInt(sortedGames[j + 1].getMinAge()) ||

(Integer.parseInt(sortedGames[j].getMinAge()) == Integer.parseInt(sortedGames[j + 1].getMinAge()) &&

Integer.parseInt(sortedGames[j].getMaxAge()) > Integer.parseInt(sortedGames[j + 1].getMaxAge()))) {

Game temp = sortedGames[j];

sortedGames[j] = sortedGames[j + 1];

sortedGames[j + 1] = temp;

}

}

}

return sortedGames;

}

public Game[] sortGames(int minAgeRange, int maxAgeRange, int rubleRange) {

Game[] sortedGames = new Game[0];

for (Game game : games) {

int minAge = Integer.parseInt(game.getMinAge());

int maxAge = Integer.parseInt(game.getMaxAge());

int ruble =

Integer.parseInt(String.valueOf(game.getCost().getRuble()));

int penny =

Integer.parseInt(String.valueOf(game.getCost().getPenny()));

if (minAge >= minAgeRange && maxAge <= maxAgeRange && rubleRange\*100

>= (ruble\*100 + penny)) {

sortedGames = changeCountOfGames(sortedGames, sortedGames.length

+ 1);

sortedGames[sortedGames.length - 1] = game;

}

}

for (int i = 0; i < sortedGames.length - 1; i++) {

for (int j = 0; j < sortedGames.length - i - 1; j++) {

int minAge1 = Integer.parseInt(sortedGames[j].getMinAge());

int maxAge1 = Integer.parseInt(sortedGames[j].getMaxAge());

int minAge2 = Integer.parseInt(sortedGames[j + 1].getMinAge());

int maxAge2 = Integer.parseInt(sortedGames[j + 1].getMaxAge());

int ruble1 = Integer.parseInt(String.valueOf(sortedGames[j].

getCost().getRuble()));

int ruble2 = Integer.parseInt(String.valueOf(sortedGames[j +

1].getCost().getRuble()));

if (minAge1 > minAge2 || (minAge1 == minAge2 && maxAge1 >

maxAge2) || (minAge1 == minAge2 && maxAge1 == maxAge2 &&

ruble1 > ruble2)) {

Game temp = sortedGames[j];

sortedGames[j] = sortedGames[j + 1];

sortedGames[j + 1] = temp;

}

}

}

return sortedGames;

} public Game[] sortGames1() {

Game maxPriceGame = null;

for (Game game : games) {

if (maxPriceGame == null || game.getCost().getRuble() >

maxPriceGame.getCost().getRuble() ||

(game.getCost().getRuble() == maxPriceGame.getCost().getRuble()

&& game.getCost().getPenny() >

maxPriceGame.getCost().getPenny())) {

maxPriceGame = game;

}

}

if (maxPriceGame != null) {

System.out.println("Игрушка с максимальной ценой:

maxPriceGame.getName());

} else {

System.out.println("Нет доступных игрушек.");

}

return new Game[] { maxPriceGame };

}

public void showAllGames(){

int i = 0;

System.out.println("№ - Наименование - Назначение - Инвентарный номер -

Цена - Дата получения");

for(Game product : games){

System.out.println("[" + (++i) + "]" + product.toString());

}

}

public void setProducts(Game[] games){

this.games = games;

}

public Game[] getGames(){

return games;

}

Game[] changeCountOfGames(Game[] games, int newLength){

int size = games.length;

Game[] newGames = new Game[newLength];

System.arraycopy(games, 0, newGames, 0, Math.min(newLength, size));

return newGames;

}

}

class Game{

private String name;

private String sort;

private Cost cost;

private String minAge;

private String maxAge;

private String number;

Game(String name, String sort, String number, Cost cost, String minAge, String maxAge){

this.name = name;

this.sort = sort;

this.number = number;

this.cost = cost;

this.minAge = minAge;

this.maxAge = maxAge;

}

public void setName(String name){

this.name = name;

}

public String getName(){

return this.name;

}

public void setSort(String sort){

this.sort = sort;

}

public String getSort(){

return sort;

}

public void setNumber(String number){

this.number = number;

}

public String getNumber(){

return number;

}

public void setCost(Cost cost){

this.cost = cost;

}

public Cost getCost(){

return cost;

}

public void setMinAge(String minAge){

this.minAge = minAge;

}

public String getMinAge(){

return minAge;

}

public void setMaxAge(String maxAge){

this.maxAge = maxAge;

}

public String getMaxAge(){

return maxAge;

}

public String toString(){

return "[" + name + "]" + "[" + sort + "]" + "[" + number + "]" + "[" + cost.toString() + "]" + "[" + minAge + "]" + "[" + maxAge + "]";

}

}

class Cost{

private int ruble;

private int penny;

Cost(int ruble, int penny){

this.ruble = ruble;

this.penny = penny;

}

public void setRuble(int ruble){

this.ruble = ruble;

}

public int getRuble(){

return ruble;

}

public void setPenny(int penny){

this.penny = penny;

}

public int getPenny(){

return penny;

}

public String toString(){

return ruble + "р. " + penny + "коп.";

}

public static boolean isOkay(int ruble, int penny){

final int MIN = 0;

final int MAX\_RUBLE = 9999;

final int MAX\_PENNY = 99;

return ruble = MIN && ruble > MIN && ruble = MAX\_RUBLE && ruble < MAX\_RUBLE && penny = MIN && penny > MIN && penny = MAX\_PENNY && penny < MAX\_PENNY && (ruble != MIN || penny != MIN);

}

}

public class Main {

static Scanner scanner = new Scanner(System.in);

static gameShop myShop = new gameShop();

public static void waiting() {

System.out.println("Для выхода в меню нажмите Enter...");

scanner.nextLine();

}

public static String inputFilePath() {

String filePath;

boolean isIncorrect;

do {

isIncorrect = false;

System.out.println("Введите путь к файлу.");

filePath = scanner.nextLine();

File file = new File(filePath);

if (!file.exists() ||!filePath.endsWith(".txt") || file.isDirectory()) {

isIncorrect = true;

System.out.println("Файл не найден");

}

if (!file.canWrite() ||!file.canRead()){

isIncorrect = true;

System.out.println("Проверьте параметры доступа файла!");

}

} while (isIncorrect);

return filePath;

}

public static Game[] takeGamesFromFile(String filePath) {

Game[] games = new Game[0];

String currentString;

boolean isIncorrect;

isIncorrect = false;

try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {

while ((reader.read() != -1) && !(isIncorrect)) {

currentString = reader.readLine();

String[] words;

words = currentString.split(" ", 6);

String name;

name = words[0];

String sort = words[1];

String number = words[2];

int ruble;

int penny;

try {

String[] rubleAndPenny = words[3].split("[.]", 2);

ruble = Integer.parseInt(rubleAndPenny[0].trim());

penny = Integer.parseInt(rubleAndPenny[1].trim());

} catch (NumberFormatException | ArrayIndexOutOfBoundsException e) {

System.out.println("Ошибка при считывании цены: " +

currentString);

continue;

}

int minAge;

int maxAge;

try {

minAge = Integer.parseInt(words[4]);

maxAge = Integer.parseInt(words[5]);

} catch (NumberFormatException e) {

System.out.println("Ошибка при считывании возраста: " +

currentString);

continue;

}

if (name.length() > 20 || sort.length() > 15 || name.isEmpty() || number.length() > 5 ||

!(number.matches("(?i).\*[0-9].\*")) || sort.isEmpty() || !Cost.isOkay(ruble, penny) || minAge > 99 || maxAge > 99) {

isIncorrect = true;

System.out.println("В файле содержатся некорректные данные...");

} else {

games = myShop.changeCountOfGames(games, games.length + 1);

games[games.length - 1] = new Game(name, sort, number, new Cost(ruble, penny), String.valueOf(minAge), String.valueOf(maxAge));

}

}

if (!isIncorrect) {

System.out.println("Информация была успешно загружена из файла.");

}

} catch (Exception e) {

System.out.println("Произошла ошибка чтения данных из файла... Убедитесь,

что все данные записаны корректно!");

}

return games;

}

public static void saveToFile(Game[] games, String filePath) {

try (PrintWriter fileOut = new PrintWriter(filePath)) {

for (Game product : games)

fileOut.println(" " + product.getName() + " " + product.getSort() + " " + product.getNumber() + " " +

product.getCost().getRuble() + "." + product.getCost().getPenny() + " " + product.getMinAge() + " " + product.getMaxAge());

} catch (Exception e) {

System.out.println("Произошла ошибка при записи в файл...");

}

System.out.println("Данные успешно сохранены в файл!");

}

public static int inputIndex(final int MAX\_INDEX){

final int MIN\_INDEX;

MIN\_INDEX = 1;

int index;

index = 0;

boolean isIncorrect;

System.out.println("Введите номер игры");

do {

isIncorrect = false;

try {

index = Integer.parseInt(scanner.nextLine());

} catch (Exception e){

System.out.println("Проверьте корректность введенных данных!");

isIncorrect = true;

}

if (!isIncorrect && (index < MIN\_INDEX || index > MAX\_INDEX)){

System.out.println("Был указан неверный номер игры!");

isIncorrect = true;

}

} while (isIncorrect);

return index;

}

public static String inputName() {

final int MAX\_SIZE;

MAX\_SIZE = 20;

String name;

boolean isIncorrect;

System.out.println("Введите название. Длина не должна превышать 20 букв.");

do {

isIncorrect = false;

name = scanner.nextLine();

if (name.length() > MAX\_SIZE) {

isIncorrect = true;

System.out.println("Вы превысили допустимую длину(не больше 20)!");

}

if (name.isEmpty()|| !(name.matches("(?i).\*[a-zA-Zа-яА-Я].\*"))) {

isIncorrect = true;

System.out.println("Введите название!");

}

} while (isIncorrect);

return name;

}

public static String inputSort() {

final int MAX\_SIZE;

MAX\_SIZE = 15;

String sort;

boolean isIncorrect;

System.out.println("Введите назначение прибора. Длина не должна превышать 15

букв.");

do {

isIncorrect = false;

sort = scanner.nextLine();

if (sort.length() > MAX\_SIZE) {

isIncorrect = true;

System.out.println("Вы превысили допустимую длину(не больше 15)!");

}

if (sort.isEmpty() || !(sort.matches("(?i).\*[a-zA-Zа-яА-Я].\*"))) {

isIncorrect = true;

System.out.println("Введите назначение!");

}

} while (isIncorrect);

return sort;

}

public static String inputInventory(){

boolean isIncorrect;

String number;

System.out.println("Введите кол-во товаров");

do{

isIncorrect = false;

number = scanner.nextLine();

if( number.isEmpty() || !(number.matches("(?i).\*[0-9].\*")) || number.length() > 4) {

isIncorrect = true;

System.out.println("Введите правильное кол-во товаров!");

}

} while (isIncorrect);

return number;

}

public static Cost inputCost(){

boolean isIncorrect;

int ruble;

ruble = 0;

int penny;

penny = 0;

String cost;

String[] rubleAndPenny;

System.out.println("Введите цену товара. Формат ввода - RUBLE.PENNY, где

RUBLE(кол-во рублей) - число от 0 до 9999, PENNY(кол-во

копеек) - число от 0 до 99");

do {

isIncorrect = false;

cost = scanner.nextLine();

if (cost.isEmpty()) {

isIncorrect = true;

System.out.println("Введите цену!");

}

if (!isIncorrect) {

try {

rubleAndPenny = cost.split("[.]", 2);

ruble = Integer.parseInt(rubleAndPenny[0].trim());

penny = Integer.parseInt(rubleAndPenny[1].trim());

if (!Cost.isOkay(ruble, penny)){

System.out.println("Вы ввели некорретное значение цены!");

isIncorrect = true;

}

} catch (Exception e) {

System.out.println("Вы ввели некорректные данные!");

isIncorrect = true;

}

}

} while (isIncorrect);

return new Cost(ruble, penny);

}

public static String inputMinAge(){

boolean isIncorrect;

String minAge;

System.out.println("Введите минимальный возраст");

do{

isIncorrect = false;

minAge = scanner.nextLine();

if( minAge.isEmpty() || !(minAge.matches("(?i).\*[0-9].\*")) || minAge.length() > 4) {

isIncorrect = true;

System.out.println("Введите корректный минимальный возраст!");

}

} while (isIncorrect);

return minAge;

}

public static String inputMaxAge(){

boolean isIncorrect;

String maxAge;

System.out.println("Введите максимальный возраст");

do{

isIncorrect = false;

maxAge = scanner.nextLine();

if( maxAge.isEmpty() || !(maxAge.matches("(?i).\*[0-9].\*")) || maxAge.length() > 4) {

isIncorrect = true;

System.out.println("Введите корректный максимальный возраст!");

}

} while (isIncorrect);

return maxAge;

}

public static int inputChoice(int kind, final int MAX\_NUM) {

final int MIN\_NUM;

MIN\_NUM = 1;

int choice;

choice = 0;

boolean isIncorrect;

if(kind == 1){

System.out.println("Меню:\n" + "1) Добавить игру\n" +

"2) Изменить информацию о игре\n" +

"3) Удалить игру\n" + "4) Показать все игры\n" +

"5) Показать все игры, для детей от 1 до 3 лет\n" +

"6) Показать самую дорогую игру из списка\n" +

"7) Показать все игры, подходящие для детей для вашего диапазона

лет\n" +

"8) Загрузить информацию о играх\n" + "9) Сохранить информацию о

играх\n" +

"10) Помощь\n" + "11) Выход из программы\n" + "Выберите одно из

действий");

} else {

System.out.println("Меню:\n" + "1) Изменить наименование\n" +

"2) Изменить назначение\n" +

"3) Изменить инвентарный номер\n" +

"4) Изменить цену\n" +

"5) Изменить минимальный возраст\n" +

"6) Изменить максимальный возраст\n" +

"7) Выйти в главное меню\n" + "Выберите

одно из действий");

}

do {

isIncorrect = false;

try {

choice = Integer.parseInt(scanner.nextLine());

} catch (Exception e){

System.out.println("Проверьте корректность введенных данных!");

isIncorrect = true;

}

if (!isIncorrect && (choice < MIN\_NUM || choice > MAX\_NUM)){

System.out.println("Выберите пукнт из меню!");

isIncorrect = true;

}

} while (isIncorrect);

return choice;

}

public static void printHelp() {

System.out.println("Помощь:\n" + "1) Максимальная длина названия игры - 20\n"

+ "2) Максимальная длина назначения товара -

15\n"

+ "3) Максимальная номера - 4 \n"

+ "4) Цена игры указывается в формате

RUBLE.PENNY(через точку!), где RUBLE -

кол-во рублей(0 - 9999) и PENNY - кол-во

копеек(0 - 99). Цена не может быть

пустой!(0.0)\n"

+ "5) При загрузке информации о игре из файла

убедитесь, что в файле каждый прибор

описан с новой строки и соответствует

примеру:\n" +

"NAME SORT RUBLE.PENNY MINAGE MAXAGE,

где NAME - наименование прибора, SORT –

назначение прибора, RUBLE.PENNY - цена

прибора, MINAGE - миниммальный возраст,

MAXAGE - максимальный возраст");

}

public static Game changeMenu(Game newGame, int choice){

switch (choice) {

case 1 -> {

newGame.setName(inputName());

System.out.println("Информация была успешно изменена");

}

case 2 -> {

newGame.setSort(inputSort());

System.out.println("Информация была успешно изменена");

}

case 3 -> {

newGame.setNumber(inputInventory());

System.out.println("Информация была успешно изменена");

}

case 4 -> {

newGame.setCost(inputCost());

System.out.println("Информация была успешно изменена");

}

case 5 -> {

newGame.setMinAge(inputMinAge());

System.out.println("Информация была успешно изменена");

}

case 6 -> {

newGame.setMaxAge(inputMaxAge());

System.out.println("Информация была успешно изменена");

}

}

return newGame;

}

public static void menu(int choice) {

switch (choice) {

case 1 -> {

myShop.addGame(inputName(), inputSort(), inputInventory(), inputCost(), inputMinAge(), inputMaxAge());

System.out.println("Игра была успешно добавлена.");

waiting();

}

case 2 -> {

int index;

if (myShop.getGames().length != 0) {

System.out.println("Ваши игры:");

myShop.showAllGames();

index = inputIndex(myShop.getGames().length) - 1;

Game newGame = myShop.getGames()[index];

int choiceChangeMenu;

do {

System.out.println("Выбранная игра:\n" + newGame.toString());

choiceChangeMenu = inputChoice(2, 7);

newGame = changeMenu(newGame, choiceChangeMenu);

} while (choiceChangeMenu != 7);

myShop.changeGame(index, newGame);

} else {

System.out.println("Игр пока нет...");

waiting();

}

}

case 3 -> {

int size;

size = myShop.getGames().length;

if (size != 0) {

System.out.println("Ваши игры:");

myShop.showAllGames();

myShop.deleteProduct(inputIndex(size) - 1);

System.out.println("Игра была успешно удалена");

} else {

System.out.println("Игр пока нет...");

}

waiting();

}

case 4 -> {

if (myShop.getGames().length != 0) {

System.out.println("Ваши игры:");

myShop.showAllGames();

} else {

System.out.println("Игр пока нет...");

}

waiting();

}

case 5 -> {

if (myShop.getGames().length != 0) {

int i;

i = 0;

System.out.println("Отсортированные игры:\n" + "№ - Наименование

- Назначение - Инвентарный номер - Цена –

Минимальный возраст - Максимальный возраст");

for (Game sortedProduct : myShop.sortGames()) {

System.out.println("[" + (++i) + "]" + sortedProduct.toString());

}

} else {

System.out.println("Игрушек пока нет...");

}

waiting();

}

case 6-> {

if (myShop.getGames().length != 0) {

Game[] sortedGames = myShop.sortGames1();

if (sortedGames.length > 0 && sortedGames[0] != null) {

} else {

System.out.println("Нет доступных игрушек.");

}

} else {

System.out.println("Игрушек пока нет...");

}

waiting();

}

case 7-> {

if (myShop.getGames().length != 0) {

Scanner scanner = new Scanner(System.in);

System.out.print("Введите минимальный возраст: ");

int minAgeRange = scanner.nextInt();

System.out.print("Введите максимальный возраст: ");

int maxAgeRange = scanner.nextInt();

System.out.print("Введите необходимую вам цену (руб.): ");

int rubleRange = scanner.nextInt();

int i = 0;

System.out.println("Отсортированные игры:");

Game[] sortedGames = myShop.sortGames(minAgeRange, maxAgeRange,

rubleRange);

if (sortedGames.length != 0) {

for (Game sortedGame : sortedGames) {

System.out.println("[" + (++i) + "] " +

sortedGame.toString());

}

} else {

System.out.println("Нет игр в указанном возрастном

диапазоне.");

}

} else {

System.out.println("Игрушек пока нет...");

}

waiting();

}

case 8 -> {

myShop.setProducts(takeGamesFromFile(inputFilePath()));

waiting();

}

case 9 -> {

if (myShop.getGames().length != 0) {

saveToFile(myShop.getGames(), inputFilePath());

} else {

System.out.println("Игрушек пока нет...");

}

waiting();

}

case 10 -> {

printHelp();

waiting();

}

case 11 -> System.out.println("Вы завершили работу с программой.

Приятного дня!");

}

}

public static void main(String[] args) {

int choice;

System.out.println("Данная программа использует данные магазина игрушек и

выводит игрушки детям от 1-3 лет. \nПеред тем, как

работать с программой, просмотрите пункт 8) Помощь");

do {

choice = inputChoice(1, 11);

menu(choice);

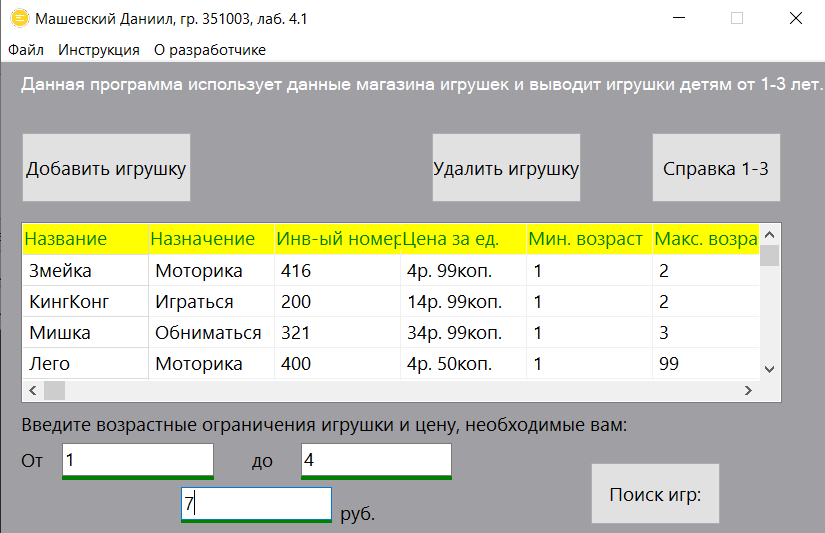
} while(choice != 11);

}

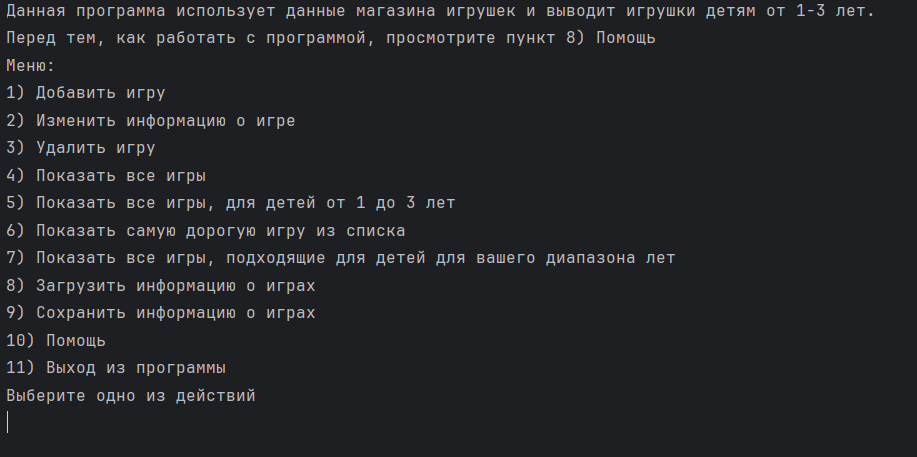
}

**Скриншоты:**

**Delphi:**



**Java:**



**Блок-схема:**

