УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №4.1

по предмету «Основы алгоритмизации и программирования»

Вариант 20

Выполнил:

Захвей И.В.

Гр. 351005

Проверил:

Данилова Г. В.

Минск 2024

**Задание:**

Сведения об ассортименте игрушек в магазине:

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

В программе должны быть реализованы следующие функции:

* ввод записей;
* корректировка записей;
* удаление записей;
* просмотр записей на экране
* отбор информации по заданному критерию.

**Код программы Delphi:**

**MainUnit.pas**

Type

ERRORS\_CODE = (SUCCESS, INCORRECT\_DATA\_FILE, A\_LOT\_OF\_DATA\_FILE,

OUT\_OF\_BORDER, OUT\_OF\_BORDER\_SIZE);

TGridCracker = Class(TStringGrid);

IntArr = Array Of Integer;

Matrix = Array Of IntArr;

Const

ERRORS: Array [ERRORS\_CODE] Of String = ('Successfull',

'Данные в файле не корректные',

'В файле неверное количество элементов или стоит

лишний пробел',

'Числа должны быть в диапазоне [-70, 70]',

'Размер должен быть в диапазоне [1, 5]');

Var

ManeForm: TManeForm;

IsSaved: Boolean = True;

Implementation

{$R \*.dfm}

Procedure TManeForm.FormCloseQuery(Sender: TObject; Var CanClose: Boolean);

Begin

If Not IsSaved Then

Case Application.MessageBox('Сохранить данные перед выходом?', 'Выход',

MB\_YESNOCANCEL + MB\_ICONQUESTION + MB\_DEFBUTTON3) Of

IDYES:

Begin

SaveButtonMenu.Click;

CanClose := True;

End;

IDNO:

CanClose := True;

IDCANCEL:

CanClose := False;

End

Else

Case Application.MessageBox('Вы точно хотите выйти?', 'Выход',

MB\_YESNO + MB\_ICONQUESTION + MB\_DEFBUTTON2) Of

IDYES:

CanClose := True;

IDNO:

CanClose := False;

End;

If CanClose = True Then

DeleteFile(СORRECTION\_FILE\_PATH);

End;

Procedure TManeForm.FormCreate(Sender: TObject);

Var

StorageFile, CorrectionFile: TToyFile;

Begin

OpenFile(STORAGE\_FILE\_PATH, StorageFile, FmReset);

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmRewrite);

CloneFileToAnother(StorageFile, CorrectionFile);

CloseFile(StorageFile);

CloseFile(CorrectionFile);

// Draw FixedRow information

RecordsGrid.Cells[0, 0] := '№';

RecordsGrid.Cells[1, 0] := 'Название';

RecordsGrid.Cells[2, 0] := 'Цена(BYN)';

RecordsGrid.Cells[3, 0] := 'Количество';

RecordsGrid.Cells[4, 0] := 'Возраст';

DrawRecordOnGrid(RecordsGrid, СORRECTION\_FILE\_PATH);

End;

Procedure TManeForm.SaveButtonMenuClick(Sender: TObject);

Var

StorageFile, CorrectionFile: TToyFile;

Begin

OpenFile(STORAGE\_FILE\_PATH, StorageFile, FmRewrite);

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmReset);

CloneFileToAnother(CorrectionFile, StorageFile);

IsSaved := True;

CloseFile(StorageFile);

CloseFile(CorrectionFile);

End;

Procedure TManeForm.SearchRecButtonClick(Sender: TObject);

Begin

FindRecForm := TFindRecForm.Create(Self);

FindRecForm.FormCreate(Self);

FindRecForm.ShowModal();

FindRecForm.Free();

End;

Procedure TManeForm.ManualButtonMenuClick(Sender: TObject);

Begin

ManualForm := TManualForm.Create(Self);

ManualForm.ShowModal;

ManualForm.Free;

End;

Procedure TManeForm.RecordsGridDblClick(Sender: TObject);

Begin

If ChangeRecButton.Enabled Then

ChangeRecButton.Click();

End;

Procedure TManeForm.RecordsGridKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_DELETE) And (DelRecButton.Enabled) Then

DelRecButton.Click

End;

Procedure TManeForm.RecordsGridSelectCell(Sender: TObject; ACol, ARow: Integer;

Var CanSelect: Boolean);

Begin

If ARow = 0 Then

Begin

DelRecButton.Enabled := False;

ChangeRecButton.Enabled := False;

End

Else

Begin

(Sender As TStringGrid).FixedRows := 1;

ChangeRecButton.Enabled := True;

DelRecButton.Enabled := True;

End;

End;

Procedure TManeForm.AddRecButtonClick(Sender: TObject);

Var

Res: TModalResult;

Begin

AddRecForm := TAddRecForm.Create(Self);

Res := AddRecForm.ShowModal();

If Res = MrOk Then

IsSaved := False;

AddRecForm.Free();

DrawRecordOnGrid(RecordsGrid, СORRECTION\_FILE\_PATH);

End;

Procedure TManeForm.ChangeRecButtonClick(Sender: TObject);

Var

RecIndex: Integer;

Res: TModalResult;

Begin

RecIndex := RecordsGrid.Row - 1;

ChangeRecForm := TChangeRecForm.Create(Self);

ChangeRecForm.FormCreate(RecIndex, Self);

Res := ChangeRecForm.ShowModal();

If Res = MrOk Then

IsSaved := False;

ChangeRecForm.Free();

DrawRecordOnGrid(RecordsGrid, СORRECTION\_FILE\_PATH);

End;

Procedure TManeForm.DelRecButtonClick(Sender: TObject);

Var

CurRecIndex: Integer;

Choice: Integer;

Begin

CurRecIndex := RecordsGrid.Row - 1;

Choice := Application.MessageBox('Вы точно хотите удалить запись?',

'Внимание!', MB\_YESNO + MB\_ICONQUESTION + MB\_DEFBUTTON2);

If (Choice = IDYES) Then

Begin

DeleteRec(CurRecIndex);

DrawRecordOnGrid(RecordsGrid, СORRECTION\_FILE\_PATH);

IsSaved := False;

End;

End;

Procedure TManeForm.DeveloperButtonMenuClick(Sender: TObject);

Begin

DeveloperForm := TDeveloperForm.Create(Self);

DeveloperForm.ShowModal;

DeveloperForm.Free;

End;

Procedure TManeForm.ExitButtonMenuClick(Sender: TObject);

Begin

Close();

End;

End.

**AddRecUnit.pas**

Var

AddRecForm: TAddRecForm;

Implementation

{$R \*.dfm}

Procedure TAddRecForm.AddRecButtonClick(Sender: TObject);

Var

Toy: RToy;

Begin

Toy.Name := RecNameEdit.Text;

Toy.Cost := StrToInt(RecCostEdit.Text);

Toy.Count := StrToInt(RecCountEdit.Text);

Toy.MinAge := StrToInt(MinAgeEdit.Text);

Toy.MaxAge := StrToInt(MaxAgeEdit.Text);

AddRecord(Toy);

End;

Function TAddRecForm.IsAllFieldCorrect: Boolean;

Var

LBorder, RBorder: Integer;

IsAllFilled, IsCorrect: Boolean;

Begin

IsAllFilled := (RecNameEdit.Text <> '') And (RecCostEdit.Text <> '')

And (RecCountEdit.Text <> '') And (MinAgeEdit.Text <> '')

And (MaxAgeEdit.Text <> '');

If IsAllFilled Then

Begin

LBorder := StrToInt(MinAgeEdit.Text);

RBorder := StrToInt(MAxAgeEdit.Text);

IsCorrect := IsAllFilled And (LBorder < RBorder)

End

Else

IsCorrect := False;

IsAllFieldCorrect := IsCorrect;

End;

Procedure TAddRecForm.MaxAgeEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And AddRecButton.Enabled Then

AddRecButton.Click();

If (Key = VK\_DOWN) And AddRecButton.Enabled Then

AddRecButton.SetFocus();

If (Key = VK\_UP) Or (Key = VK\_LEFT) Then

MinAgeEdit.SetFocus();

End;

Procedure TAddRecForm.AgeEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With Sender As TEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_AGE, MAX\_AGE, Text);

End;

Procedure TAddRecForm.MinAgeEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift))

And Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And AddRecButton.Enabled Then

AddRecButton.Click();

If (Key = VK\_DOWN) Or (Key = VK\_RIGHT) Or (Key = VK\_RETURN) Then

MaxAgeEdit.SetFocus();

If (Key = VK\_UP) Then

RecCountEdit.SetFocus();

End;

Procedure TAddRecForm.AnyEditChange(Sender: TObject);

Begin

AddRecButton.Enabled := IsAllFieldCorrect();

End;

Procedure TAddRecForm.RecCostEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And AddRecButton.Enabled Then

AddRecButton.Click();

If (Key = VK\_RETURN) Or (Key = VK\_DOWN) Then

RecCountEdit.SetFocus();

If (Key = VK\_UP) Then

RecNameEdit.SetFocus();

End;

Procedure TAddRecForm.RecCostEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With RecCostEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_COST, MAX\_COST, Text);

End;

Procedure TAddRecForm.RecCountEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And AddRecButton.Enabled Then

AddRecButton.Click();

If (Key = VK\_RETURN) Or (Key = VK\_DOWN) Then

MinAgeEdit.SetFocus();

If (Key = VK\_UP) Then

RecCostEdit.SetFocus();

End;

Procedure TAddRecForm.RecCountEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With RecCountEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_COUNT, MAX\_COUNT, Text);

End;

Procedure TAddRecForm.RecNameEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And AddRecButton.Enabled Then

AddRecButton.Click();

If (Key = VK\_RETURN) Or (Key = VK\_DOWN) Then

RecCostEdit.SetFocus();

End;

End.

**ChangeRecUnit.pas**

Var

ChangeRecForm: TChangeRecForm;

Implementation

{$R \*.dfm}

Procedure TChangeRecForm.ChangeRecButtonClick(Sender: TObject);

Var

CurToy: RToy;

Begin

CurToy.Name := RecNameEdit.Text;

CurToy.Cost := StrToInt(RecCostEdit.Text);

CurToy.Count := StrToInt(RecCountEdit.Text);

CurToy.MinAge := StrToInt(MinAgeEdit.Text);

CurToy.MaxAge := StrToInt(MaxAgeEdit.Text);

ChangeToy(CurIndex, CurToy);

End;

Procedure TChangeRecForm.FormCreate(RecIndex: Integer; Sender: TObject);

Var

CurToy: RToy;

Begin

CurIndex := RecIndex;

CurToy := GetRecFromFile(CurIndex);

RecNameEdit.Text := CurToy.Name;

RecCostEdit.Text := IntToStr(CurToy.Cost);

RecCountEdit.Text := IntToStr(CurToy.Count);

MinAgeEdit.Text := IntToStr(CurToy.MinAge);

MaxAgeEdit.Text := IntToStr(CurToy.MaxAge);

End;

Function TChangeRecForm.IsAllFieldCorrect(): Boolean;

Var

LBorder, RBorder: Integer;

IsAllFilled, IsCorrect: Boolean;

Begin

IsAllFilled := (RecNameEdit.Text <> '') And (RecCostEdit.Text <> '')

And (RecCountEdit.Text <> '') And (MinAgeEdit.Text <> '')

And (MaxAgeEdit.Text <> '');

If IsAllFilled Then

Begin

LBorder := StrToInt(MinAgeEdit.Text);

RBorder := StrToInt(MAxAgeEdit.Text);

IsCorrect := IsAllFilled And (LBorder < RBorder)

End

Else

IsCorrect := False;

IsAllFieldCorrect := IsCorrect;

End;

Procedure TChangeRecForm.AnyEditChange(Sender: TObject);

Begin

ChangeRecButton.Enabled := IsAllFieldCorrect();

End;

Procedure TChangeRecForm.MaxAgeEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And ChangeRecButton.Enabled Then

ChangeRecButton.Click();

If (Key = VK\_DOWN) And ChangeRecButton.Enabled Then

ChangeRecButton.SetFocus();

If (Key = VK\_UP) Or (Key = VK\_LEFT) Then

MinAgeEdit.SetFocus();

End;

Procedure TChangeRecForm.AgeEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With Sender as TEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_AGE, MAX\_AGE, Text);

End;

Procedure TChangeRecForm.MinAgeEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And ChangeRecButton.Enabled Then

ChangeRecButton.Click();

If (Key = VK\_DOWN) Or (Key = VK\_RIGHT) Or (Key = VK\_RETURN) Then

MaxAgeEdit.SetFocus();

If (Key = VK\_UP) Then

RecCountEdit.SetFocus();

End;

Procedure TChangeRecForm.RecCostEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And ChangeRecButton.Enabled Then

ChangeRecButton.Click();

If (Key = VK\_RETURN) Or (Key = VK\_DOWN) Then

RecCountEdit.SetFocus();

If (Key = VK\_UP) Then

RecNameEdit.SetFocus();

End;

Procedure TChangeRecForm.RecCostEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With RecCostEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_COST, MAX\_COST, Text);

End;

Procedure TChangeRecForm.RecCountEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And ChangeRecButton.Enabled Then

ChangeRecButton.Click();

If (Key = VK\_RETURN) Or (Key = VK\_DOWN) Then

MinAgeEdit.SetFocus();

If (Key = VK\_UP) Then

RecCostEdit.SetFocus();

End;

Procedure TChangeRecForm.RecCountEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With RecCountEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_COUNT, MAX\_COUNT, Text);

End;

Procedure TChangeRecForm.RecNameEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_RETURN) And ChangeRecButton.Enabled Then

ChangeRecButton.Click();

If (Key = VK\_RETURN) Or (Key = VK\_DOWN) Then

RecCostEdit.SetFocus();

End;

End.

**FindRecUnit.pas**

Var

FindRecForm: TFindRecForm;

Implementation

{$R \*.dfm}

Procedure TFindRecForm.AgeEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With Sender As TEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_AGE, MAX\_AGE, Text);

End;

Procedure TFindRecForm.WriteCorrectRecToFile(Path: String);

Var

CorrectionFile, BufferFile: TToyFile;

I, Age, MaxCost: Integer;

Toy: RToy;

Begin

Age := StrToInt(AgeEdit.Text);

MaxCost := StrToInt(RecCostEdit.Text);

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmReset);

OpenFile(BUFFER\_FILE\_PATH, BufferFile, FmRewrite);

For I := 1 To FileSize(CorrectionFile) Do

Begin

Read(CorrectionFile, Toy);

If (Toy.Cost <= MaxCost) And (Toy.MinAge <= Age) Then

Write(BufferFile, Toy);

End;

CloseFile(CorrectionFile);

CloseFile(BufferFile);

End;

Procedure TFindRecForm.FormCreate(Sender: TObject);

Begin

// Draw FixedRow information

RecordsGrid.Cells[0, 0] := '№';

RecordsGrid.Cells[1, 0] := 'Название';

RecordsGrid.Cells[2, 0] := 'Цена(BYN)';

RecordsGrid.Cells[3, 0] := 'Количество';

RecordsGrid.Cells[4, 0] := 'Возраст';

WriteCorrectRecToFile(BUFFER\_FILE\_PATH);

DrawRecordOnGrid(RecordsGrid, BUFFER\_FILE\_PATH);

DeleteFile(BUFFER\_FILE\_PATH);

End;

Function TFindRecForm.IsAllFieldCorrect: Boolean;

Var

IsCorrect: Boolean;

Begin

IsCorrect := (RecCostEdit.Text <> '') And (AgeEdit.Text <> '');

IsAllFieldCorrect := IsCorrect;

End;

Procedure TFindRecForm.AnyEditChange(Sender: TObject);

Begin

If IsAllFieldCorrect() Then

Begin

WriteCorrectRecToFile(BUFFER\_FILE\_PATH);

DrawRecordOnGrid(RecordsGrid, BUFFER\_FILE\_PATH);

End

Else

DrawRecordOnGrid(RecordsGrid, BUFFER\_FILE\_PATH);

DeleteFile(BUFFER\_FILE\_PATH);

End;

Procedure TFindRecForm.AgeEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_DOWN) Then

RecCostEdit.SetFocus();

End;

Procedure TFindRecForm.RecCostEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If ((SsShift In Shift) Or (SsCtrl In Shift)) And

Not((Key = VK\_RIGHT) Or (Key = VK\_LEFT)) Then

Key := 0;

If Key = VK\_DELETE Then

Key := 0;

If (Key = VK\_UP) Then

AgeEdit.SetFocus();

End;

Procedure TFindRecForm.RecCostEditKeyPress(Sender: TObject; Var Key: Char);

Begin

With RecCostEdit Do

TotalKeyPress(Key, SelStart, SelLength, MIN\_COST, MAX\_COST, Text);

End;

End.

**BackEndUnit.pas**

Type

TFileMode = (FmReset, FmRewrite);

TAge = 1 .. 120;

RToy = Record

Name: String[20];

Cost: Integer;

Count: Integer;

MinAge, MaxAge: TAge;

End;

TToyFile = File Of Rtoy;

Procedure TotalKeyPress(Var Key: Char; SelStart, SelLength: Integer;

Const MIN, MAX: Integer; Text: String);

Procedure AddRecord(Toy: RToy);

Procedure CloneFileToAnother(Var FromFile, ToFile: TToyFile);

Procedure OpenFile(Path: String; Var ThisFile: TToyFile; Mode: TFileMode);

Function GetRecFromFile(Index: Integer): RToy;

Procedure ChangeToy(Index: Integer; NewToy: RToy);

Procedure DeleteRec(Index: Integer);

Procedure DrawRecordOnGrid(Grid: TStringGrid; Path: String);

Const

STORAGE\_FILE\_PATH = 'StorageFile.txt';

СORRECTION\_FILE\_PATH = 'CorrectionFile.txt';

BUFFER\_FILE\_PATH = 'BufferFile.txt';

VOID = #0;

BACKSPACE = #8;

MIN\_COUNT = 0;

MAX\_COUNT = 2000000000;

MIN\_COST = 0;

MAX\_COST = 2000000000;

MIN\_AGE = 1;

MAX\_AGE = 120;

Implementation

Function IsToyCorrect(Toy: RToy): Boolean;

Begin

With Toy do

IsToyCorrect := (Length(Name) <= 21) And

(Cost >= MIN\_COST) And (Cost <= MAX\_COST)

And (Count >= MIN\_COUNT) And (Count <= MAX\_COUNT)

And (MinAge < MaxAge) And (MinAge >= MIN\_AGE)

And (MaxAge <= MAX\_AGE);

End;

Procedure DrawRecordOnGrid(Grid: TStringGrid; Path: String);

Var

CorrectionFile: TToyFile;

RecCount, I: Integer;

Toy: RToy;

IsFileIncorrect: Boolean;

Begin

OpenFile(Path, CorrectionFile, FmReset);

RecCount := FileSize(CorrectionFile);

IsFileIncorrect := False;

Grid.RowCount := RecCount + 1;

For I := 1 To RecCount Do

Begin

Read(CorrectionFile, Toy);

if IsToyCorrect(Toy) then

Begin

Grid.Cells[0, I] := IntToStr(I) + '.';

With Toy Do

Begin

Grid.Cells[1, I] := '"' + Name + '"';

Grid.Cells[2, I] := IntToStr(Cost);

Grid.Cells[3, I] := IntToStr(Count);

Grid.Cells[4, I] := IntToStr(MinAge) + '-' + IntToStr(MaxAge);

End;

End

Else

Begin

Grid.Cells[1, I] := 'файл с данными был пов';

Grid.Cells[2, I] := 'режден,пере';

Grid.Cells[3, I] := 'запустите';

Grid.Cells[4, I] := 'прогу';

IsFileIncorrect := True;

End;

End;

CloseFile(CorrectionFile);

if IsFileIncorrect then

DeleteFile(Path);

End;

Procedure DeleteRec(Index: Integer);

Var

CorrectionFile, BufferFile: TToyFile;

I: Integer;

Toy: RToy;

Begin

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmReset);

OpenFile(BUFFER\_FILE\_PATH, BufferFile, FmRewrite);

For I := 0 To FileSize(CorrectionFile) - 1 Do

If I <> Index Then

Begin

Read(CorrectionFile, Toy);

Write(BufferFile, Toy);

End

Else

Seek(CorrectionFile, FilePos(CorrectionFile) + 1);

CloseFile(CorrectionFile);

CloseFile(BufferFile);

DeleteFile(СORRECTION\_FILE\_PATH);

RenameFile(BUFFER\_FILE\_PATH, СORRECTION\_FILE\_PATH);

End;

Procedure ChangeToy(Index: Integer; NewToy: RToy);

Var

CorrectionFile: TToyFile;

Begin

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmReset);

Seek(CorrectionFile, Index);

Write(CorrectionFile, NewToy);

CloseFile(CorrectionFile);

End;

Function GetRecFromFile(Index: Integer): RToy;

Var

CorrectionFile: TToyFile;

Toy: RToy;

Begin

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmReset);

Seek(CorrectionFile, Index);

Read(CorrectionFile, Toy);

CloseFile(CorrectionFile);

GetRecFromFile := Toy;

End;

Procedure OpenFile(Path: String; Var ThisFile: TToyFile; Mode: TFileMode);

Begin

AssignFile(ThisFile, Path);

If Mode = FmReset Then

Begin

If Not(FileExists(Path)) Then

Rewrite(ThisFile)

Else

Reset(ThisFile);

End

Else

Rewrite(ThisFile);

End;

Procedure CloneFileToAnother(Var FromFile, ToFile: TToyFile);

Var

Toy: RToy;

Begin

While Not EoF(FromFile) Do

Begin

Read(FromFile, Toy);

if IsToyCorrect(Toy) then

Write(ToFile, Toy);

End;

End;

Procedure AddRecord(Toy: RToy);

Var

CorrectionFile: TToyFile;

Begin

OpenFile(СORRECTION\_FILE\_PATH, CorrectionFile, FmReset);

Seek(CorrectionFile, FileSize(CorrectionFile));

Write(CorrectionFile, Toy);

CloseFile(CorrectionFile);

End;

Function InsertKey(Index: Integer; SubStr: Char; SelLen: Integer;

Text: String): String;

Var

ResultText: String;

Begin

ResultText := Text;

If (SubStr = BACKSPACE) And (SelLen = 0) Then

Delete(ResultText, Index, 1)

Else

Begin

Delete(ResultText, Index + 1, SelLen);

If Substr <> BACKSPACE Then

ResultText.Insert(Index, String(SubStr));

End;

InsertKey := ResultText;

End;

Function CountOfSymbolInt(Num: Integer): Integer;

Var

NumLen: Integer;

Begin

NumLen := 0;

If Num < 0 Then

Inc(NumLen);

Repeat

Inc(NumLen);

Num := Num Div 10;

Until (Num = 0);

CountOfSymbolInt := NumLen;

End;

Procedure TotalKeyPress(Var Key: Char; SelStart, SelLength: Integer;

Const MIN, MAX: Integer; Text: String);

Var

ResultNum, RBorder, NumLen: Integer;

Buffer, Output: String;

Begin

Output := InsertKey(SelStart, Key, SelLength, Text);

If (Length(Output) <> 0) And (Output <> '-') Then

Begin

Try

ResultNum := StrToInt(Output);

Except

Key := VOID;

End;

If Key <> VOID Then

Begin

NumLen := CountOfSymbolInt(ResultNum);

If NumLen <> Length(Output) Then

Key := VOID;

If (ResultNum > MAX) Or (ResultNum < MIN) Then

Key := VOID;

End;

End

Else If (Output = '-') And (MIN >= 0) Then

Key := VOID;

End;

End.

**Код программы Java:**

import java.io.\*;

import java.util.Scanner;

record Toy (String name, int count, int cost, int minAge, int maxAge) {

@Override

public String toString() {

return String.format("| %-20s | %-10d | %-10d | %5d - %-5d |",

this.name, this.count, this.cost, this.minAge, this.maxAge);

}

}

public class Main {

enum ErrCode {

SUCCESS,

INCORRECT\_DATA,

INCORRECT\_NAME,

EMPTY\_LINE,

NO\_SUCH\_REC,

IO\_EXCEPTION,

TO\_MUCH\_RECORDS,

}

enum Choice {

addRec("Добавить"),

deleteRec("Удалить"),

changeRec("Изменить"),

save("Сохранить изменения"),

findByAge("Найти игрушки подходящие по возрасту"),

findByCost("Найти игрушку по цене"),

close("Закрыть");

private final String inf;

Choice (String infLine) {

this.inf = infLine;

}

private String getInf(){return this.ordinal() + ") " + this.inf;}

}

static final int MIN\_COUNT = 0,

MAX\_COUNT = 2000000000,

MIN\_COST = 0,

MAX\_COST = 2000000000,

MIN\_AGE = 1,

MAX\_AGE = 120,

MAX\_REC\_COUNT = 999;

static final String[] ERRORS = {"Удача",

"Данные не корректные или число слишком большое (должно быть

от %d до %d)\n",

"Имя должно быть максимум 20 символов в длинну",

"Строка пустая, будьте внимательны",

"Записи с таким номером нет в списке!",

"Ошибка чтения/записи файла",

"Записей не может быть больше чем %d\n"};

static final String STORAGE\_FILE\_PATH = "StorageFile.txt",

BUFFER\_FILE\_PATH = "BufferFile.txt",

CORRECTION\_FILE\_PATH = "CorrectionFile.txt",

INFORMATION\_TEXT = """

Каталог игрушек.

Инструкция:

1) Имя ограничено 20 символами

2) Кол-во и цена могут принимать значенияот 0 до 2000000000

3) Возраст принимает значения от 1 до 120

4) Чтобы внесенные изменения остались, требуется сохраниться

или выйти через кнопку

""",

START\_GRID\_LINE = """

------------------------------------------------------------------------

| № | ИМЯ | КОЛИЧЕСТВО | ЦЕНА(BYN) | ВОЗРАСТ |

------------------------------------------------------------------------

""";

static void printInf(Scanner input) {

System.out.println(INFORMATION\_TEXT);

System.out.println("нажмите enter чтобы продолжить");

input.nextLine();

}

static DataOutputStream openFileToWrite(String fileName) throws IOException{

File file = new File(fileName);

if (!file.exists())

file.createNewFile();

return new DataOutputStream(new FileOutputStream(fileName));

}

static DataInputStream openFileToRead(String fileName) throws IOException{

File file = new File(fileName);

if (!file.exists()) {

file.createNewFile();

}

return new DataInputStream(new FileInputStream(fileName));

}

static Toy readRec(DataInputStream file) throws IOException{

String name;

int count, cost, minAge, maxAge;

name = file.readUTF();

count = file.readInt();

cost = file.readInt();

minAge = file.readInt();

maxAge = file.readInt();

return new Toy(name, count, cost, minAge, maxAge);

}

static void writeRec(DataOutputStream file, Toy toy) throws IOException{

file.writeUTF(toy.name());

file.writeInt(toy.count());

file.writeInt(toy.cost());

file.writeInt(toy.minAge());

file.writeInt(toy.maxAge());

}

static void renameFileTo(String oldName, String newName) {

File oldFile = new File(newName);

oldFile.delete();

File newFile = new File(oldName);

newFile.renameTo(oldFile);

}

static void copyFile(String destFilePath, String soursFilePath) {

try(DataInputStream inputFile = openFileToRead(soursFilePath);

DataOutputStream outputFile = openFileToWrite(destFilePath)) {

while (inputFile.available() > 0)

writeRec(outputFile, readRec(inputFile));

} catch (IOException e) {

System.err.println(ERRORS[ErrCode.IO\_EXCEPTION.ordinal()]);

}

}

static void addRecToFile(String fileName, Toy toy) {

int countRec = 0;

try(DataOutputStream outputFile = openFileToWrite(BUFFER\_FILE\_PATH);

DataInputStream inputFile = openFileToRead(fileName)) {

while (inputFile.available() > 0) {

writeRec(outputFile, readRec(inputFile));

countRec++;

}

if (countRec < MAX\_REC\_COUNT)

writeRec(outputFile, toy);

else

System.err.printf(ERRORS[ErrCode.TO\_MUCH\_RECORDS.ordinal()], MAX\_REC\_COUNT);

} catch (IOException e) {

System.err.println(ERRORS[ErrCode.IO\_EXCEPTION.ordinal()]);

}

renameFileTo(BUFFER\_FILE\_PATH, fileName);

}

static void printFile(String fileName) {

Toy toy;

String line;

int count = 0;

try(DataInputStream file = openFileToRead(fileName)) {

System.out.printf(START\_GRID\_LINE);

while (file.available() > 0) {

count++;

toy = readRec(file);

line = String.format("| %-3d " + toy, count);

System.out.printf(line + "\n");

System.out.println("-".repeat(line.length()));

}

} catch (IOException e) {

System.err.println(ERRORS[ErrCode.IO\_EXCEPTION.ordinal()]);

}

}

static ErrCode deleteRec(int index) {

ErrCode err = ErrCode.NO\_SUCH\_REC;

try (DataInputStream inputFile = openFileToRead(CORRECTION\_FILE\_PATH);

DataOutputStream outputFile = openFileToWrite(BUFFER\_FILE\_PATH)) {

for (int i = 1; inputFile.available() > 0; i++)

if (i != index)

writeRec(outputFile, readRec(inputFile));

else {

err = ErrCode.SUCCESS;

readRec(inputFile);

}

} catch (IOException e) {

err = ErrCode.IO\_EXCEPTION;

}

renameFileTo(BUFFER\_FILE\_PATH, CORRECTION\_FILE\_PATH);

return err;

}

static ErrCode enterOneNum(int[] numberArr, Scanner input, final int MIN, final int MAX) {

int number = 0;

ErrCode err = ErrCode.SUCCESS;

try {

number = Integer.parseInt(input.nextLine());

} catch (NumberFormatException e) {

err = ErrCode.INCORRECT\_DATA;

}

if ((err == ErrCode.SUCCESS) && (number < MIN || number > MAX))

err = ErrCode.INCORRECT\_DATA;

numberArr[0] = err == ErrCode.SUCCESS ? number : 0;

return err;

}

static ErrCode enterNameConsole(String[] nameArr, Scanner input) {

ErrCode err = ErrCode.SUCCESS;

nameArr[0] = input.nextLine();

if (nameArr[0].length() > 20)

err = ErrCode.INCORRECT\_NAME;

if (nameArr[0].isEmpty())

err = ErrCode.EMPTY\_LINE;

return err;

}

static String getName(Scanner input) {

String[] nameArr = {""};

ErrCode err;

do {

err = enterNameConsole(nameArr, input);

if (err != ErrCode.SUCCESS) {

System.err.println(ERRORS[err.ordinal()]);

System.out.println("Введите снова");

}

} while (err != ErrCode.SUCCESS);

return nameArr[0];

}

static int getNumConsole(Scanner input, final int MIN, final int MAX) {

ErrCode err;

int[] numberArr = {0};

do {

err = enterOneNum(numberArr, input, MIN, MAX);

if (err != ErrCode.SUCCESS) {

System.err.printf(ERRORS[err.ordinal()], MIN, MAX);

System.out.println("Введите снова");

}

} while (err != ErrCode.SUCCESS);

return numberArr[0];

}

static Toy enterNewToyFromConsole(Scanner input) {

String name;

int count, cost, minAge, maxAge;

System.out.println("Введите имя игрушки:");

name = getName(input);

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

count = getNumConsole(input, MIN\_COUNT, MAX\_COUNT);

System.out.println("Введите цену игрушки (в BYN):");

cost = getNumConsole(input, MIN\_COST, MAX\_COST);

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

minAge = getNumConsole(input, MIN\_AGE, MAX\_AGE - 1);

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

maxAge = getNumConsole(input, minAge+1, MAX\_AGE);

return new Toy(name, count, cost, minAge, maxAge);

}

static ErrCode changeRec(int index, Scanner input) {

ErrCode err = ErrCode.NO\_SUCH\_REC;

try (DataInputStream inputFile = openFileToRead(CORRECTION\_FILE\_PATH);

DataOutputStream outputFile = openFileToWrite(BUFFER\_FILE\_PATH)) {

for (int i = 1; inputFile.available() > 0; i++)

if (i != index)

writeRec(outputFile, readRec(inputFile));

else {

err = ErrCode.SUCCESS;

readRec(inputFile);

writeRec(outputFile, enterNewToyFromConsole(input));

}

} catch (IOException e) {

err = ErrCode.IO\_EXCEPTION;

}

renameFileTo(BUFFER\_FILE\_PATH, CORRECTION\_FILE\_PATH);

return err;

}

static void printMenu() {

Choice[] choices = Choice.values();

for (Choice choice : choices) {

System.out.println(choice.getInf());

}

}

static Choice getChoice(Scanner input) {

int choice;

int maxChoice = Choice.values().length - 1;

choice = getNumConsole(input, 0, maxChoice);

return Choice.values()[choice];

}

static void findRec(Choice choice, int criteria) {

Toy toy;

boolean isValid;

try (DataInputStream inputFile = openFileToRead(CORRECTION\_FILE\_PATH);

DataOutputStream outputFile = openFileToWrite(BUFFER\_FILE\_PATH)) {

while (inputFile.available() > 0) {

toy = readRec(inputFile);

if (choice.equals(Choice.findByAge))

isValid = criteria >= toy.minAge();

else

isValid = criteria >= toy.cost();

if (isValid)

writeRec(outputFile, toy);

}

} catch (IOException e) {

System.err.println(ERRORS[ErrCode.IO\_EXCEPTION.ordinal()]);

}

printFile(BUFFER\_FILE\_PATH);

new File(BUFFER\_FILE\_PATH).delete();

}

static boolean doFunction(Scanner input) {

Choice choice = getChoice(input);

ErrCode err;

boolean isClose = false;

switch (choice) {

case addRec -> {

Toy toy = enterNewToyFromConsole(input);

addRecToFile(CORRECTION\_FILE\_PATH, toy);

}

case deleteRec -> {

System.out.print("Введите номер записи, которую хотите удалить: ");

int index = getNumConsole(input, 0, MAX\_COUNT);

err = deleteRec(index);

if (err != ErrCode.SUCCESS) {

System.err.println(ERRORS[err.ordinal()]);

}

}

case changeRec -> {

System.out.print("Введите номер записи, которую хотите изменить: ");

int index = getNumConsole(input, 0, MAX\_COUNT);

err = changeRec(index, input);

if (err != ErrCode.SUCCESS) {

System.err.println(ERRORS[err.ordinal()]);

}

}

case save -> {

copyFile(STORAGE\_FILE\_PATH, CORRECTION\_FILE\_PATH);

}

case findByAge -> {

System.out.print("Введите возраст, для которого хотите найти игрушки: ");

int age = getNumConsole(input, MIN\_AGE, MAX\_AGE);

findRec(choice, age);

System.out.println("нажмите enter чтобы продолжить");

input.nextLine();

}

case findByCost -> {

System.out.print("Введите цену, до которой хотите найти игрушки: ");

int cost = getNumConsole(input, MIN\_COST, MAX\_COST);

findRec(choice, cost);

System.out.println("нажмите enter чтобы продолжить");

input.nextLine();

}

case close -> {

renameFileTo(CORRECTION\_FILE\_PATH, STORAGE\_FILE\_PATH);

isClose = true;

}

}

return isClose;

}

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

printInf(input);

copyFile(CORRECTION\_FILE\_PATH, STORAGE\_FILE\_PATH);

boolean isClose;

do {

printFile(CORRECTION\_FILE\_PATH);

printMenu();

isClose = doFunction(input);

} while (!isClose);

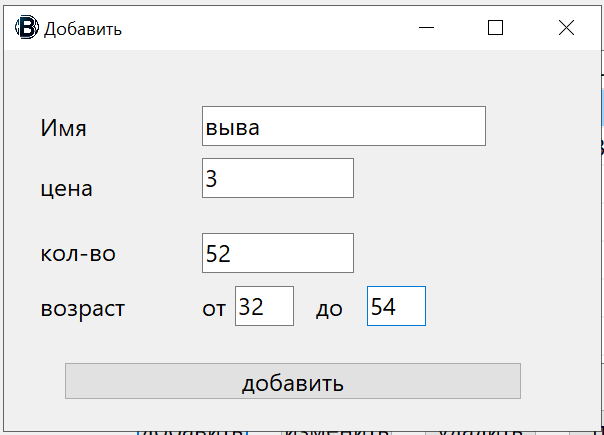
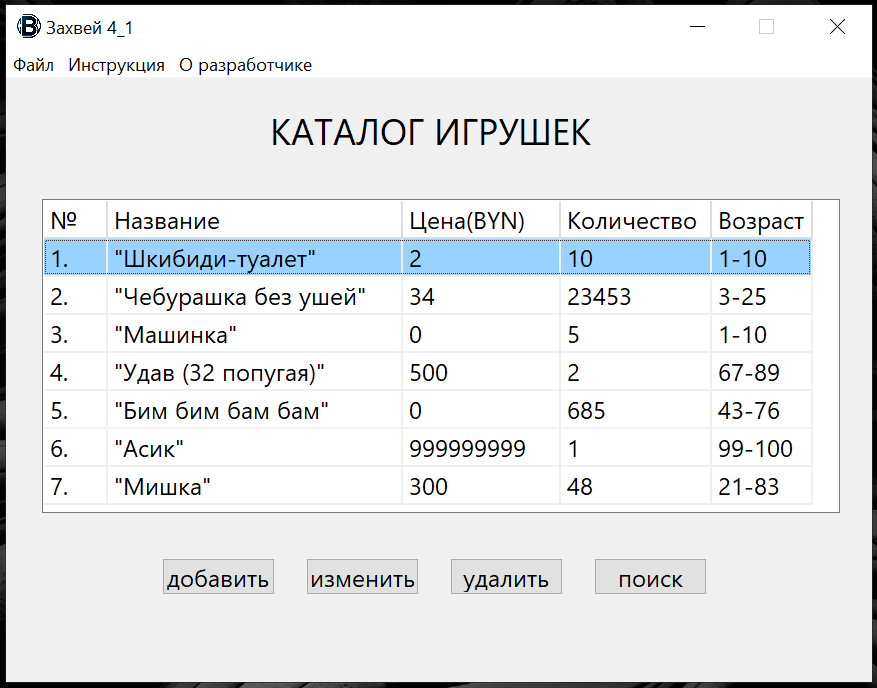
input.close();

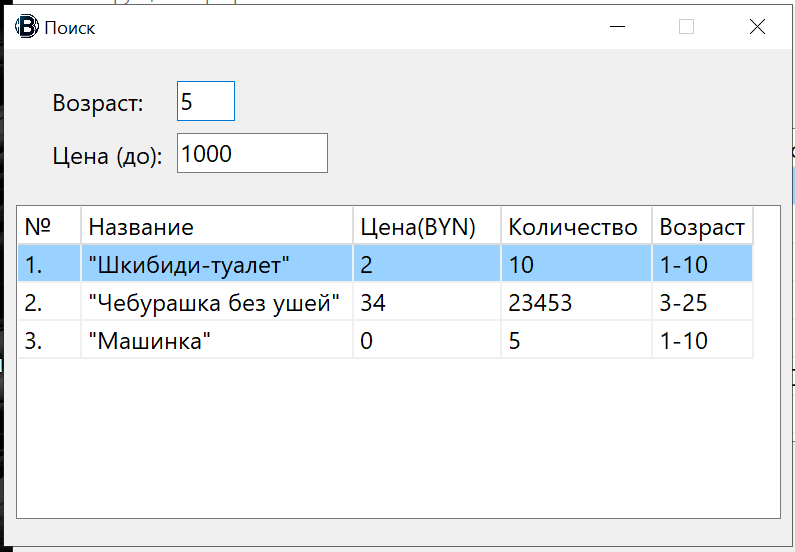
}

}

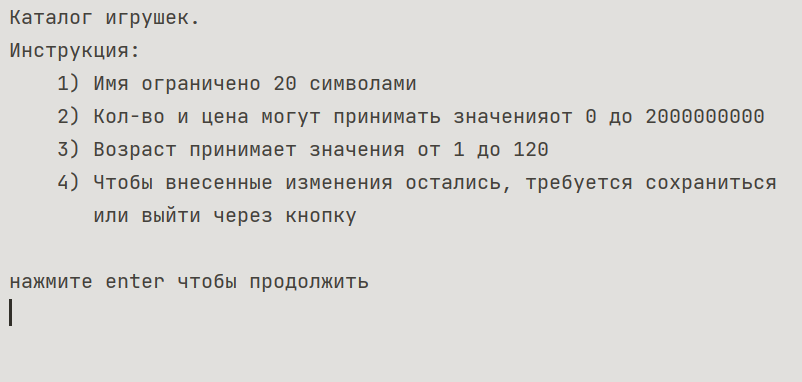
**Скриншоты:**

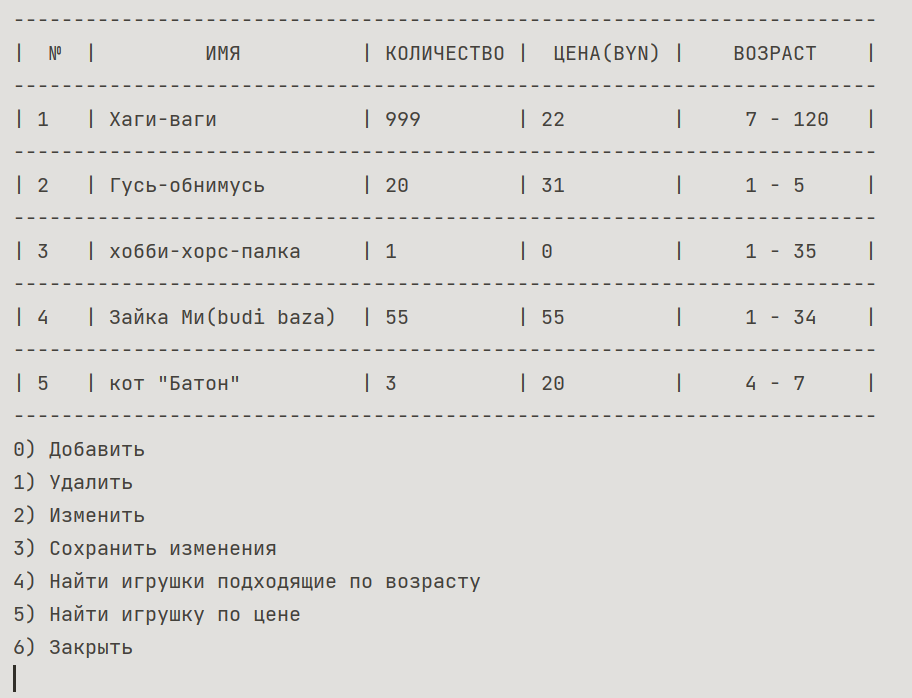
**Delphi:**

****

****

**Java:**

****

****

****

**Блок-схема:**

