Донецкий Национальный Технический Университет

Лабораторная работа № 1

«Организация таблиц у трансляторах и работа с ними»

Выполнил:

ст. группы ИПЗ -13

Лысенко А. С.

Проверила:

доцент каф. ПМИ

Дмитриева О. А.

Покровск 2017

Задание

**1.2.1 Таблиці для транслятора з мови C**

4) Таблиця операцій і її ініціалізація.

|  |  |  |  |
| --- | --- | --- | --- |
| id | обозначение операции | приоритет | ассоциативность |
| 1 | = | 13 | Справа-налево |
| 2 | + | 4 | Слева-направо |
| 3 | - | 4 | Слева-направо |
| 4 | \* | 3 | Слева-направо |
| 5 | / | 3 | Слева-направо |
| 6 | % | 3 | Слева-направо |
| 7 | ++ | 1 | Справа-налево |
| 8 | -- | 1 | Справа-налево |
| 9 | == | 7 | Слева-направо |
| 10 | != | 7 | Слева-направо |
| 11 | > | 6 | Слева-направо |
| 12 | < | 6 | Слева-направо |
| 13 | >= | 6 | Слева-направо |
| 14 | <= | 6 | Слева-направо |
| 15 | ! | 1 | Справа-налево |
| 16 | && | 11 | Слева-направо |
| 17 | || | 12 | Слева-направо |
| 18 | ~ | 1 | Справа-налево |
| 19 | & | 8 | Слева-направо |
| 20 | | | 10 | Слева-направо |
| 21 | ^ | 9 | Слева-направо |
| 22 | << | 5 | Слева-направо |
| 23 | >> | 5 | Слева-направо |
| 24 | +(ун) | 2 | Справа-налево |
| 25 | -(ун) | 2 | Справа-налево |

Программа написанная на javafx с использованием графического редактора scene builder 2.0. Программа представляет собой таблицу с функциями, добавить, удалить, изменить, данные задаются статически в виде массива объектов.

Main.java – запускает главное окно программы

**package** sample;  
**import** javafx.application.Application;  
**import** javafx.fxml.FXMLLoader;  
**import** javafx.scene.Parent;  
**import** javafx.scene.Scene;  
**import** javafx.stage.Stage;  
**import** java.awt.event.ActionEvent;  
**public class** Main **extends** Application {  
 @Override  
 **public void** start(Stage primaryStage) **throws** Exception{  
 FXMLLoader fxmlLoader = **new** FXMLLoader();  
 fxmlLoader.setLocation(getClass().getResource(**"sample.fxml"**));  
 Parent fxmlMain = fxmlLoader.load();  
 Controller controller = fxmlLoader.getController();  
 controller.setMainStage(primaryStage);  
 primaryStage.setTitle(**"Table"**);  
 primaryStage.setMinHeight(375);  
 primaryStage.setMinWidth(650);  
 primaryStage.setScene(**new** Scene(fxmlMain, 650, 375));  
 primaryStage.show();  
 testData(); // функция заполнения данных  
 }  
 **private void** testData()  
 {  
 CollectionTable collectionTable = **new** CollectionTable();  
 collectionTable.fillTestData();  
 *//collectionTable.print();* }  
 **public static void** main(String[] args) {  
 *launch*(args);  
 }  
}

Controller.java – обработчик кнопок главного окна

**package** sample;  
**import** javafx.collections.ListChangeListener;  
**import** javafx.event.ActionEvent;  
**import** javafx.event.EventHandler;  
**import** javafx.fxml.FXML;  
**import** javafx.fxml.FXMLLoader;  
**import** javafx.scene.Node;  
**import** javafx.scene.Parent;  
**import** javafx.scene.Scene;  
**import** javafx.scene.control.\*;  
**import** javafx.scene.control.Button;  
**import** javafx.scene.control.Label;  
**import** javafx.scene.control.TextField;  
**import** javafx.scene.control.cell.PropertyValueFactory;  
**import** javafx.scene.input.MouseEvent;  
**import** javafx.stage.Modality;  
**import** javafx.stage.Stage;  
**import** javafx.stage.Window;  
  
**import** java.io.IOException;  
  
**public class** Controller  
{  
  
 **private** Stage **mainStage**;  
 @FXML  
 **public** TextField **txtAssociation**;  
 @FXML  
 **private** TableColumn<Object, String> **columnId**;  
  
 @FXML  
 **private** TableColumn<Object, String> **columnOperation**;  
  
 @FXML  
 **private** TableColumn<Object, String> **columnPriority**;  
  
 @FXML  
 **private** TableColumn<Object, String> **columnAssociation**;  
  
 @FXML  
 **private** Button **btnEdit**;  
  
 @FXML  
 **private** Button **btnDelete**;  
  
 @FXML  
 **private** TextField **txtSearch**;  
  
 @FXML  
 **private** Button **btnSearch**;  
  
 @FXML  
 **private** TableView **tableOperation**;  
  
 @FXML  
 **private** Label **tableCount**;  
  
 @FXML  
 **private** TextField **txtOperation**;  
  
 @FXML  
 **private** TextField **txtPriority**;  
  
 @FXML  
 **private** Button **btnOk**;  
  
 @FXML  
 **private** Button **btnCancel**;  
  
 @FXML  
 **private** Button **btnAdd**;  
  
 **private** Parent **fxmlEdit**;  
 **private** FXMLLoader **fxmlLoader** = **new** FXMLLoader();  
 **private** EditController **editController**;  
 **private** Stage **editDialogStage**;  
  
 **private** CollectionTable **collectionTableImpl** = **new** CollectionTable();  
  
 @FXML  
 **public void** initialize() {  
  
 **columnOperation**.setCellValueFactory(**new** PropertyValueFactory<Object, String>(**"operation"**));  
 **columnPriority**.setCellValueFactory(**new** PropertyValueFactory<Object, String>(**"priority"**));  
 **columnAssociation**.setCellValueFactory(**new** PropertyValueFactory<Object, String>(**"association"**));  
 **columnId**.setCellValueFactory(**new** PropertyValueFactory<Object, String>(**"id"**));  
  
 initListeners();  
  
 **collectionTableImpl**.fillTestData();  
  
 **tableOperation**.setItems(**collectionTableImpl**.getObjectList());  
  
 initLoader();  
  
 }  
  
 **public void** setMainStage(Stage mainStage) {  
 **this**.**mainStage** = mainStage;  
 }  
  
 **private void** initListeners() {  
 **collectionTableImpl**.getObjectList().addListener(**new** ListChangeListener<Object>() {  
 @Override  
 **public void** onChanged(Change<? **extends** Object> c) {  
 updateCountLabel();  
 }  
 });  
  
 **tableOperation**.setOnMouseClicked(**new** EventHandler<MouseEvent>() {  
 @Override  
 **public void** handle(MouseEvent event) {  
 **if**(event.getClickCount() == 2)  
 {  
 **editController**.setObject((Object) **tableOperation**.getSelectionModel().getSelectedItem());  
 showDialog();  
 }  
 }  
 });  
}  
**private void** initLoader()  
{  
 **try** {  
 **fxmlLoader**.setLocation(getClass().getResource(**"edit.fxml"**));  
 **fxmlEdit** = **fxmlLoader**.load();  
 **editController** = **fxmlLoader**.getController();  
 } **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
  
 **private void** updateCountLabel()  
 {  
 **tableCount**.setText(**"Количество записей: "**+ **collectionTableImpl**.getObjectList().size());  
 }  
 **public void** actionButtonPressed(ActionEvent actionEvent) {  
  
 java.lang.Object source = actionEvent.getSource();  
  
 **if**(!(source **instanceof** Button))  
 {  
 **return**;  
 }  
 Button clickedButton = (Button) source;  
 *// Object selectedObject = (Object)tableOperation.getSelectionModel().getSelectedItem();  
 //editController.setObject(selectedObject);* **switch** (clickedButton.getId())  
 {  
 **case "btnAdd"**:  
 **editController**.setObject(**new** Object());  
 showDialog();  
 **collectionTableImpl**.addRecord(**editController**.getObject());  
 **break**;  
 **case "btnEdit"**:  
 **editController**.setObject((Object) **tableOperation**.getSelectionModel().getSelectedItem());  
 showDialog();  
 **break**;  
 **case "btnDelete"**:  
 **collectionTableImpl**.deleteRecord((Object) **tableOperation**.getSelectionModel().getSelectedItem());  
 **break**;  
 }  
 */\* Stage stage = new Stage();  
 Parent root = FXMLLoader.load(getClass().getResource("edit.fxml"));  
 stage.setTitle("Редактирование записи");  
 stage.setMinHeight(150);  
 stage.setMinWidth(300);  
 stage.setResizable(false);  
 stage.setScene(new Scene(root));  
 stage.initModality(Modality.WINDOW\_MODAL);  
 stage.initOwner(((Node)actionEvent.getSource()).getScene().getWindow());  
 stage.show();\*/* }  
**private void** showDialog()  
{  
 **if** (**editDialogStage** == **null**)  
 {  
 **editDialogStage** = **new** Stage();  
 **editDialogStage**.setTitle(**"Редактирование записи"**);  
 **editDialogStage**.setMinHeight(150);  
 **editDialogStage**.setMinWidth(300);  
 **editDialogStage**.setResizable(**false**);  
 **editDialogStage**.setScene(**new** Scene(**fxmlEdit**));  
 **editDialogStage**.initModality(Modality.***WINDOW\_MODAL***);  
 **editDialogStage**.initOwner(**mainStage**);  
 }  
  
 **editDialogStage**.showAndWait(); *// для ожидания закрытия окна  
  
 //editDialogStage.show();*}  
}

Action.java – интерфейс коллекций

**package** sample;  
**public interface** Action  
{  
 **void** addRecord(Object object);  
 **void** changeRecord(Object object);  
 **void** deleteRecord(Object object);  
}

sample.fxml – форма главного окна

*<?***xml version="1.0" encoding="UTF-8"***?>  
  
<?***import javafx.geometry.\****?>  
<?***import javafx.scene.control.\****?>  
<?***import java.lang.\****?>  
<?***import javafx.scene.layout.\****?>  
<?***import javafx.geometry.Insets***?>  
<?***import javafx.scene.layout.GridPane***?>  
<?***import javafx.scene.control.Button***?>  
<?***import javafx.scene.control.Label***?>*<**VBox maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="381.0" prefWidth="637.0"  
 xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller"**>  
 <**children**>  
 <**HBox maxHeight="50.0" prefHeight="50.0" prefWidth="200.0"**>  
 <**children**>  
 <**Button fx:id="btnAdd" mnemonicParsing="false" onAction="#actionButtonPressed" text="Добавить"**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Button**>  
 <**Button fx:id="btnEdit" mnemonicParsing="false" onAction="#actionButtonPressed" text="Изменить"**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Button**>  
 <**Button fx:id="btnDelete" mnemonicParsing="false" onAction="#actionButtonPressed" text="Удалить"**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Button**>  
 </**children**>  
 </**HBox**>  
 <**AnchorPane prefHeight="167.0" prefWidth="597.0" VBox.vgrow="ALWAYS"**>  
 <**children**>  
 <**TableView fx:id="tableOperation" minHeight="-Infinity" minWidth="-Infinity" prefHeight="124.0" prefWidth="597.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0"**>  
 <**columns**>  
 <**TableColumn fx:id="columnId" prefWidth="85.0" text="id"** />  
 <**TableColumn fx:id="columnOperation" prefWidth="167.0" text="Обозначение операции"** />  
 <**TableColumn fx:id="columnPriority" minWidth="0.0" prefWidth="147.0" text="Приоритет"** />  
 <**TableColumn fx:id="columnAssociation" prefWidth="197.0" text="Ассоциативность"** />  
 </**columns**>  
 <**columnResizePolicy**>  
 <**TableView fx:constant="CONSTRAINED\_RESIZE\_POLICY"** />  
 </**columnResizePolicy**>  
 </**TableView**>  
 </**children**>  
 <**VBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**VBox.margin**>  
 </**AnchorPane**>  
 <**AnchorPane prefHeight="50.0" prefWidth="200.0"**>  
 <**children**>  
 <**HBox layoutX="4.0" layoutY="47.0" prefHeight="57.0" prefWidth="630.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0"**>  
 <**children**>  
 <**Label fx:id="tableCount" prefHeight="17.0" prefWidth="138.0" text="Количество запсей: "**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Label**>  
 </**children**>  
 </**HBox**>  
 </**children**>  
 </**AnchorPane**>  
 </**children**>  
</**VBox**>

edit.fxml – формы дополнительных окон (добавить, удалить, изменить)

*<?***xml version="1.0" encoding="UTF-8"***?>  
<?***import javafx.geometry.\****?>  
<?***import javafx.scene.control.\****?>  
<?***import java.lang.\****?>  
<?***import javafx.scene.layout.\****?>  
<?***import javafx.geometry.Insets***?>  
<?***import javafx.scene.layout.GridPane***?>  
<?***import javafx.scene.control.Button***?>  
<?***import javafx.scene.control.Label***?>*<**VBox maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="381.0" prefWidth="637.0"  
 xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller"**>  
 <**children**>  
 <**HBox maxHeight="50.0" prefHeight="50.0" prefWidth="200.0"**>  
 <**children**>  
 <**Button fx:id="btnAdd" mnemonicParsing="false" onAction="#actionButtonPressed" text="Добавить"**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Button**>  
 <**Button fx:id="btnEdit" mnemonicParsing="false" onAction="#actionButtonPressed" text="Изменить"**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Button**>  
 <**Button fx:id="btnDelete" mnemonicParsing="false" onAction="#actionButtonPressed" text="Удалить"**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Button**>  
 </**children**>  
 </**HBox**>  
 <**AnchorPane prefHeight="167.0" prefWidth="597.0" VBox.vgrow="ALWAYS"**>  
 <**children**>  
 <**TableView fx:id="tableOperation" minHeight="-Infinity" minWidth="-Infinity" prefHeight="124.0" prefWidth="597.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0"**>  
 <**columns**>  
 <**TableColumn fx:id="columnId" prefWidth="85.0" text="id"** />  
 <**TableColumn fx:id="columnOperation" prefWidth="167.0" text="Обозначение операции"** />  
 <**TableColumn fx:id="columnPriority" minWidth="0.0" prefWidth="147.0" text="Приоритет"** />  
 <**TableColumn fx:id="columnAssociation" prefWidth="197.0" text="Ассоциативность"** />  
 </**columns**>  
 <**columnResizePolicy**>  
 <**TableView fx:constant="CONSTRAINED\_RESIZE\_POLICY"** />  
 </**columnResizePolicy**>  
 </**TableView**>  
 </**children**>  
 <**VBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**VBox.margin**>  
 </**AnchorPane**>  
 <**AnchorPane prefHeight="50.0" prefWidth="200.0"**>  
 <**children**>  
 <**HBox layoutX="4.0" layoutY="47.0" prefHeight="57.0" prefWidth="630.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0"**>  
 <**children**>  
 <**Label fx:id="tableCount" prefHeight="17.0" prefWidth="138.0" text="Количество запсей: "**>  
 <**HBox.margin**>  
 <**Insets bottom="20.0" left="20.0" right="20.0" top="20.0"** />  
 </**HBox.margin**>  
 </**Label**>  
 </**children**>  
 </**HBox**>  
 </**children**>  
 </**AnchorPane**>  
 </**children**>  
</**VBox**>

EditController.java – обработчик событий доп. окон (добавить, удалить, изменить)

**package** sample;  
  
**import** javafx.event.ActionEvent;  
**import** javafx.fxml.FXML;  
**import** javafx.scene.Node;  
**import** javafx.scene.control.TextField;  
**import** javafx.stage.Stage;  
  
**public class** EditController  
{  
 @FXML  
 **private** TextField **txtOperation**;  
  
 @FXML  
 **private** TextField **txtPriority**;  
  
 @FXML  
 **public** TextField **txtAssociation**;  
  
 *//метод для обработки окна-справочника, кнопка - отмена* **public void** actionClose(ActionEvent actionEvent)  
 {  
 Node source = (Node) actionEvent.getSource();  
 Stage stage = (Stage) source.getScene().getWindow();  
 stage.hide();  
 }  
  
 **private** Object **object**;  
  
 **public void** setObject(Object object)  
 {  
 **this**.**object** = object;  
  
  
 **txtOperation**.setText(object.getOperation());  
 **txtPriority**.setText(object.getPriority());  
 **txtAssociation**.setText(object.getAssociation());  
  
 }  
 **public** TextField getTxtOperation() {  
 **return txtOperation**;  
 }  
 **public** TextField getTxtPriority() {  
 **return txtPriority**;  
 }  
 **public** TextField getTxtAssociation() {  
 **return txtAssociation**;  
 }  
 **public** Object getObject() {  
 **return object**;  
 }  
 **public void** actionSave(ActionEvent actionEvent)  
 {  
 **object**.setOperation(**txtOperation**.getText());  
 **object**.setPriority(**txtPriority**.getText());  
 **object**.setAssociation(**txtAssociation**.getText());  
 actionClose(actionEvent);  
 }  
}

СollectionTable.java – реализация интерфейса коллекций

**package** sample;  
  
**import** javafx.collections.FXCollections;  
**import** javafx.collections.ObservableList;  
  
*// класс реализовывет интерфейс с помощбью коллекции***public class** CollectionTable **implements** Action {  
  
 **private** ObservableList<Object> **objectList** = FXCollections.*observableArrayList*();  
  
 @Override  
 **public void** deleteRecord(Object object) {  
 **objectList**.remove(object);  
 }  
 *// для коллекции не используется, но пригодится для случая, когда данные хранятся в БД и пр.* @Override  
 **public void** changeRecord(Object object) {  
 *// т.к. коллекция является хринилищем, то ничего обновлять не нужно  
 // если бы данные хранились в БД или в файле - в этом методе нужно было бы обновить соотв. запись* }  
  
 @Override  
 **public void** addRecord(Object object) {  
 **objectList**.add(object);  
 }  
 **public** ObservableList<Object> getObjectList()  
 {  
 **return objectList**;  
 }  
  
  
  
 **public void** print()  
 {  
 *//objectList.remove(0);* **int** number = 0;  
 System.***out***.println();  
 **for**(Object object : **objectList**)  
 {  
 number++;  
 System.***out***.println(number+**") Операция: "** + object.getOperation()  
 + **" Приоритет: "** + object.getPriority()  
 + **" Ассоциативность "** + object.getAssociation());  
 }  
  
 }  
 **public void** fillTestData()  
 {  
 **objectList**.add(**new** Object(1,**""**, **"слева-направо"**, **"2"**));  
 **objectList**.add(**new** Object(2,**"- "**, **"слева-направо"**, **"3"**));  
 **objectList**.add(**new** Object(3,**"/ "**, **"слева-направо"**, **"4"**));  
 **objectList**.add(**new** Object(4,**"\* "**, **"слева-направо"**, **"5"**));  
 **objectList**.add(**new** Object(5,**"= "**, **"слева-направо"**, **"6"**));  
 **objectList**.add(**new** Object(6,**"=="**, **"слева-направо"**, **"7"**));  
 **objectList**.add(**new** Object(7,**"% "**, **"слева-направо"**, **"8"**));  
 }  
}

Object.java – заполнения данных в главной форме

**package** sample;  
  
**import** javafx.collections.FXCollections;  
**import** javafx.collections.ObservableList;  
  
*// класс реализовывет интерфейс с помощбью коллекции***public class** CollectionTable **implements** Action {  
  
 **private** ObservableList<Object> **objectList** = FXCollections.*observableArrayList*();  
  
 @Override  
 **public void** deleteRecord(Object object) {  
 **objectList**.remove(object);  
 }  
 *// для коллекции не используется, но пригодится для случая, когда данные хранятся в БД и пр.* @Override  
 **public void** changeRecord(Object object) {  
 *// т.к. коллекция является хринилищем, то ничего обновлять не нужно  
 // если бы данные хранились в БД или в файле - в этом методе нужно было бы обновить соотв. запись* }  
  
 @Override  
 **public void** addRecord(Object object) {  
 **objectList**.add(object);  
 }  
 **public** ObservableList<Object> getObjectList()  
 {  
 **return objectList**;  
 }  
  
 **public void** print()  
 {  
 *//objectList.remove(0);* **int** number = 0;  
 System.***out***.println();  
 **for**(Object object : **objectList**)  
 {  
 number++;  
 System.***out***.println(number+**") Операция: "** + object.getOperation()  
 + **" Приоритет: "** + object.getPriority()  
 + **" Ассоциативность "** + object.getAssociation());  
 }  
  
 }  
  
 **public void** fillTestData()  
 {  
 **objectList**.add(**new** Object(1,**""**, **"слева-направо"**, **"2"**));  
 **objectList**.add(**new** Object(2,**"- "**, **"слева-направо"**, **"3"**));  
 **objectList**.add(**new** Object(3,**"/ "**, **"слева-направо"**, **"4"**));  
 **objectList**.add(**new** Object(4,**"\* "**, **"слева-направо"**, **"5"**));  
 **objectList**.add(**new** Object(5,**"= "**, **"слева-направо"**, **"6"**));  
 **objectList**.add(**new** Object(6,**"=="**, **"слева-направо"**, **"7"**));  
 **objectList**.add(**new** Object(7,**"% "**, **"слева-направо"**, **"8"**));  
 }  
  
}

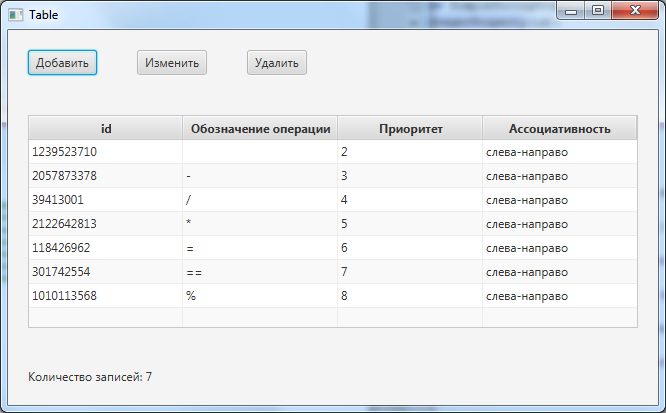


Рис. 1 – главная форма программы

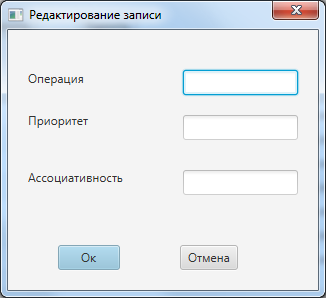


Рис. 2 – операция добавления записи

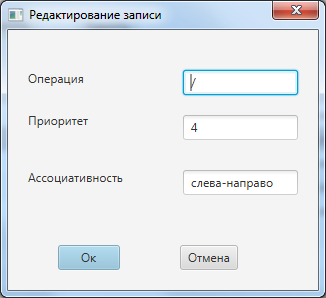


Рис. 3 - операция изменения записи

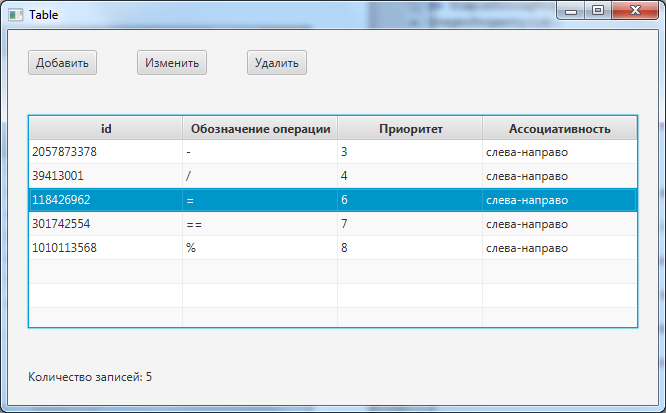


Рис. 4 – удаление записей из таблицы