МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Програмна інженерія та інформаційні технології управління»

Звіт з лабораторної роботи №3

З предмету «Об’єктно-орієнтоване програмування»

Виконав

Студент групи КН-36а

Рубан Ю.Д.

Перевірили:

Козуля М.М.

Кізілов О.С.

Харків 2017

Створення програм графічного інтерфейсу користувача

1 Завдання на лабораторну роботу

1.1 Індивідуальне завдання

Необхідно реалізувати мовою Java за допомогою засобів JavaFX застосунок графічного інтерфейсу користувача, в якому здійснюється обробка даних індивідуальних завдань попередніх лабораторних робіт. Головне вікно повинно містити меню, в якому необхідно реалізувати такі функції:

* створення нового набору даних
* завантаження даних з XML-документу для редагування
* зберігання змінених даних в XML-документі
* пошук за ознаками, визначеними в [лабораторній роботі № 5 курсу "Алгоритмізація та програмування" (друга частина)](http://iwanoff.96.lt/algorithmization_2/LabTraining05.html)
* здійснення сортування за ознаками, визначеними в [лабораторній роботі № 6 курсу "Алгоритмізація та програмування" (друга частина)](http://iwanoff.96.lt/algorithmization_2/LabTraining06.html)
* отримання вікна "Про програму" з даними про програму і автора.

У лівій частині вікна слід розташувати рядки для введення скалярних даних, область відображення для результатів пошуку, а також кнопки, які забезпечують виконання основних функцій програми. В середній частині вікна слід розташувати таблицю для відображення та редагування даних.

Розробці програми графічного інтерфейсу користувача повинне передувати створення консольного застосунку, в якому здійснюється читання даних з підготовленого XML-документу, пошук певних даних, сортування і зберігання у новому XML-документі. Для роботи з XML скористатися технологією зв'язування даних.

1.2 Перелік для опису місяців року

Створити перелік "Місяць". Необхідно визначати у конструкторі і зберігати кількість днів (для невисокосного року). Додати методи отримання попереднього та наступного місяця, а також функцію, яка повертає сезон для кожного місяця. Передбачити виведення місяців українською (російською) мовою. Створити статичну функцію виведення даних про усі місяці. Протестувати переліку в функції main() тестового класу.

1.3 Дані про користувачів

Представити дані про користувачів у вигляді асоціативного масиву (ім'я / пароль) з припущенням, що всі імена користувачів різні. Вивести дані про користувачів з довжиною пароля більше 6 символів.

1.4 Міні-калькулятор

Створити застосунок графічного інтерфейсу користувача, в якому після введення чисел у двох рядках типу TextFieldвиконується одна з чотирьох арифметичних дій (залежно від вибраної кнопки RadioButton). Результат виводиться в інше текстове поле.

1.5 Словник (додаткове завдання)

Розробити програму графічного інтерфейсу користувача перегляду слів невеличкого англо-українського (англо-російського) словника. Реалізувати функції пошуку слова, додавання нових слів.

2 Хід виконання роботи

**Завдання 1.** Реалізовано мовою Java за допомогою засобів JavaFX застосунок графічного інтерфейсу користувача, в якому здійснюється обробка даних індивідуальних завдань попередніх лабораторних робіт. Головне вікно містить меню, в якому реалізувано такі функції:

* створення нового набору даних
* завантаження даних з XML-документу для редагування
* зберігання змінених даних в XML-документі
* пошук за ознаками, визначеними в [лабораторній роботі № 5 курсу "Алгоритмізація та програмування" (друга частина)](http://iwanoff.96.lt/algorithmization_2/LabTraining05.html)
* здійснення сортування за ознаками, визначеними в [лабораторній роботі № 6 курсу "Алгоритмізація та програмування" (друга частина)](http://iwanoff.96.lt/algorithmization_2/LabTraining06.html)
* отримання вікна "Про програму" з даними про програму і автора.

Код програми:

Клас Controller:

package sample;  
  
  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.TableColumn;  
import javafx.scene.control.TableView;  
import javafx.scene.control.TextField;  
import javafx.scene.control.cell.TextFieldTableCell;  
import javafx.stage.FileChooser;  
import javafx.util.converter.NumberStringConverter;  
  
import java.io.File;  
  
  
public class Controller {  
 Main main;  
 @FXML  
 private TextField addTitle;  
 @FXML  
 private TextField addName;  
 @FXML  
 private TextField addVisitors;  
 @FXML  
 private TextField addComment;  
 @FXML  
 private TableView<Show> ShowTableView;  
 @FXML  
 private TableView<Day> DayTableView;  
 @FXML  
 private TableColumn<Show, String> titleSH;  
 @FXML  
 private TableColumn<Show, String> nameSH;  
 @FXML  
 private TableColumn<Day, Number> visitorsNum;  
 @FXML  
 private TableColumn<Day, String> commentN;  
  
 public void initialize() {  
 titleSH.setCellFactory(TextFieldTableCell.*forTableColumn*());  
 titleSH.setCellValueFactory(cellBack -> cellBack.getValue().getPropertyTitle());  
 titleSH.setOnEditCommit(event -> event.getTableView().getItems().get(event.getTablePosition().getRow()).setTitle(event.getNewValue()));  
 nameSH.setCellValueFactory(cellBack -> cellBack.getValue().getPropertyName());  
 nameSH.setCellFactory(TextFieldTableCell.*forTableColumn*());  
 nameSH.setOnEditCommit(  
 event -> (event.getTableView().getItems().get(  
 event.getTablePosition().getRow())  
 ).setName(event.getNewValue())  
 );  
  
 showDetails(null);  
  
 visitorsNum.setCellFactory(TextFieldTableCell.*forTableColumn*(new NumberStringConverter()));  
 visitorsNum.setOnEditCommit(  
 event -> (event.getTableView().getItems().get(  
 event.getTablePosition().getRow())  
 ).setCountOfvisitors(event.getNewValue().intValue())  
 );  
 commentN.setCellFactory(TextFieldTableCell.*forTableColumn*());  
 commentN.setOnEditCommit(  
 event -> (event.getTableView().getItems().get(  
 event.getTablePosition().getRow())  
 ).setComment(event.getNewValue())  
 );  
  
 ShowTableView.getSelectionModel().selectedItemProperty().addListener(  
 (observable, old, selected) -> showDetails(selected));  
 }  
  
  
 private void showDetails(Show tramStation) {  
 if (tramStation != null) {  
 visitorsNum.setCellValueFactory(cellBack -> cellBack.getValue().passengersProperty());  
 commentN.setCellValueFactory(cellBack -> cellBack.getValue().commentProperty());  
 ObservableList<Day> temp = FXCollections.*observableArrayList*();  
 temp.addAll(tramStation.getDays());  
 DayTableView.setItems(temp);  
 }  
 }  
  
 public void setMain(Main main) {  
 this.main = main;  
 ObservableList<Show> temp = FXCollections.*observableArrayList*();  
 temp.addAll(main.getList());  
 ShowTableView.setItems(temp);  
 }  
  
 @FXML  
 private void handleSaveAs() {  
 main.saveToXML(main.getStage(),ShowTableView.getItems());  
 }  
  
 @FXML  
 private void handleAddShow() {  
 if(addName.getText().length() > 1 && addTitle.getText().length() > 1) {  
 main.getList().add(new Show(addTitle.getText(), addName.getText()));  
 ObservableList<Show> temp = FXCollections.*observableArrayList*();  
 temp.addAll(main.getList());  
 ShowTableView.setItems(temp);  
 addTitle.clear();  
 addName.clear();  
  
 }  
 }  
 @FXML  
 private void handleAbout() {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setTitle("NewApp");  
 alert.setHeaderText("About: version 1.1.0");  
 alert.setContentText("Author: Yura Ruban\n");  
 alert.showAndWait();  
 }  
 @FXML  
 private void handleAddDay() {  
 private void handleAddDay() {  
 if(addVisitors.getText().length() >= 1 && addComment.getText().length() >= 1&&  
 Integer.*parseInt*(addVisitors.getText())>0) {  
 Show selected = ShowTableView.getSelectionModel().selectedItemProperty().get();  
 selected.add(new Day(Integer.*parseInt*(addVisitors.getText()),  
 String.*valueOf*(addComment.getText())));  
 ObservableList<Day> temp = FXCollections.*observableArrayList*();  
 temp.addAll(selected.getDays());  
 DayTableView.setItems(FXCollections.*observableArrayList*(temp));  
 addVisitors.clear();  
 addComment.clear();  
 showDetails(selected);  
  
 }  
  
}

@FXML  
 private void handleOpen() {  
 FileChooser fileChooser = new FileChooser();  
 FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter(  
 "XML files (\*.xml)", "\*.xml");  
 fileChooser.getExtensionFilters().add(extFilter);  
  
 File file = fileChooser.showOpenDialog(main.getStage());  
  
 if (file != null) {  
 main.loadDataFromFile(file);  
 ObservableList<Show> temp = FXCollections.*observableArrayList*();  
 temp.addAll(main.getList());  
 ShowTableView.setItems(temp);  
 }  
  
 }  
  
 @FXML  
 public void handleDelete(){  
 Show show = ShowTableView.getSelectionModel().selectedItemProperty().get();  
 Day day = DayTableView.getSelectionModel().selectedItemProperty().get();  
 if(day == null){  
 main.getList().remove(show);  
 ObservableList<Show> temp = FXCollections.*observableArrayList*();  
 temp.addAll(main.getList());  
 DayTableView.setItems(null);  
 ShowTableView.setItems(temp);  
 }else{  
 show.getDays().remove(day);  
 ObservableList<Day> temp = FXCollections.*observableArrayList*();  
 temp.addAll(show.getDays());  
 DayTableView.setItems(temp);  
 }  
 }  
  
 @FXML  
 public void handleVisitorSort()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 Show temp = ShowTableView.getItems().get(index);  
 temp.sortViaVisitors();  
 DayTableView.setItems(FXCollections.*observableArrayList*(temp.getDays()));  
 }  
 @FXML  
 public void handleABCSort()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 Show temp = ShowTableView.getItems().get(index);  
 temp.sortViaABC();  
 DayTableView.setItems(FXCollections.*observableArrayList*(temp.getDays()));  
 }  
 @FXML  
 public void handleShowSumCount()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 Show temp = ShowTableView.getItems().get(index);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("visitors");  
 alert.setTitle("visitors");  
 alert.setContentText(temp.visitors().toString());  
 alert.show();  
 }  
 @FXML  
 public void handleMaxVisitors()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 Show temp = ShowTableView.getItems().get(index);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("Max Visitors");  
 alert.setTitle("Max Visitors");  
 alert.setContentText(temp.maxVisitors().toString());  
 alert.show();  
 }  
 @FXML  
 public void handleLargeComment()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 Show temp = ShowTableView.getItems().get(index);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("Large Comment");  
 alert.setTitle("Large Comment");  
 alert.setContentText(temp.dayWithLargeComment());  
 alert.show();  
 }  
}

Клас Day:

package sample;  
  
  
import javafx.beans.property.IntegerProperty;  
import javafx.beans.property.SimpleIntegerProperty;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.beans.property.StringProperty;  
import javax.xml.bind.annotation.XmlElement;  
  
public class Day {  
 private IntegerProperty countOfvisitors;  
 private StringProperty comment;  
  
 public IntegerProperty passengersProperty() {  
 return countOfvisitors;  
 }  
 public StringProperty commentProperty() {  
 return comment;  
 }  
 @XmlElement(name = "comment")  
 public String getComment() {  
 return comment.get();  
 }  
 @XmlElement(name = "countOfvisitors")  
 public int getCountOfvisitors() {  
 return countOfvisitors.get();  
 }  
 public void setComment(String comment) {  
 this.comment.set(comment);  
 }  
 public void setCountOfvisitors(int passengers) {  
 this.countOfvisitors.setValue(passengers);  
 }  
  
 public Day() {  
 this(0, null);  
 }  
  
 public Day(int countOfvisitors) {  
 this.countOfvisitors.set(countOfvisitors);  
 this.comment.set(null);  
 }  
  
 @Override  
 public String toString() {  
 return "\ncountOfvisitors = " + countOfvisitors.get()  
 + " Comment = "+ comment.get();  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 Day s = (Day) obj;  
 return this.countOfvisitors == s.countOfvisitors;  
 }  
  
 public Day(int countOfvisitors, String comment) {  
 this.countOfvisitors = new SimpleIntegerProperty(countOfvisitors);  
 this.comment = new SimpleStringProperty(comment);  
 }  
}

Клас Main:

package sample;  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Alert;  
import javafx.scene.control.Button;  
import javafx.scene.control.Label;  
import javafx.scene.layout.AnchorPane;  
import javafx.scene.layout.VBox;  
import javafx.stage.FileChooser;  
import javafx.stage.Modality;  
import javafx.stage.Stage;  
import javax.xml.bind.JAXBContext;  
import javax.xml.bind.Marshaller;  
import javax.xml.bind.Unmarshaller;  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Main extends Application {  
 private ArrayList<Show> list = new ArrayList<>();  
 private Stage primaryStage;  
 private AnchorPane root;  
 public ArrayList<Show> getList() {  
 return list;  
 }  
  
 public Stage getStage() {  
 return primaryStage;  
 }  
  
 public void setList(ArrayList<Show> list) {  
 this.list = list;  
 }  
  
 public Main(){  
  
 FileChooser fileChooser = new FileChooser();  
 fileChooser.setTitle("Open Document");  
 FileChooser.ExtensionFilter extFilter =  
 new FileChooser.ExtensionFilter("XML files (\*.xml)", "\*.xml");  
 fileChooser.getExtensionFilters().add(extFilter);  
 File file = fileChooser.showOpenDialog(primaryStage);  
 if (file != null) {  
 loadDataFromFile(file);  
 }  
 }  
 @Override  
 public void start(Stage primaryStage) throws Exception{  
 this.primaryStage = primaryStage;  
 primaryStage.setOnCloseRequest(e -> {  
 if(!display("Exit", "Do you want to exit?")) e.consume();  
 });  
 this.primaryStage.setTitle("Shows");  
 initRootLayout();  
  
 }  
  
 public void initRootLayout() {  
 try {  
 // Загружаем корневой макет из fxml файла.  
 FXMLLoader loader = new FXMLLoader();  
 loader.setLocation(Main.class  
 .getResource("sample.fxml"));  
 root = (AnchorPane) loader.load();  
  
 // Отображаем сцену, содержащую корневой макет.  
 Scene scene = new Scene(root);  
 primaryStage.setScene(scene);  
  
 // Даём контроллеру доступ к главному прилодению.  
 Controller controller = loader.getController();  
 controller.setMain(this);  
  
 primaryStage.show();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void loadDataFromFile(File file) {  
 if (file != null) {  
 try {  
 JAXBContext context = JAXBContext  
 .*newInstance*(Shows.class);  
 Unmarshaller um = context.createUnmarshaller();  
 Shows shows = (Shows) um.unmarshal(file);  
  
 getList().clear();  
 getList().addAll(shows.getList());  
  
 } catch (Exception e) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Error");  
 alert.setHeaderText("Could not load data");  
 alert.setContentText("Could not load data from file:\n" + file.getPath());  
  
 alert.showAndWait();  
 }  
 }  
 }  
  
 public void saveToXML(Stage stage,List<Show>list) {  
 FileChooser fileChooser = new FileChooser();  
 fileChooser.setTitle("Save Document");  
 FileChooser.ExtensionFilter extFilter =  
 new FileChooser.ExtensionFilter("XML files (\*.xml)", "\*.xml");  
 fileChooser.getExtensionFilters().add(extFilter);  
 File file = fileChooser.showSaveDialog(stage);  
 if (file != null) {  
 try {  
 JAXBContext context = JAXBContext  
 .*newInstance*(Shows.class);  
 Marshaller marshaller = context.createMarshaller();  
 marshaller.setProperty(Marshaller.*JAXB\_FORMATTED\_OUTPUT*, true);  
  
 Shows shows = new Shows();  
 shows.setList(list);  
  
 marshaller.marshal(shows, file);  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Error");  
 alert.setHeaderText("Could not save data");  
 alert.setContentText("Could not save data to file:\n" + file.getPath());  
  
 alert.showAndWait();  
 }  
 }  
 }  
  
 boolean answer;  
  
 public boolean display(String title, String message) {  
 Stage window = new Stage();  
 window.initModality(Modality.*APPLICATION\_MODAL*);  
 window.setTitle(title);  
 window.setMinWidth(250);  
 Label label = new Label();  
 label.setText(message);  
  
 Button yesButton = new Button("Yes");  
 Button noButton = new Button("No");  
  
 yesButton.setOnAction(e -> {  
 answer = true;  
 window.close();  
 });  
 noButton.setOnAction(e -> {  
 answer = false;  
 window.close();  
 });  
  
 VBox layout = new VBox(10);  
 layout.getChildren().addAll(label, yesButton, noButton);  
 layout.setAlignment(Pos.*CENTER*);  
 Scene scene = new Scene(layout);  
 window.setScene(scene);  
 window.showAndWait();  
 return answer;  
 }  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}

Клас Show:

package sample;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.beans.property.StringProperty;  
import javax.xml.bind.annotation.XmlElement;  
import java.util.ArrayList;  
import java.util.Collections;  
  
  
public class Show {  
 private StringProperty title;  
 private StringProperty name;  
  
 private ArrayList<Day> days = new ArrayList<>();  
 public void sortViaVisitors()  
 {  
 Collections.*sort*(days,(Day o1, Day o2)->(Integer.*compare*(o2.getCountOfvisitors(),o1.getCountOfvisitors())));  
 }  
 public void sortViaABC()  
 {  
 Collections.*sort*(days,(Day o1,Day o2)->o1.getComment().compareTo(o2.getComment()));  
 }  
 @Override  
 public String toString() {  
 return "Show{" +  
 "title='" + title.get() + '\'' +  
 ", name='" + name.get() + '\'' + days.toString() + '}';  
 }  
  
 @XmlElement(name = "name")  
 public String getName() {  
 return name.get();  
 }  
 @XmlElement(name = "title")  
 public String getTitle() {  
 return title.get();  
 }  
 @XmlElement(name = "days")  
 public ArrayList<Day> getDays() {  
 return days;  
 }  
  
 public StringProperty getPropertyName() {  
 return name;  
 }  
 public StringProperty getPropertyTitle() {  
 return title;  
 }  
 public void setTitle(String title) {  
 this.title.set(title);  
 }  
  
 public void setName(String name) {  
 this.name.set(name);  
 }  
  
 public void setDays(ArrayList<Day> days) {  
 this.days = days;  
 }  
  
 public void add(Day ws) {  
 boolean exist = false;  
 for (Day s:  
 days) {  
 if(s.getCountOfvisitors() == ws.getCountOfvisitors()) {  
 exist = true;  
 break;  
 }  
 }  
 if(!exist) days.add(ws);  
 }  
 public Integer visitors()  
 {  
 Integer count = 0;  
 for(Day d : days)  
 {  
 count+=d.getCountOfvisitors();  
 }  
 return count;  
 }  
 public Integer maxVisitors()  
 {  
 int max = 0;  
 for(Day d : days)  
 {  
 if(max<d.getCountOfvisitors())  
 {  
 max = d.getCountOfvisitors();  
 }  
 }  
 return max;  
 }  
 public String dayWithLargeComment()  
 {  
 int max = 0;  
 int index=0;  
 for(int i =0;i<days.size();i++)  
 {  
 if(max<days.get(i).getComment().length())  
 {  
 max = days.get(i).getComment().length();  
 index = i;  
 }  
 }  
 return days.get(index).getComment();  
 }  
 public String search(Day ws) {  
 for (Day day: days) {  
 if(day.equals(ws)){  
 return ws.toString();  
 }  
 }  
 System.*out*.println("This CountOfvisitors ("+ ws.getCountOfvisitors() +") not exist!");  
 return null;  
 }  
  
 public int total() {  
 return days.size();  
 }  
 public Day getCountOfvisitors(int i) {  
 return days.get(i);  
 }  
 public Show() {  
 this(null, null);  
 }  
  
 public Show(String title, String name) {  
 this.title = new SimpleStringProperty(title);  
 this.name = new SimpleStringProperty(name);  
 }  
}

Клас Shows:

package sample;  
  
  
import javax.xml.bind.annotation.XmlElement;  
import javax.xml.bind.annotation.XmlRootElement;  
import java.util.List;  
  
@XmlRootElement(name="Shows")  
public class Shows {  
 private List<Show> shows;  
  
 public Shows() { }  
  
 @XmlElement(name = "Show")  
 public List<Show> getList() {  
 return shows;  
 }  
  
 @Override  
 public String toString() {  
 String res = "";  
 for (Show ts:  
 shows) {  
 res+= "\nTitle: " + ts.getTitle()  
 + "\nName: " + ts.getName() + "\n"+ ts.getDays();  
 }  
 return res;  
 }  
  
 public void setList(List<Show> ws) {  
 this.shows = ws;  
 }  
}

Файл sample.fxml

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.ButtonBar?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.Menu?>  
<?import javafx.scene.control.MenuBar?>  
<?import javafx.scene.control.MenuItem?>  
<?import javafx.scene.control.SplitPane?>  
<?import javafx.scene.control.TableColumn?>  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/8.0.112" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller">  
 <children>  
 <SplitPane dividerPositions="0.4816053511705686" layoutX="133.0" layoutY="24.0" prefHeight="376.0" prefWidth="600.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="24.0">  
 <items>  
 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="398.0" prefWidth="249.0">  
 <children>  
 <TableView fx:id="ShowTableView" editable="true" layoutX="8.0" layoutY="40.0" prefHeight="308.0" prefWidth="285.0" AnchorPane.bottomAnchor="66.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <columns>  
 <TableColumn fx:id="titleSH" prefWidth="140.0" text="Title" />  
 <TableColumn fx:id="nameSH" minWidth="0.0" prefWidth="144.0" text="Name" />  
 </columns>  
 </TableView>  
 <TextField fx:id="addTitle" layoutX="64.0" layoutY="308.0" prefHeight="25.0" prefWidth="137.0" />  
 <Label layoutX="14.0" layoutY="312.0" text="Title" />  
 <Label layoutX="14.0" layoutY="340.0" text="Name" />  
 <TextField fx:id="addName" layoutX="64.0" layoutY="336.0" prefHeight="25.0" prefWidth="137.0" />  
 <Button layoutX="221.0" layoutY="308.0" mnemonicParsing="false" onAction="#handleAddShow" prefHeight="49.0" prefWidth="38.0" text="Add" />  
 </children>  
 </AnchorPane>  
 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="160.0" prefWidth="100.0">  
 <children>  
 <ButtonBar buttonMinWidth="55.0" layoutX="145.0" layoutY="334.0" AnchorPane.leftAnchor="110.0" AnchorPane.rightAnchor="60.0" AnchorPane.topAnchor="334.0" />  
 <TableView fx:id="DayTableView" editable="true" layoutX="-1.0" layoutY="31.0" prefHeight="308.0" prefWidth="307.0" AnchorPane.bottomAnchor="66.0" AnchorPane.leftAnchor="-1.0" AnchorPane.rightAnchor="1.0" AnchorPane.topAnchor="0.0">  
 <columns>  
 <TableColumn fx:id="visitorsNum" prefWidth="92.0" text="Visitors" />  
 <TableColumn fx:id="commentN" prefWidth="214.0" text="Comment" />  
 </columns>  
 </TableView>  
 <Label layoutX="14.0" layoutY="317.0" text="Visitors" />  
 <TextField fx:id="addVisitors" layoutX="72.0" layoutY="313.0" prefHeight="25.0" prefWidth="80.0" />  
 <Button layoutX="160.0" layoutY="313.0" mnemonicParsing="false" onAction="#handleAddDay" text="Add" />  
 <Button layoutX="209.0" layoutY="313.0" mnemonicParsing="false" onAction="#handleDelete" text="Delete" />  
 <Label layoutX="13.0" layoutY="348.0" prefHeight="17.0" prefWidth="80.0" text="Comment" />  
 <TextField fx:id="addComment" layoutX="72.0" layoutY="344.0" prefHeight="25.0" prefWidth="188.0" />  
 </children>  
 </AnchorPane>  
 </items>  
 </SplitPane>  
 <MenuBar layoutY="2.0" AnchorPane.bottomAnchor="375.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <menus>  
 <Menu mnemonicParsing="false" text="File">  
 <items>  
 <MenuItem mnemonicParsing="false" onAction="#handleOpen" text="Open" />  
 <MenuItem mnemonicParsing="false" onAction="#handleSaveAs" text="Save" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Help">  
 <items>  
 <MenuItem mnemonicParsing="false" onAction="#handleAbout" text="About" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Sort &amp; Search">  
 <items>  
 <MenuItem mnemonicParsing="false" onAction="#handleVisitorSort" text="SortViaVisitors" />  
 <MenuItem mnemonicParsing="false" onAction="#handleABCSort" text="SortViaABC" />  
 <MenuItem mnemonicParsing="false" onAction="#handleShowSumCount" text="Show count of visitors" />  
 <MenuItem mnemonicParsing="false" onAction="#handleMaxVisitors" text="Show max visitors" />  
 <MenuItem mnemonicParsing="false" onAction="#handleLargeComment" text="largest comment" />  
 </items>  
 </Menu>  
 </menus>  
 </MenuBar>  
 </children>  
</AnchorPane>

Результати виконання показані на рисунках 1, 2

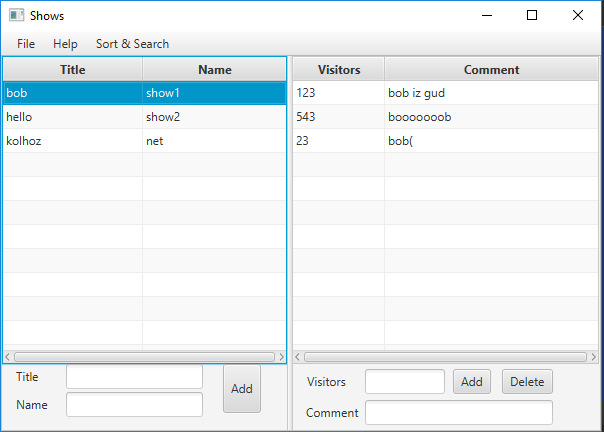


Рис.2.1 – Головне меню

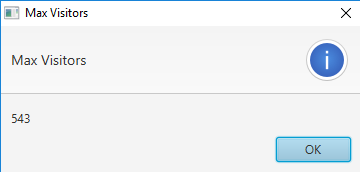


Рис.2.2 – Вікно, яке показує максимальне число відвідувачів

**Завдання 2.** Створено перелік "Місяць". Додано методи отримання попереднього та наступного місяця, а також функцію, яка повертає сезон для кожного місяця. Передбачено виведення місяців українською (російською) мовою. Створено статичну функцію виведення даних про усі місяці. Протестувано перелік в функції main() тестового класу.

Код програми:

Клас Main:

enum MonthOfYear {  
 *JANUARY*(31), *FEBRUARY*(28), *MARCH*(31), *APRIL*(30), *MAY*(31), *JUNE*(30), *JULY*(31), *AUGUST*(31), *SEPTEMBER*(30), *OCTOBER*(31), *NOVEMBER*(30), *DECEMBER*(31);  
 private int countOfDays;  
 MonthOfYear(int countOfDays)  
 {  
 this.countOfDays=countOfDays;  
 }  
 @Override  
 public String toString() {  
 return name()+ " кол-во дней в месяце " + countOfDays;  
 }  
 MonthOfYear next() {  
 MonthOfYear month = *values*()[(ordinal() + 1) % *values*().length];  
 return month;  
 }  
 MonthOfYear prev() {  
 MonthOfYear month = *values*()[(ordinal()-1) % *values*().length];  
 return month;  
 }  
 String season()  
 {  
 switch (this){  
 case *JANUARY*:  
 return "Winter";  
 case *FEBRUARY*:  
 return "Winter";  
 case *MARCH*:  
 return "Spring";  
 case *APRIL*:  
 return "Spring";  
 case *MAY*:  
 return "Spring";  
 case *JUNE*:  
 return "Summer";  
 case *JULY*:  
 return "Summer";  
 case *AUGUST*:  
 return "Summer";  
 case *SEPTEMBER*:  
 return "Autumn";  
 case *OCTOBER*:  
 return "Autumn";  
 case *NOVEMBER*:  
 return "Autumn";  
 case *DECEMBER*:  
 return "Winter";  
 default:  
 return "Error";  
 }  
 }  
 String rus()  
 {  
 switch (this){  
 case *JANUARY*:  
 return "Январь";  
 case *FEBRUARY*:  
 return "Февраль";  
 case *MARCH*:  
 return "Март";  
 case *APRIL*:  
 return "Апрель";  
 case *MAY*:  
 return "Май";  
 case *JUNE*:  
 return "Июнь";  
 case *JULY*:  
 return "Июль";  
 case *AUGUST*:  
 return "Август";  
 case *SEPTEMBER*:  
 return "Сентябрь";  
 case *OCTOBER*:  
 return "Октябрь";  
 case *NOVEMBER*:  
 return "Ноябрь";  
 case *DECEMBER*:  
 return "Декабрь";  
 default:  
 return "Error";  
 }  
 }  
}  
public class Main {  
  
 public static void main(String[] args) {  
 MonthOfYear m = MonthOfYear.*DECEMBER*;  
 System.*out*.println(m.next());  
 for (int i = 0; i < 12; i++) {  
 System.*out*.println("Месяц : "+ MonthOfYear.*values*()[i] + " сезон: " +MonthOfYear.*values*()[i].season()+ " рус: " + MonthOfYear.*values*()[i].rus());  
 }  
 }  
}

Результат виконання зображений на рисунку 3

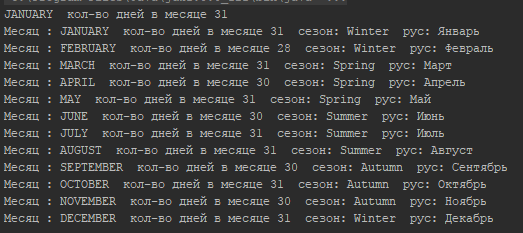


Рис. 2.3 – Результат програми 2

### Завдання 3: Представлено дані про користувачів у вигляді асоціативного масиву (ім'я / пароль) з припущенням, що всі імена користувачів різні. Виведено дані про користувачів з довжиною пароля більше 6 символів.

Код програми:

import java.util.\*;  
  
public class Main {  
 public static void main(String[] args) {  
 Map<String,String> user=new TreeMap<>();  
 user.put("8sumbols","First user");  
 user.put("7sumbol","Second user");  
 user.put("6sumbo","Third user");  
 user.put("5sumb","Fourth user");  
 for(Map.Entry<String,String>m : user.entrySet())  
 {  
 if(m.getKey().length()>6)  
 System.*out*.println(m.getKey() + " " + m.getValue());  
  
 }  
 }  
}

Результат виконання на рисунку 4.



Рис. 2.4 – Результат завдання 3

**Завдання 4:** Створено застосунок графічного інтерфейсу користувача, в якому після введення чисел у двох рядках типу TextField виконується одна з чотирьох арифметичних дій (залежно від вибраної кнопки RadioButton). Результат виводиться в інше текстове поле.

Код програми:

Клас Main:

package sample;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
public class Main extends Application {  
 @Override  
 public void start(Stage primaryStage) throws Exception{  
 Parent root = FXMLLoader.*load*(getClass().getResource("sample.fxml"));  
 primaryStage.setTitle("Calculator");  
 primaryStage.setScene(new Scene(root));  
 primaryStage.show();  
 }  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}

Клас Controller:

package sample;  
import javafx.event.ActionEvent;  
import javafx.fxml.Initializable;  
import javafx.scene.control.Alert;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.TextField;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
  
public class Controller implements Initializable{  
 public RadioButton plus;  
 public RadioButton minus;  
 public RadioButton div;  
 public RadioButton mul;  
 public TextField field1, field2, field3;  
 public void dd(ActionEvent event)  
 {  
 plus.setOnAction(this::buttonClick);  
 minus.setOnAction(this::buttonClick1);  
 mul.setOnAction(this::buttonClick2);  
 div.setOnAction(this::buttonClick3);  
 }  
 private void buttonClick(ActionEvent actionEvent) {  
 try {  
 double i = Double.*parseDouble*(field1.getText());  
 double j = Double.*parseDouble*(field2.getText());  
 double k = i + j;  
 field3.setText(k + "");  
 }  
 catch (NumberFormatException e1) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Помилка");  
 alert.setHeaderText("Хибні дані!");  
 alert.showAndWait();  
 }  
 }  
 private void buttonClick1(ActionEvent actionEvent) {  
 try {  
 double i = Double.*parseDouble*(field1.getText());  
 double j = Double.*parseDouble*(field2.getText());  
 double k = i - j;  
 field3.setText(k + "");  
 }  
 catch (NumberFormatException e1) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Помилка");  
 alert.setHeaderText("Хибні дані!");  
 alert.showAndWait();  
 }  
 }  
 private void buttonClick2(ActionEvent actionEvent) {  
 try {  
 double a = Double.*parseDouble*(field1.getText());  
 double b = Double.*parseDouble*(field2.getText());  
 double k = a \* b;  
 field3.setText(k + "");  
 }  
 catch (NumberFormatException e1) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Помилка");  
 alert.setHeaderText("Хибні дані!");  
 alert.showAndWait();  
 }  
 }  
 private void buttonClick3(ActionEvent actionEvent) {  
 try {  
 double i = Double.*parseDouble*(field1.getText());  
 double j = Double.*parseDouble*(field2.getText());  
 double k = i / j;  
 field3.setText(k + "");  
 }  
 catch (NumberFormatException e1) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Помилка");  
 alert.setHeaderText("Хибні дані!");  
 alert.showAndWait();  
 }  
 }  
  
 @Override  
 public void initialize(URL location, ResourceBundle resources) {  
 }  
}

Файл sample.fxml

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.RadioButton?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.ColumnConstraints?>  
<?import javafx.scene.layout.GridPane?>  
<?import javafx.scene.layout.RowConstraints?>  
  
<GridPane alignment="center" hgap="10" vgap="10" xmlns="http://javafx.com/javafx/8.0.121" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller">  
 <columnConstraints>  
 <ColumnConstraints />  
 <ColumnConstraints />  
 <ColumnConstraints />  
 </columnConstraints>  
 <rowConstraints>  
 <RowConstraints />  
 <RowConstraints />  
 </rowConstraints>  
 <children>  
 <AnchorPane prefHeight="400.0" prefWidth="400.0" GridPane.columnIndex="1">  
 <children>  
 <GridPane layoutX="20.0" layoutY="156.0" prefHeight="49.0" prefWidth="360.0">  
 <columnConstraints>  
 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />  
 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />  
 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />  
 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0" prefWidth="100.0" />  
 </columnConstraints>  
 <rowConstraints>  
 <RowConstraints minHeight="10.0" prefHeight="30.0" vgrow="SOMETIMES" />  
 </rowConstraints>  
 <children>  
 <RadioButton fx:id="plus" mnemonicParsing="false" onAction="#dd" text="+" />  
 <RadioButton fx:id="minus" mnemonicParsing="false" onAction="#dd" text="-" GridPane.columnIndex="1" />  
 <RadioButton fx:id="div" mnemonicParsing="false" onAction="#dd" text="/" GridPane.columnIndex="2" />  
 <RadioButton fx:id="mul" mnemonicParsing="false" onAction="#dd" text="\*" GridPane.columnIndex="3" />  
 </children>  
 </GridPane>  
 <TextField fx:id="field2" layoutX="196.0" layoutY="103.0" onAction="#dd" prefHeight="26.0" prefWidth="142.0" />  
 <TextField fx:id="field1" layoutX="20.0" layoutY="103.0" onAction="#dd" />  
 <Label layoutX="114.0" layoutY="270.0" prefHeight="26.0" prefWidth="68.0" text="Результат" />  
 <TextField fx:id="field3" layoutX="200.0" layoutY="270.0" onAction="#dd" prefHeight="26.0" prefWidth="142.0" />  
 </children>  
 </AnchorPane>  
 </children>  
</GridPane>

Результат виконання на рисунку 5

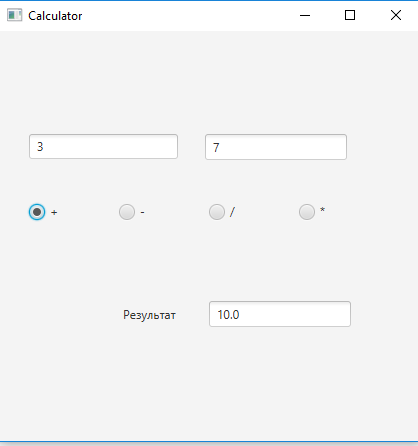


Рис. 2.5 – Програма 4.

**Завдання 5:** Розроблено програму графічного інтерфейсу користувача перегляду слів невеличкого англо-українського (англо-російського) словника. Реалізувано функції пошуку слова, додавання нових слів.

Код програми:

Клас Main:

package sample;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
public class Main extends Application {  
  
 @Override  
 public void start(Stage primaryStage) throws Exception{  
 Parent root = FXMLLoader.*load*(getClass().getResource("sample.fxml"));  
 primaryStage.setTitle("Word");  
 primaryStage.setScene(new Scene(root));  
 primaryStage.show();  
 }  
  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}

Клас Word:

package sample;  
  
import javafx.beans.property.SimpleStringProperty;  
  
public class Word {  
 private SimpleStringProperty key;  
 private SimpleStringProperty value;  
 public Word(String key, String value)  
 {  
 this.key=new SimpleStringProperty(key);  
 this.value= new SimpleStringProperty(value);  
 }  
 public Word(){}  
  
 public String getKey() {  
 return key.get();  
 }  
  
 public void setValue(String value) {  
 this.value.set(value);  
 }  
  
 public String getValue() {  
 return value.get();  
 }  
  
 public void setKey(String key) {  
 this.key.set(key);  
 }  
}

Клас Controller:

package sample;  
  
import javafx.beans.property.SimpleStringProperty;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.TableColumn;  
import javafx.scene.control.TableView;  
import javafx.scene.control.TextField;  
import javafx.scene.control.cell.PropertyValueFactory;  
  
import java.awt.\*;  
import java.net.URL;  
import java.util.ResourceBundle;  
  
public class Controller implements Initializable{  
 @FXML private TableView<Word> tableView;  
 @FXML private TextField field1;  
 @FXML private TextField field2;  
 @FXML private TextField field3;  
 @FXML private Button addButton;  
 @FXML private TableColumn<Word, SimpleStringProperty> wordCol;  
 @FXML private TableColumn<Word, SimpleStringProperty> translateCol;  
 private ObservableList<Word> vocabulary;  
 @Override  
 public void initialize(URL location, ResourceBundle resources) {  
 wordCol.setCellValueFactory(new PropertyValueFactory<Word,SimpleStringProperty>("key"));  
 translateCol.setCellValueFactory(new PropertyValueFactory<Word,SimpleStringProperty>("value"));  
 tableView.setItems(getVocabulary());  
 }  
  
 public ObservableList<Word> getVocabulary() {  
 vocabulary = FXCollections.*observableArrayList*();  
 vocabulary.add(new Word("слово","word"));  
 vocabulary.add(new Word("привет","hello"));  
 vocabulary.add(new Word("имя","name"));  
 return vocabulary;  
 }  
  
 public void newWord(){  
 Word newWord= new Word(field2.getText(),field3.getText());  
 tableView.getItems().add(newWord);  
 }  
  
 public void search()  
 {  
 if(!field1.getText().isEmpty())  
 {  
 for(int i=0;i<vocabulary.size();i++)  
 {  
 if(vocabulary.get(i).getKey().equals(field1.getText())){  
 tableView.getSelectionModel().select(vocabulary.get(i));  
 break;  
 }  
 if(vocabulary.get(i).getValue().equals(field1.getText()))  
 {  
 tableView.getSelectionModel().select(vocabulary.get(i));  
 break;  
 }  
 }  
 }  
 }  
}

Файл sample.fxml

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.SplitPane?>  
<?import javafx.scene.control.TableColumn?>  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.ColumnConstraints?>  
<?import javafx.scene.layout.GridPane?>  
<?import javafx.scene.layout.RowConstraints?>  
<?import javafx.scene.text.Font?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/8.0.112" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller">  
 <children>  
 <SplitPane dividerPositions="0.4066115702479339, 0.9950413223140496" layoutY="-5.0" prefHeight="411.0" prefWidth="607.0">  
 <items>  
 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="409.0" prefWidth="243.0">  
 <children>  
 <Label layoutX="65.0" layoutY="1.0" text="Словарь">  
 <font>  
 <Font name="System Italic" size="18.0" />  
 </font>  
 </Label>  
 <TableView fx:id="tableView" layoutX="6.0" layoutY="28.0" prefHeight="376.0" prefWidth="232.0">  
 <columns>  
 <TableColumn fx:id="wordCol" prefWidth="116.0" text="Word" />  
 <TableColumn fx:id="translateCol" prefWidth="115.0" text="Translate" />  
 </columns>  
 </TableView>  
 </children>  
 </AnchorPane>  
 <GridPane prefHeight="409.0" prefWidth="350.0">  
 <columnConstraints>  
 <ColumnConstraints hgrow="SOMETIMES" maxWidth="181.0" minWidth="10.0" prefWidth="45.0" />  
 <ColumnConstraints hgrow="SOMETIMES" maxWidth="213.0" minWidth="10.0" prefWidth="204.0" />  
 <ColumnConstraints hgrow="SOMETIMES" maxWidth="133.0" minWidth="7.0" prefWidth="37.0" />  
 <ColumnConstraints hgrow="SOMETIMES" maxWidth="228.0" minWidth="10.0" prefWidth="207.0" />  
 </columnConstraints>  
 <rowConstraints>  
 <RowConstraints maxHeight="49.0" minHeight="10.0" prefHeight="49.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="61.0" minHeight="10.0" prefHeight="43.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="60.0" minHeight="5.0" prefHeight="24.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="141.0" minHeight="10.0" prefHeight="80.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="92.0" minHeight="10.0" prefHeight="89.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="76.0" minHeight="10.0" prefHeight="31.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="140.0" minHeight="10.0" prefHeight="42.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="124.0" minHeight="10.0" prefHeight="38.0" vgrow="SOMETIMES" />  
 <RowConstraints maxHeight="89.0" minHeight="10.0" prefHeight="45.0" vgrow="SOMETIMES" />  
 </rowConstraints>  
 <children>  
 <TextField fx:id="field1" prefHeight="25.0" prefWidth="169.0" GridPane.columnIndex="1" GridPane.rowIndex="1" />  
 <TextField fx:id="field2" prefHeight="25.0" prefWidth="195.0" GridPane.columnIndex="1" GridPane.rowIndex="6" />  
 <TextField fx:id="field3" prefHeight="25.0" prefWidth="138.0" GridPane.columnIndex="3" GridPane.rowIndex="6" />  
 <Label text="Англ" GridPane.columnIndex="1" GridPane.rowIndex="5">  
 <font>  
 <Font name="System Italic" size="18.0" />  
 </font>  
 </Label>  
 <Label text="Рус" GridPane.columnIndex="3" GridPane.rowIndex="5">  
 <font>  
 <Font name="System Italic" size="18.0" />  
 </font>  
 </Label>  
 <Button mnemonicParsing="false" onAction="#newWord" prefHeight="25.0" prefWidth="120.0" text="add" GridPane.columnIndex="3" GridPane.rowIndex="7" />  
 <Button mnemonicParsing="false" onAction="#search" prefHeight="25.0" prefWidth="120.0" text="find" GridPane.columnIndex="3" GridPane.rowIndex="1" />  
 </children>  
 </GridPane>  
 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="409.0" prefWidth="381.0" />  
 </items>  
 </SplitPane>  
 </children>  
</AnchorPane>

Результати виконання на рисунку 6

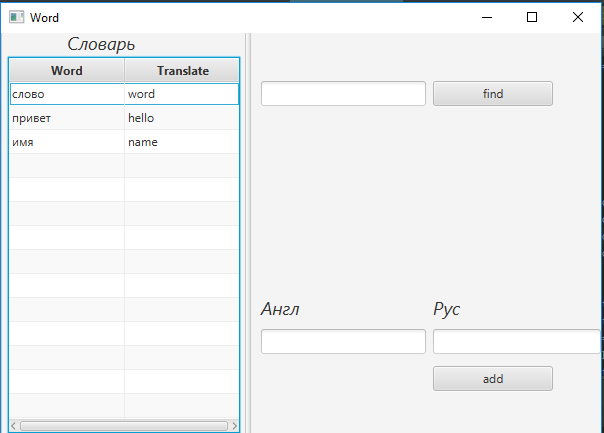


Рис. 2.6 – Програма 5.

**Висновки:** В даній лабораторній роботі були здобуті навички розробки графічного інтерфейсу користувача, а також був вивчений теоретичний матеріал з використання асоціативних масивів.