### 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 TextField виконується 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.

**Хід роботи**

### 1.1 Індивідуальне завдання

*Код програми 1.1:*

**Main.java**

package application;

import javafx.application.Application;

import javafx.stage.Stage;

import javafx.scene.Scene;

import javafx.scene.layout.BorderPane;

import javafx.fxml.FXMLLoader;

public class Main extends Application {

@Override

public void start(Stage primaryStage) {

try {

BorderPane root = (BorderPane)FXMLLoader.load(getClass().getResource("Sample.fxml"));

Scene scene = new Scene(root,700,500);

scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());

primaryStage.setScene(scene);

primaryStage.show();

} catch(Exception e) {

e.printStackTrace();

}

}

public static void main(String[] args) {

launch(args);

}

}

**Hour.java**

package seclab.f.ex;

import java.io.FileReader;

import java.io.FileWriter;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class DocArrTXT extends DocArr {

public DocArrTXT() {

}

public DocArrTXT(String name, String spec, Recept[] recepts) {

super(name, spec, recepts);

}

public void toTxt(String name) {

try (FileWriter fReader = new FileWriter(name)) {

fReader.write(this.getName() + "\n");

fReader.write(this.getSpec() + "\n");

for (Recept rec : getRecepts()) {

fReader.write(rec.toString() + "\n");

}

} catch (Exception e) {

// TODO: handle exception

}

}

public DocArrTXT fromTxt() {

DocArrTXT sArr = new DocArrTXT();

try (Scanner scanner = new Scanner(new FileReader("Doctor.txt"))) {

sArr.setName(scanner.nextLine());

sArr.setSpec(scanner.nextLine());

List<Recept> sList = new ArrayList<>();

while (scanner.hasNext()) {

String[] string = scanner.nextLine().split(",");

Integer integer = Integer.parseInt(string[2].replaceAll(" ", ""));

sList.add(new Recept(integer, string[0], string[1]));

}

sArr.setRecepts(sList.toArray(new Recept[sList.size()]));

} catch (Exception e) {

e.printStackTrace();

}

return sArr;

}

}package application;

import java.util.StringTokenizer;

public class Hour implements Comparable<Hour>{

private int customers;

private String comment;

public Hour(int customers, String comment){

this.customers = customers;

this.comment = comment;

}

public String getComment() {

return comment;

}

public void setComment(String comment) {

this.comment = comment;

}

public int getCustomers(){

return customers;

}

public void setCustomers(int customers){

this.customers = customers;

}

public boolean containsWord(String word) {

StringTokenizer st = new StringTokenizer(comment);

while(st.hasMoreTokens()){

if(st.nextToken().equalsIgnoreCase(word)){

return true;

}

}

return false;

}

private void testWord(String word){

if(containsWord(word)){

System.out.print("Comment of hour has word " + word + "\n");

}

else{

System.out.print("Comment of hour has not word " + word + "\n");

}

}

public int compareTo(Hour h) {

return Integer.compare(getCustomers(), h.getCustomers());

}

}

**SampleController.java**

package application;

import java.io.File;

import java.io.IOException;

import java.net.URL;

import java.util.ArrayList;

import java.util.List;

import java.util.ResourceBundle;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.Alert;

import javafx.scene.control.TableColumn.CellEditEvent;

import javafx.scene.control.Label;

import javafx.scene.control.TextField;

import javafx.scene.control.TextArea;

import javafx.scene.control.TableView;

import javafx.scene.control.TableColumn;

import javafx.scene.control.Alert.AlertType;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.control.cell.TextFieldTableCell;

import javafx.stage.FileChooser;

import javafx.util.converter.IntegerStringConverter;

import javafx.application.Platform;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

public class SampleController implements Initializable {

private XmlStore store = new XmlStore();

private ObservableList<Hour> observableList;

public static void showMessage(String message) {

Alert alert = new Alert(AlertType.INFORMATION);

alert.setTitle("");

alert.setHeaderText(message);

alert.showAndWait();

}

public static void showError(String message) {

Alert alert = new Alert(AlertType.ERROR);

alert.setTitle("Ошибка");

alert.setHeaderText(message);

alert.showAndWait();

}

public static FileChooser getFileChooser(String title) {

FileChooser fileChooser = new FileChooser();

fileChooser.setInitialDirectory(new File("."));

fileChooser.getExtensionFilters().add(

new FileChooser.ExtensionFilter("XML-файлы (\*.xml)", "\*.xml"));

fileChooser.getExtensionFilters().add(

new FileChooser.ExtensionFilter("Все файлы (\*.\*)", "\*.\*"));

fileChooser.setTitle(title);

return fileChooser;

}

@FXML TextField textFieldStore;

@FXML TextField textFieldAddress;

@FXML TextField textFieldText;

@FXML TextArea textAreaResults;

@FXML TableView<Hour> tableViewHours;

@FXML TableColumn<Hour, Integer> tableColumnCustomers;

@FXML TableColumn<Hour, String> tableColumnComments;

@Override

public void initialize(URL location, ResourceBundle resources) {

tableViewHours.setPlaceholder(new Label(""));

}

@FXML public void doNew(ActionEvent event) {

store = new XmlStore();

observableList = null;

textFieldStore.setText("");

textFieldAddress.setText("");

textFieldText.setText("");

textAreaResults.setText("");

tableViewHours.setItems(null);

tableViewHours.setPlaceholder(new Label(""));

}

@FXML public void doOpen(ActionEvent event) {

FileChooser fileChooser = getFileChooser("Открыть XML-файл");

File file;

if ((file = fileChooser.showOpenDialog(null)) != null) {

try {

store.readFromFile(file.getCanonicalPath());

textFieldStore.setText(store.getName());

textFieldAddress.setText(store.getAddress() + "");

textAreaResults.setText("");

tableViewHours.setItems(null);

updateTable();

}

catch (IOException e) {

showError("Файл не найден");

}

}

}

@FXML public void doSave(ActionEvent event) {

FileChooser fileChooser = getFileChooser("Сохрани XML-файл");

File file;

if ((file = fileChooser.showSaveDialog(null)) != null) {

try {

updateSourceData();

store.writeToFile(file.getCanonicalPath());

showMessage("Результаты успешно сохранены");

}

catch (Exception e) {

showError("Ошибка записи в файл");

}

}

}

@FXML public void doExit(ActionEvent event) {

Platform.exit();

}

@FXML public void doAdd(ActionEvent event) {

if (observableList == null) {

updateTable();

}

observableList.add(new Hour(0, ""));

}

@FXML public void doRemove(ActionEvent event) {

if (observableList == null) {

return;

}

if (observableList.size() > 0) {

observableList.remove(observableList.size() - 1);

}

if (observableList.size() <= 0) {

observableList = null;

}

}

@FXML public void doSortByCustomers(ActionEvent event) {

updateSourceData();

store.sortByCustomers();

updateTable();

}

@FXML public void doSortByComments(ActionEvent event) {

updateSourceData();

store.sortByComments();

updateTable();

}

@FXML public void doAbout(ActionEvent event) {

Alert alert = new Alert(AlertType.INFORMATION);

alert.setTitle("О программе...");

alert.setHeaderText("Данные про посетителей");

alert.setContentText("Версия 1.0");

alert.showAndWait();

}

@FXML public void nameChanged(ActionEvent event) {

store.setName(textFieldStore.getText());

}

@FXML public void addressChanged(ActionEvent event) {

try {

String address = textFieldAddress.getText();

store.setAddress(address);

}

catch (NumberFormatException e) {

textFieldAddress.setText(store.getAddress() + "");

}

}

@FXML public void doSearchByWord(ActionEvent event) {

updateSourceData();

textAreaResults.setText("");

for (int i = 0; i < store.hoursCount(); i++) {

Hour c = store.getHour(i);

if (c.containsWord(textFieldText.getText())) {

showResults(c);

}

}

}

private void showResults(Hour hour) {

textAreaResults.appendText("Посетителей " + hour.getCustomers() + "\n");

textAreaResults.appendText("Комvентарий: " + hour.getComment() + "\n");

textAreaResults.appendText("\n");

}

private void updateSourceData() {

store.clearHours();

for (Hour c : observableList) {

store.addHour(c);

}

}

private void updateCustomers(CellEditEvent<Hour, Integer> t) {

Hour c = (Hour) t.getTableView().getItems().get(t.getTablePosition().getRow());

if (t.getNewValue() < 0){

c.setCustomers(0);

} else {

c.setCustomers(t.getNewValue());

}

}

private void updateComments(CellEditEvent<Hour, String> t) {

Hour c = (Hour) t.getTableView().getItems().get(t.getTablePosition().getRow());

c.setComment(t.getNewValue());

}

private void updateTable() {

List<Hour> list = new ArrayList<Hour>();

observableList = FXCollections.observableList(list);

for (int i = 0; i < store.hoursCount(); i++) {

list.add(store.getHour(i));

}

tableViewHours.setItems(observableList);

tableColumnCustomers.setCellValueFactory(new PropertyValueFactory<>("customers"));

tableColumnCustomers.setCellFactory(

TextFieldTableCell.<Hour, Integer>forTableColumn(new IntegerStringConverter()));

tableColumnCustomers.setOnEditCommit(t -> updateCustomers(t));

tableColumnComments.setCellValueFactory(new PropertyValueFactory<>("comment"));

tableColumnComments.setCellFactory(TextFieldTableCell.forTableColumn());

tableColumnComments.setOnEditCommit(t -> updateComments(t));

}

}

**XMLStore.java**

package application;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Arrays;

import java.util.Collections;

import javax.xml.bind.JAXBContext;

import javax.xml.bind.JAXBException;

import javax.xml.bind.Marshaller;

import javax.xml.bind.Unmarshaller;

import xml.StoreData;

import xml.StoreData.HourData;

public class XmlStore extends AbstractStore{

private StoreData sd;

public XmlStore(){

sd = new StoreData();

}

@Override

public String getName() {

return sd.getName();

}

@Override

public void setName(String name) {

sd.setName(name);

}

@Override

public String getAddress() {

return sd.getAddress();

}

@Override

public void setAddress(String address) {

sd.setAddress(address);

}

@Override

public Hour getHour(int i) {

HourData hourData = sd.getHourData().get(i);

return new Hour(hourData.getCustomers(), hourData.getComment());

}

@Override

public void setHour(int i, Hour hour) {

HourData hourData = new HourData();

hourData.setCustomers(hour.getCustomers());

hourData.setComment(hour.getComment());

sd.getHourData().set(i, hourData);

}

@Override

public void addHour(Hour hour) {

HourData hourData = new HourData();

hourData.setCustomers(hour.getCustomers());

hourData.setComment(hour.getComment());

sd.getHourData().add(hourData);

}

@Override

public int hoursCount() {

return sd.getHourData().size();

}

@Override

public void clearHours() {

sd.getHourData().clear();

}

@Override

public void sortByCustomers() {

Hour[] arr = new Hour[hoursCount()];

for (int i = 0; i < arr.length; i++) {

arr[i] = getHour(i);

}

Arrays.sort(arr, Collections.reverseOrder());

for (int i = 0; i < arr.length; i++) {

setHour(i, arr[i]);

}

}

@Override

public void sortByComments() {

Hour[] arr = new Hour[hoursCount()];

for (int i = 0; i < arr.length; i++) {

arr[i] = getHour(i);

}

Arrays.sort(arr, new CompareByComments());

for (int i = 0; i < arr.length; i++) {

setHour(i, arr[i]);

}

}

@Override

public boolean readFromFile(String fileName) {

try {

JAXBContext jaxbContext = JAXBContext.newInstance("xml");

Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();

sd = (StoreData) unmarshaller.unmarshal(new FileInputStream(fileName));

return true;

}

catch (FileNotFoundException | JAXBException e) {

return false;

}

}

@Override

public boolean writeToFile(String fileName) {

try {

JAXBContext jaxbContext = JAXBContext.newInstance("xml");

Marshaller marshaller = jaxbContext.createMarshaller();

marshaller.setProperty(Marshaller.JAXB\_FORMATTED\_OUTPUT, Boolean.TRUE);

marshaller.marshal(sd, new FileWriter(fileName));

return true;

}

catch (JAXBException | IOException e) {

return false;

}

}

}

**Sample.fxml**

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.\*?>

<?import java.lang.\*?>

<?import javafx.scene.layout.\*?>

<?import javafx.scene.layout.BorderPane?>

<BorderPane prefHeight="500.0" prefWidth="700.0"

xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1"

fx:controller="application.SampleController">

<top>

<MenuBar BorderPane.alignment="CENTER">

<menus>

<Menu mnemonicParsing="false" text="Файл">

<items>

<MenuItem mnemonicParsing="false" text="Новый" onAction="#doNew"/>

<MenuItem mnemonicParsing="false" text="Открыть..." onAction="#doOpen"/>

<MenuItem mnemonicParsing="false" text="Сохранить..." onAction="#doSave"/>

<MenuItem mnemonicParsing="false" text="Выход" onAction="#doExit"/>

</items>

</Menu>

<Menu mnemonicParsing="false" text="Правка">

<items>

<MenuItem mnemonicParsing="false" text="Добавить строку" onAction="#doAdd"/>

<MenuItem mnemonicParsing="false" text="Удалить последнюю строку" onAction="#doRemove"/>

</items>

</Menu>

<Menu mnemonicParsing="false" text="Работа">

<items>

<MenuItem mnemonicParsing="false" text="Сортировать по количеству покупателей" onAction="#doSortByCustomers"/>

<MenuItem mnemonicParsing="false" text="Сортировать по комментариям" onAction="#doSortByComments"/>

</items>

</Menu>

<Menu mnemonicParsing="false" text="Помощь">

<items>

<MenuItem mnemonicParsing="false" text="О программе..." onAction="#doAbout"/>

</items>

</Menu>

</menus>

</MenuBar>

</top>

<left>

<AnchorPane prefHeight="472.0" prefWidth="200.0" BorderPane.alignment="CENTER">

<children>

<Label text="Магазин" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="14.0" />

<TextField fx:id="textFieldStore" prefHeight="22.0" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="35.0" />

<Label text="Адрес" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="69.0" />

<TextField fx:id="textFieldAddress" prefHeight="22.0" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="90.0" onAction="#addressChanged" />

<Label text="Текст для поиска:" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="154.0" />

<TextField fx:id="textFieldText" prefHeight="22.0" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="175.0" />

<Button mnemonicParsing="false" prefHeight="22.0"

text="Искать слово" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="210.0" onAction="#doSearchByWord"/>

<TextArea fx:id="textAreaResults" AnchorPane.bottomAnchor="11.0"

AnchorPane.leftAnchor="11.0" AnchorPane.rightAnchor="11.0"

AnchorPane.topAnchor="280.0" />

</children>

</AnchorPane>

</left>

<center>

<TableView fx:id="tableViewHours" prefHeight="473.0" prefWidth="114.0"

BorderPane.alignment="CENTER" editable="true" >

<columns>

<TableColumn fx:id="tableColumnCustomers" prefWidth="100.0" text="Покупатели" editable="true" />

<TableColumn fx:id="tableColumnComments" prefWidth="200.0" text="Комментарий" editable="true" />

</columns>

</TableView>

</center>

</BorderPane>

**Sample.fxml**

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.\*?>

<?import java.lang.\*?>

<?import javafx.scene.layout.\*?>

<?import javafx.scene.layout.BorderPane?>

<BorderPane prefHeight="500.0" prefWidth="700.0"

xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1"

fx:controller="application.SampleController">

<top>

<MenuBar BorderPane.alignment="CENTER">

<menus>

<Menu mnemonicParsing="false" text="Файл">

<items>

<MenuItem mnemonicParsing="false" text="Новый" onAction="#doNew"/>

<MenuItem mnemonicParsing="false" text="Открыть..." onAction="#doOpen"/>

<MenuItem mnemonicParsing="false" text="Сохранить..." onAction="#doSave"/>

<MenuItem mnemonicParsing="false" text="Выход" onAction="#doExit"/>

</items>

</Menu>

<Menu mnemonicParsing="false" text="Правка">

<items>

<MenuItem mnemonicParsing="false" text="Добавить строку" onAction="#doAdd"/>

<MenuItem mnemonicParsing="false" text="Удалить последнюю строку" onAction="#doRemove"/>

</items>

</Menu>

<Menu mnemonicParsing="false" text="Работа">

<items>

<MenuItem mnemonicParsing="false" text="Сортировать по количеству покупателей" onAction="#doSortByCustomers"/>

<MenuItem mnemonicParsing="false" text="Сортировать по комментариям" onAction="#doSortByComments"/>

</items>

</Menu>

<Menu mnemonicParsing="false" text="Помощь">

<items>

<MenuItem mnemonicParsing="false" text="О программе..." onAction="#doAbout"/>

</items>

</Menu>

</menus>

</MenuBar>

</top>

<left>

<AnchorPane prefHeight="472.0" prefWidth="200.0" BorderPane.alignment="CENTER">

<children>

<Label text="Магазин" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="14.0" />

<TextField fx:id="textFieldStore" prefHeight="22.0" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="35.0" />

<Label text="Адрес" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="69.0" />

<TextField fx:id="textFieldAddress" prefHeight="22.0" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="90.0" onAction="#addressChanged" />

<Label text="Текст для поиска:" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="154.0" />

<TextField fx:id="textFieldText" prefHeight="22.0" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="175.0" />

<Button mnemonicParsing="false" prefHeight="22.0"

text="Искать слово" AnchorPane.leftAnchor="11.0"

AnchorPane.rightAnchor="11.0" AnchorPane.topAnchor="210.0" onAction="#doSearchByWord"/>

<TextArea fx:id="textAreaResults" AnchorPane.bottomAnchor="11.0"

AnchorPane.leftAnchor="11.0" AnchorPane.rightAnchor="11.0"

AnchorPane.topAnchor="280.0" />

</children>

</AnchorPane>

</left>

<center>

<TableView fx:id="tableViewHours" prefHeight="473.0" prefWidth="114.0"

BorderPane.alignment="CENTER" editable="true" >

<columns>

<TableColumn fx:id="tableColumnCustomers" prefWidth="100.0" text="Покупатели" editable="true" />

<TableColumn fx:id="tableColumnComments" prefWidth="200.0" text="Комментарий" editable="true" />

</columns>

</TableView>

</center>

</BorderPane>

*Файл sefe.xml містить такі дані:*

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<StoreData Name="Digma" Address="Saltovskoe shosse, 100">

<HourData Customers="17" Comment="open"/>

<HourData Customers="16" Comment="open airbomb"/>

<HourData Customers="10" Comment="open"/>

<HourData Customers="6" Comment="open"/>

<HourData Customers="3" Comment="open import"/>

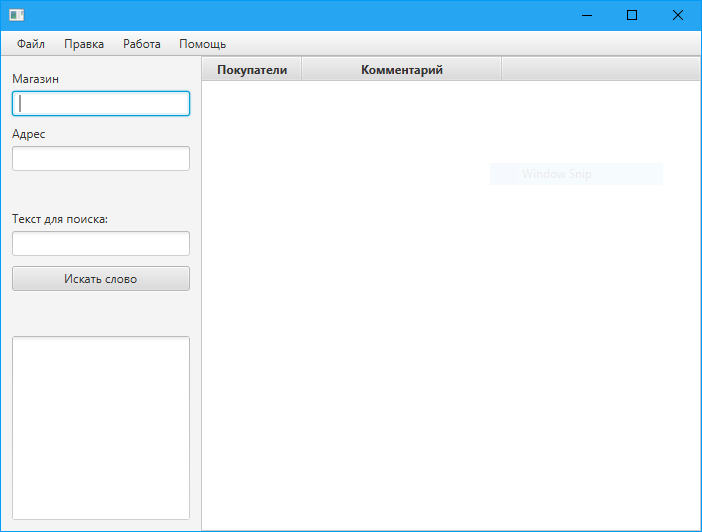
<HourData Customers="18" Comment="open incident"/>

<HourData Customers="7" Comment="open incident"/>

<HourData Customers="11" Comment="open theft"/>

</StoreData>

*Після запуску програми відкривається наступне вікно:*

**