**Белорусский Государственный Университет Информатики и Радиоэлектроники**

Кафедра экономической информатики

Отчёт о лабораторной работе №7

Разработка Java приложений с использованием коллекций

Вариант 1

|  |  |
| --- | --- |
| Выполнил | Студент Рудович Д.П. |
|  | Группа 974001 |
|  |  |
| Проверил | Кдимович К.А. |

Минск 2020

**Задание:** Разработать контейнерный класс «Картинная галерея», элементы которого содержат информацию о картинах. Каждый объект картины содержит поля: название картины, художник, год создания, размер (ширина×высота), вид красок. Разместить объекты картин в контейнере в алфавитном порядке. Реализовать в контейнере метод, сортирующий картины по названию, а также метод, осуществляющий поиск картин одного художника. Результаты выводить в таблицу. Контейнер должен иметь возможность добавления, удаления и редактирования объектов картин.

**Программный код:**

**public** **class** ArtGallery **implements** Comparable<ArtGallery> {

**private** String name;

**private** String painter;

**private** String releaseYear;

**private** String size;

**private** String paintType;

**public** ArtGallery(String name, String painter, String releaseYear, String size, String paintType) **throws** Exception {

**if**(name.isEmpty() || !(name.matches("^\\D+$"))) **throw** **new** Exception("Name information was entered not correctly");

**this**.name = name;

**if**(painter.isEmpty()) **throw** **new** Exception("Information about the author was entered not correctly");

**this**.painter = painter;

**if**(!(releaseYear.matches("[+]?\\d+"))) **throw** **new** Exception("Information about the data of creation was entered not correctly");

**this**.releaseYear = releaseYear;

**if**(size.isEmpty()) **throw** **new** Exception("Information about the size of the picture was entered not correctly");

**this**.size = size;

**if**(paintType.isEmpty()) **throw** **new** Exception("Type of paints was entered not correctly");

**this**.paintType = paintType;

}

**public** String getName() {

**return** name;

}

**public** String getPainter() {

**return** painter;

}

**public** String getReleaseYear() {

**return** releaseYear;

}

**public** String getSize() {

**return** size;

}

**public** String getPaintType() {

**return** paintType;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **void** setPainter(String painter) {

**this**.painter = painter;

}

**public** **void** setReleaseYear(String releaseYear) {

**this**.releaseYear = releaseYear;

}

**public** **void** setSize(String size) {

**this**.size = size;

}

**public** **void** setPaintType(String paintType) {

**this**.paintType = paintType;

}

@Override

**public** **int** compareTo(ArtGallery o) {

**int** result = **this**.name.compareTo(o.name);

**return** result;

}

}

**import** javax.swing.table.AbstractTableModel;

**import** java.util.ArrayList;

**import** java.util.Collections;

**public** **class** ArtGalleryTable **extends** AbstractTableModel {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** **int** columnCount = 5;

**private** ArrayList<ArtGallery> dataArrayList;

**public** ArtGalleryTable(){

dataArrayList = **new** ArrayList<ArtGallery>();

}

@Override

**public** **int** getRowCount() {

**return** dataArrayList.size();

}

**public** ArtGallery getRow(**int** i) {

**return** dataArrayList.get(i);

}

@Override

**public** **int** getColumnCount() {

**return** columnCount;

}

@Override

**public** Object getValueAt(**int** rowIndex, **int** columnIndex) {

ArtGallery rows = dataArrayList.get(rowIndex);

**switch**(columnIndex){

**case** 0: **return** rows.getName();

**case** 1: **return** rows.getPainter();

**case** 2: **return** rows.getReleaseYear();

**case** 3: **return** rows.getSize();

**case** 4: **return** rows.getPaintType();

}

**return** "";

}

@Override

**public** String getColumnName(**int** columnIndex){

**switch**(columnIndex){

**case** 0: **return** "Name";

**case** 1: **return** "Artist";

**case** 2: **return** "Data of creation";

**case** 3: **return** "Size";

**case** 4: **return** "Type of paints";

}

**return** "";

}

**public** **void** addData(ArtGallery row){

dataArrayList.add(row);

Collections.*sort*(dataArrayList);

}

**public** ArrayList<ArtGallery> findSimilar(String painter){

ArrayList<ArtGallery> similarArrayList = **new** ArrayList<ArtGallery>();

**for**(**int** i = 0; i < dataArrayList.size();i++) {

ArtGallery row = dataArrayList.get(i);

String currPainter = row.getPainter();

**if** (currPainter.equals(painter)){

similarArrayList.add(row);

}

}

**return** similarArrayList;

}

**public** **void** clear() {

dataArrayList.clear();

}

**public** **void** removeRow(**int** i) {

dataArrayList.remove(i);

}

}

**import** javax.swing.\*;

**public** **class** DataException **extends** Exception {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** String info;

**public** DataException (String str) {

info = str;

}

**public** **void** getInfo() {

JOptionPane.*showMessageDialog*(**null**, **this**.info, "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

**public** **class** Main {

**public** **static** **void** main(String[] args) **throws** Exception {

Reader r = **new** Reader();

}

}

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.util.ArrayList;

**public** **class** Reader {

JButton addButton, deleteButton, clearButton, similarButton;

**static** JTextField *newName*, *newPainter*, *newReleaseYear*, *newSize*, *newPaintType*;

**static** JLabel *l1*, *l2*, *l3*, *l4*, *l5*;

JFrame frame;

JPanel panelButton, panelButton1;

JTable artTable;

ArtGalleryTable atm;

ArtGallery art = **null**, art1 = **null**, art2 = **null**, art3 = **null**, art4 = **null**, art5=**null**;

**public** Reader() **throws** Exception {

frame = **new** JFrame("Art Gallery");

frame.setSize(**new** Dimension(600, 400));

frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

frame.setLocationRelativeTo(**null**);

frame.setLayout(**new** BorderLayout());

addButton = **new** JButton("Add");

addButton.setBackground(Color.*decode*("#90EE90"));

addButton.addActionListener(**new** AddActionListener());

deleteButton = **new** JButton("Delete");

deleteButton.setBackground(Color.*decode*("#F08080"));

deleteButton.addActionListener(**new** DeleteActionListener());

clearButton = **new** JButton("Clear");

clearButton.setBackground(Color.*decode*("#DC143C"));

clearButton.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

atm.clear();

atm.fireTableDataChanged();

}

});

similarButton = **new** JButton("Search of paint");

similarButton.setBackground(Color.***white***);

similarButton.addActionListener(**new** SearchActionListener());

atm = **new** ArtGalleryTable();

artTable = **new** JTable(atm);

frame.add(**new** JScrollPane(artTable), BorderLayout.***CENTER***);

frame.setVisible(**true**);

**try** {

art = **new** ArtGallery("Mona Lisa", "Da Vinci", "1503", "77 x 53", "Oil");

art1 = **new** ArtGallery("Starry Night", "Vincent Van Gogh", "1889", "74 x 92", "Oil");

art2 = **new** ArtGallery("Scream", "Edvard Munch", "1893", "91 x 73", "Gouache");

art3 = **new** ArtGallery("Night Watch", "Rembrandt", "1642", "363 x 437", "Oil");

art4 = **new** ArtGallery("Gleaners", "Jean-Francois Millet", "1857", "84 x 112", "Acrylic");

} **catch** (DataException dataException) {

dataException.getInfo();

}

atm.addData(art);

atm.addData(art1);

atm.addData(art2);

atm.addData(art3);

atm.addData(art4);

*newName* = **new** JTextField(15);

*newPainter* = **new** JTextField(15);

*newReleaseYear* = **new** JTextField(15);

*newSize* = **new** JTextField(15);

*newPaintType* = **new** JTextField(15);

*l1* = **new** JLabel("Name of the pictire");

*l2* = **new** JLabel("Artist");

*l3* = **new** JLabel("Date of creation");

*l4* = **new** JLabel("Size");

*l5* = **new** JLabel("Type of paints");

panelButton = **new** JPanel();

panelButton.setLayout(**new** GridLayout());

panelButton.add(addButton);

panelButton.add(similarButton);

panelButton.add(deleteButton);

panelButton.add(clearButton);

panelButton.setVisible(**true**);

frame.add(panelButton, BorderLayout.***PAGE\_END***);

panelButton1 = **new** JPanel();

panelButton1.setLayout(**new** BoxLayout(panelButton1,BoxLayout.***Y\_AXIS***));

panelButton1.add(*l1*);

panelButton1.add(*newName*);

panelButton1.add(*l2*);

panelButton1.add(*newPainter*);

panelButton1.add(*l3*);

panelButton1.add(*newReleaseYear*);

panelButton1.add(*l4*);

panelButton1.add(*newSize*);

panelButton1.add(*l5*);

panelButton1.add(*newPaintType*);

panelButton.setVisible(**true**);

frame.add(panelButton1, BorderLayout.***LINE\_END***);

frame.pack();

}

**public** **class** SearchActionListener **implements** ActionListener {

**public** **void** actionPerformed(ActionEvent e) {

String searchInfo = JOptionPane.*showInputDialog*(**null**, "Enter the name of the artist to search",

"Search the paint with the name of the artist", JOptionPane.***QUESTION\_MESSAGE***);

ArrayList<ArtGallery> similarArrayList = atm.findSimilar(searchInfo);

SimilarData similarData = **new** SimilarData(similarArrayList);

similarData.setVisible(**true**);

similarData.setLocationRelativeTo(**null**);

}

}

**public** **class** AddActionListener **implements** ActionListener {

**public** **void** actionPerformed(ActionEvent e) {

String name= *newName*.getText();

String painter= *newPainter*.getText();

String year= *newReleaseYear*.getText();

String size= *newSize*.getText();

String paintType= *newPaintType*.getText();

**try** {

atm.addData(**new** ArtGallery(name,painter,year,size,paintType));

} **catch** (DataException dataException) {

dataException.getInfo();

} **catch** (Exception e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

atm.fireTableDataChanged();

*newName*.setText("");

*newPainter*.setText("");

*newReleaseYear*.setText("");

*newSize*.setText("");

*newPaintType*.setText("");

}

}

**public** **class** DeleteActionListener **implements** ActionListener {

**public** **void** actionPerformed(ActionEvent e) {

**int** i = artTable.getSelectedRow();

**if**(i >= 0) {

atm.removeRow(i);

atm.fireTableDataChanged();

}

**else** JOptionPane.*showMessageDialog*(**null**, "Enter the string which you want to delete", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

}

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.util.ArrayList;

**public** **class** SimilarData **extends** JFrame{

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

JTable similarTable;

ArtGalleryTable stm;

**public** SimilarData(ArrayList<ArtGallery> similarArrayList){

**super**("Paintings with the similar painter");

setSize(**new** Dimension(600, 300));

setLocationRelativeTo(**null**);

setLayout(**new** BorderLayout());

stm = **new** ArtGalleryTable();

similarTable = **new** JTable(stm);

add(**new** JScrollPane(similarTable), BorderLayout.***CENTER***);

**for**(**int** i = 0; i < similarArrayList.size();i++){

stm.addData(similarArrayList.get(i));

}

}

}

