МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра вычислительной техники

ОТЧЕТ

по лабораторной работе № 4
по дисциплине «Объектно-ориентированное программирование»
Тема: Обработка исключений

Студент гр. 3312 _____ Сабиров Р. Д.

Преподаватель _____ Павловский М. Г.

Санкт-Петербург
2024

Содержание

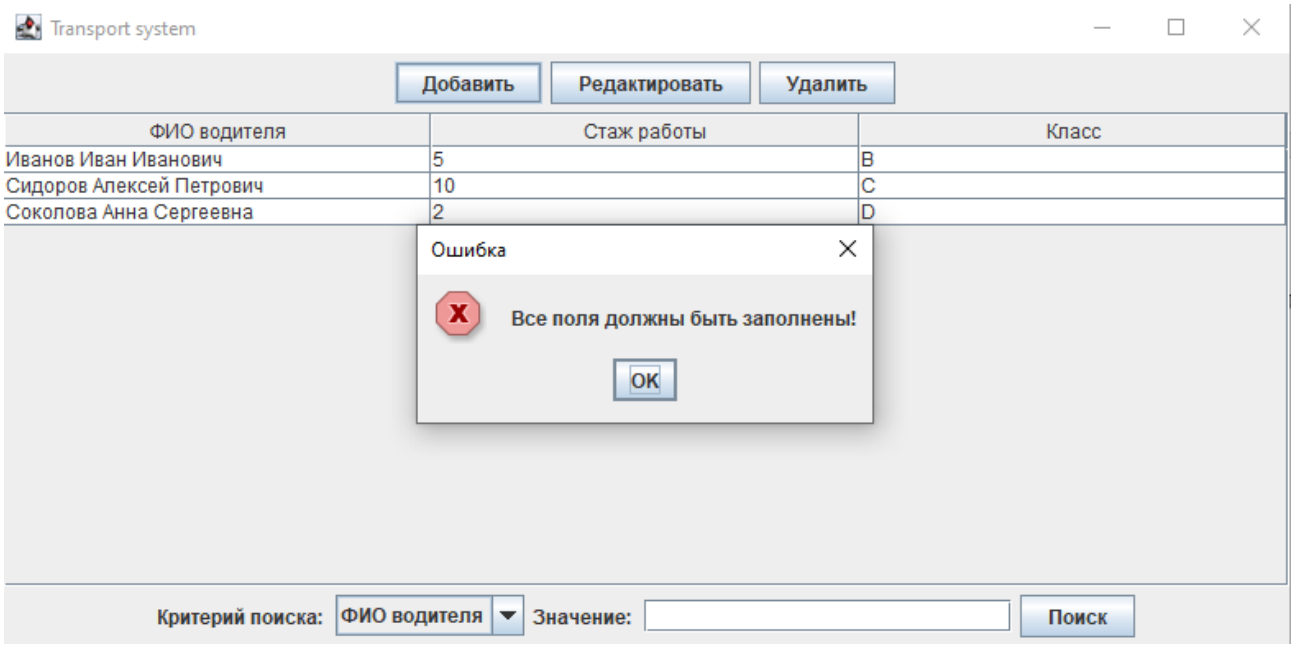
Цель работы

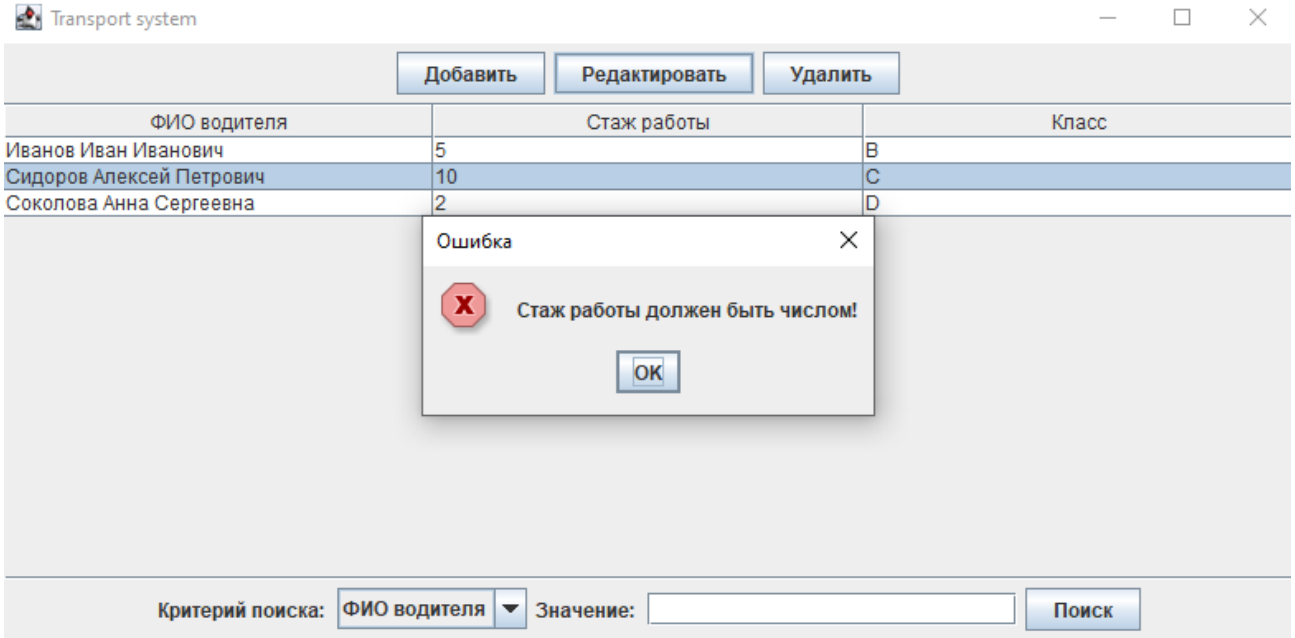
Знакомство с механизмом обработки исключений в языке программирования Java.

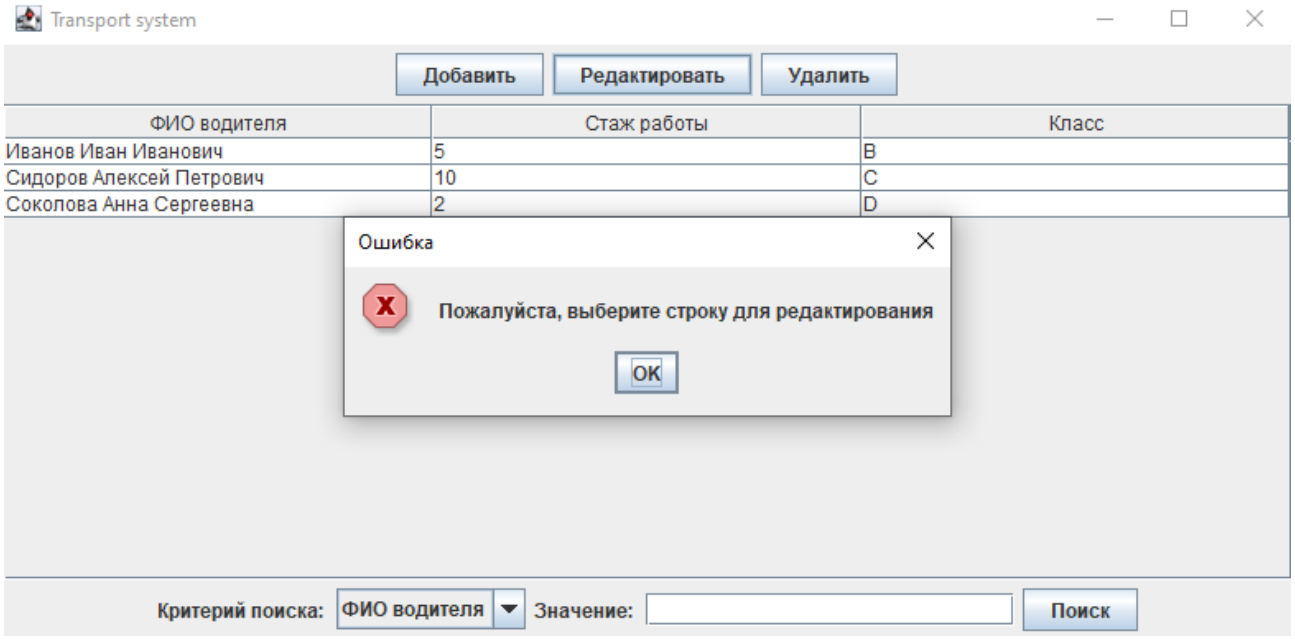
Перечень ситуаций, которые контролируются с помощью исключений

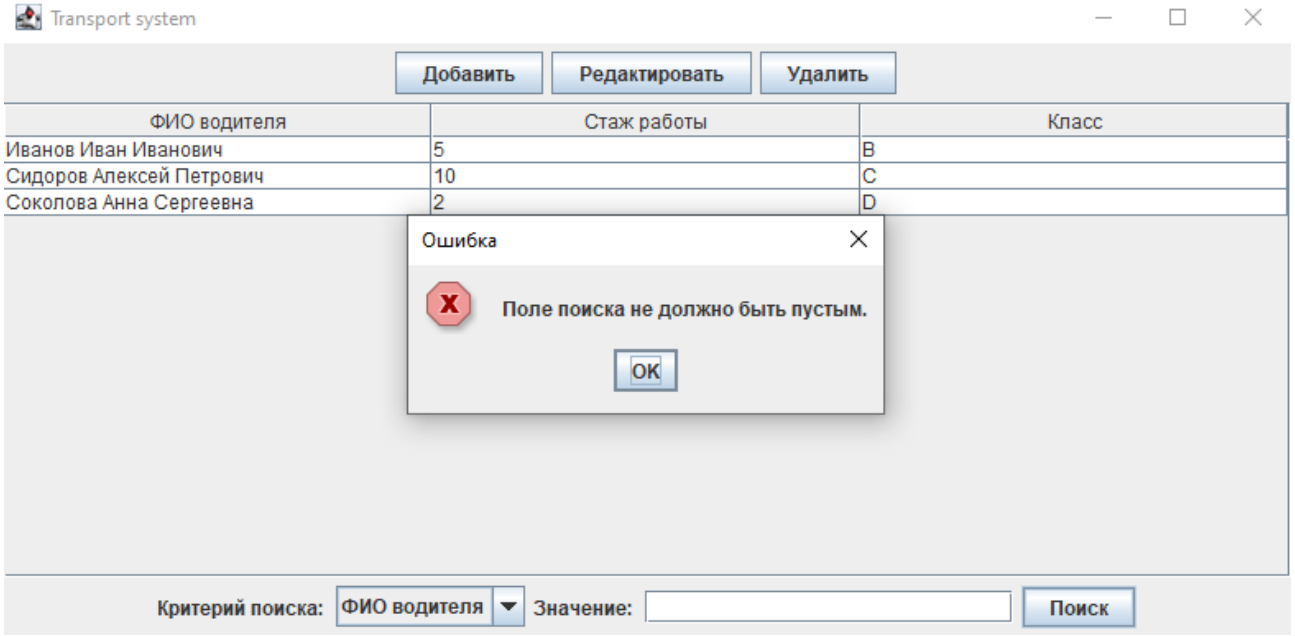
1. Пустые поля при добавлении или редактировании записи
    - При попытке добавить или редактировать запись с пустыми полями используется исключение InvalidFieldException.
    - Это исключение уведомляет пользователя, что все поля должны быть заполнены, чтобы избежать неполных данных в таблице.
2. Некорректный формат стажа работы
    - Если в поле "Стаж работы" введены некорректные данные, используется исключение InvalidFieldException.
    - Исключение информирует пользователя о необходимости вводить числовое значение, предотвращая ошибки формата данных.
3. Отсутствие выбора строки при редактировании и удалении
    - При нажатии кнопки "Редактировать" или "Удалить" без выбора строки обрабатывается исключение RowNotSelectedException.
    - Это исключение предупреждает пользователя, что необходимо выбрать строку для редактирования или удаления, что предотвращает попытку редактирования и удаления несуществующей записи.
4. Отсутствие текста в поле для поиска
    - При нажатии на кнопку "Поиск", если поле для поиска не заполнено, используется исключение EmptySearchFieldException.
    - Исключение информирует пользователя о том, что необходимо ввести значение для поиска записей.

Скриншоты, иллюстрирующие работу обработчиков ситуаций






Transport system
Добавить
Редактировать
Удалить
ФИО водителя
Стаж работы
Класс
Иванов Иван Иванович
5
B
Сидоров Алексей Петрович
10
C
Соколова Анна Сергеевна
2
D
Ошибка
Пожалуйста, выберите строку для редактирования
OK
Критерий поиска:
ФИО водителя
Значение:
Поиск


Transport system
Добавить
Редактировать
Удалить
ФИО водителя
Стаж работы
Класс
Иванов Иван Иванович
5
B
Сидоров Алексей Петрович
10
C
Соколова Анна Сергеевна
2
D
Ошибка
Поле поиска не должно быть пустым.
OK
Критерий поиска:
ФИО водителя
Значение:
Поиск

Текст документации, сгенерированный Javadoc

PACKAGE CLASS USE TREE INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class lab04

java.lang.Object
lab04

```
public class lab04
extends Object
```

Version:

1.0

Author:

Сабиров Роман 3312

Constructor Summary

Constructors

Constructor	Description
lab04()	

Method Summary

All Methods | **Static Methods** | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
void	Bus()	
static void	main(String[] args)	

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

lab04

```
public lab04()
```

Method Details

Bus

```
public void Bus()
```

main

```
public static void main(String[] args)
```

Parameters:

args - аргументы командной строки (не используются)

Исходные тексты слушателей

```
    /**
     * Метод для добавления слушателей к кнопкам
     */
    private void addListeners() {
        // Слушатель для кнопки "Добавить"
        addButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                addDriverDialog();
            }
        });

        // Слушатель для кнопки "Редактировать"
        editButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    int selectedRow = driverTable.getSelectedRow();
                    if (selectedRow == -1) {
                        throw new RowNotSelectedException("Пожалуйста,
выберите строку для редактирования");
                    }
                    editDriverDialog(selectedRow);
                } catch (RowNotSelectedException ex) {
                    JOptionPane.showMessageDialog(frame, ex.getMessage(),
"Ошибка", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        // Слушатель для кнопки "Удалить"
        deleteButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    int selectedRow = driverTable.getSelectedRow();
                    if (selectedRow == -1) {
                        throw new RowNotSelectedException("Пожалуйста,
выберите строку для удаления");
                    }
                    tableModel.removeRow(selectedRow);
                    JOptionPane.showMessageDialog(frame, "Водитель
удален");

                } catch (RowNotSelectedException ex) {
                    JOptionPane.showMessageDialog(frame, ex.getMessage(),
"Ошибка", JOptionPane.ERROR_MESSAGE);
                }
            }
        });
        // Слушатель для кнопки поиск
        searchButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    String criterion = (String)
searchCriteria.getSelectedItem();
                    String value =
searchField.getText().trim().toLowerCase();
```

```
                    // Проверка на пустое поле поиска
                    if (value.isEmpty()) {
                        throw new EmptySearchFieldException("Поле поиска
не должно быть пустым.");
                    }

                    driverTable.clearSelection();
                    boolean found = false;

                    // Поиск значений в таблице по критерию
                    for (int i = 0; i < tableModel.getRowCount(); i++) {
                        String cellValue = tableModel.getValueAt(i,
searchCriteria.getSelectedIndex()).toString().toLowerCase();
                        if (cellValue.contains(value)) {
                            driverTable.addRowSelectionInterval(i, i);
                            found = true;
                        }
                    }

                    // Сообщение об успехе или отсутствии совпадений
                    if (found) {
                        JOptionPane.showMessageDialog(frame, "Найдены
совпадения по критерию: " + criterion);
                    } else {
                        JOptionPane.showMessageDialog(frame, "Совпадений
не найдено");
                    }

                } catch (EmptySearchFieldException ex) {
                    JOptionPane.showMessageDialog(frame, ex.getMessage(),
"Ошибка", JOptionPane.ERROR_MESSAGE);
                }
            }
        });
    }

    /**
     * Метод для отображения диалогового окна добавления водителя
     */
    private void addDriverDialog() {
        JPanel panel = new JPanel(new GridLayout(3, 2));
        JTextField nameField = new JTextField(20);
        JTextField experienceField = new JTextField(20);
        JTextField categoryField = new JTextField(20);

        panel.add(new JLabel("ФИО: "));
        panel.add(nameField);
        panel.add(new JLabel("Стаж работы: "));
        panel.add(experienceField);
        panel.add(new JLabel("Класс: "));
        panel.add(categoryField);

        int result = JOptionPane.showConfirmDialog(frame, panel, "Добавить
водителя", JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
```

```
        if (result == JOptionPane.OK_OPTION) {
            try {
                String name = nameField.getText();
                String experience = experienceField.getText();
                String category = categoryField.getText();

                validateFields(name, experience, category); // Проверка
полей

                tableModel.addRow(new Object[]{name, experience,
category});
                JOptionPane.showMessageDialog(frame, "Водитель
добавлен!");

            } catch (InvalidFieldException ex) {
                JOptionPane.showMessageDialog(frame, ex.getMessage(),
"Ошибка", JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    /**
     * Метод для отображения диалогового окна редактирования водителя
     * @param row номер редактируемой строки
     */
    private void editDriverDialog(int row) {
        JPanel panel = new JPanel(new GridLayout(3, 2));
        JTextField nameField = new JTextField((String)
tableModel.getValueAt(row, 0), 20);
        JTextField experienceField = new JTextField((String)
tableModel.getValueAt(row, 1), 20);
        JTextField categoryField = new JTextField((String)
tableModel.getValueAt(row, 2), 20);

        panel.add(new JLabel("ФИО: "));
        panel.add(nameField);
        panel.add(new JLabel("Стаж работы: "));
        panel.add(experienceField);
        panel.add(new JLabel("Класс: "));
        panel.add(categoryField);

        int result = JOptionPane.showConfirmDialog(frame, panel,
"Редактировать водителя", JOptionPane.OK_CANCEL_OPTION,
JOptionPane.PLAIN_MESSAGE);

        if (result == JOptionPane.OK_OPTION) {
            try {
                String name = nameField.getText();
                String experience = experienceField.getText();
                String category = categoryField.getText();

                validateFields(name, experience, category); // Проверка
полей

                tableModel.setValueAt(name, row, 0);
                tableModel.setValueAt(experience, row, 1);
                tableModel.setValueAt(category, row, 2);
                JOptionPane.showMessageDialog(frame, "Данные водителя
обновлены!");
```

```
            } catch (InvalidFieldException ex) {
                JOptionPane.showMessageDialog(frame, ex.getMessage(),
"Ошибка", JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    /**
     * Метод для проверки корректности введенных данных
     * @param name        ФИО водителя
     * @param experience  Стаж
     * @param category    Класс
     * @throws InvalidFieldException если хотя бы одно из полей пустое или
стаж работы не является целым числом
     */
    private void validateFields(String name, String experience, String
category) throws InvalidFieldException {
        if (name.isEmpty() || experience.isEmpty() || category.isEmpty())
{
            throw new InvalidFieldException("Все поля должны быть
заполнены!");
        }
        try {
            Integer.parseInt(experience); // Проверка, что стаж - целое
число
        } catch (NumberFormatException ex) {
            throw new InvalidFieldException("Стаж работы должен быть
числом!");
        }
    }

/**
 * Исключение, если одно или несколько полей формы содержат недопустимые
данные.
 */
class InvalidFieldException extends Exception {
    public InvalidFieldException(String message) {
        super(message);
    }
}

/**
 * Исключение, если строка таблицы не выбрана
 */
class RowNotSelectedException extends Exception {
    public RowNotSelectedException(String message) {
        super(message);
    }
}

/**
 * Исключение, если поле поиска пустое
 */
class EmptySearchFieldException extends Exception {
    public EmptySearchFieldException(String message) {
        super(message);
    }
}
```

Вывод

В ходе выполнения работы была разработана обработка исключений при работе с экранной формой для отображения и управления списком водителей с использованием языка программирования Java.

Ссылка на репозиторий: https://github.com/sabiroma/OOP_labs/tree/main/lab04

Ссылка на видео: https://disk.yandex.ru/i/5IyIpaMo36FZuA