**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. 4311 |  | Левицкий М. Ю. |
| Преподаватель |  | Лазарев Э.А. |

Санкт-Петербург

2025

# ЦЕЛЬ

Знакомство с механизмом обработки исключений в языке Java.

* 1. Создайте новый проект, который будет дублировать проект лабораторной работы № 3.
* 2. Проанализируйте методы приложения и определите ошибочные ситуации, которые необходимо в них контролировать.
* 3. Напишите 1−2 класса собственных исключений. В текст классов должны быть вставлены комментарии документации, поясняющие типы контролируемых ситуаций.
* 4. Задайте спецификацию исключений и разработайте код методов, где генерируются исключительные ситуации.
* 5. Разработайте код методов, где контролируются и обрабатываются исключительные ситуации.
* 6. Запустите приложение и снимите с экрана скриншоты, иллюстрирующие работу обработчиков ситуаций.
* 7. Сгенерируйте документацию с помощью Javadoc и просмотрите ее в браузере.

**ПЕРЕЧЕНЬ СИТУАЦИЙ, КОТОРЫЕ КОНТРОЛИРУЮТСЯ С ПОМОЩЬЮ ИСКЛЮЧЕНИЙ.**

1.Пустые поля при добавлении/редактировании клиента  
2. Неверный формат имени клиента  
3. Неверный формат телефона  
4. Пустой поисковый запрос  
5. Попытка удалить клиента без выбора

6. Попытка редактировать клиента без выбора

7. Результаты поиска не найдены

**СКРИНШОТЫ, ИЛЛЮСТРИРУЮЩИЕ РАБОТУ ОБРАБОТЧИКОВ СИТУАЦИЙ**

|  |
| --- |
|  |

**Скриншот 1**

|  |
| --- |
|  |

**Скриншот 2**

|  |
| --- |
|  |

**Скриншот 3**

|  |
| --- |
|  |

**Скриншот 4**

|  |
| --- |
|  |

**Скриншот 5**

|  |
| --- |
|  |

**Скриншот 6**

|  |
| --- |
|  |

**Скриншот 7**

# ТЕКСТ ДОКУМЕНТАЦИИ

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

**ИСХОДНЫЕ ТЕКСТЫ КЛАССОВ СОБСТВЕННЫХ ИСКЛЮЧЕНИЙ, МЕТОДОВ, ГДЕ ГЕНЕРИРУЮТСЯ И ОБРАБАТЫВАЮТСЯ ИСКЛЮЧИТЕЛЬНЫЕ СИТУАЦИИ**

**1.Классы собственных исключений**

MyException.java

|  |
| --- |
| public class MyException extends Exception {  public MyException(String message) {  super(message);  }  } |

ClientNotFoundException.java

|  |
| --- |
| public class ClientNotFoundException extends Exception {  public ClientNotFoundException(String message) {  super(message);  }  } |

**2.Методы, где генерируются исключительные ситуации**

Метод валидации данных

|  |
| --- |
| private void validateClientData(String name, String phone, String address, String newspaper)  throws MyException {    if (name.isEmpty() || phone.isEmpty() || address.isEmpty() || newspaper.isEmpty()) {  throw new MyException("All fields must be filled!");  }    if (!name.matches("[a-zA-Zа-яА-Я]+")) {  throw new MyException("Client name must contain only letters!");  }    if (!phone.matches("\\d{3}-\\d{2}-\\d{2}")) {  throw new MyException("Phone must be in format XXX-XX-XX!");  }  } |

Обработчик кнопки Delete

|  |
| --- |
| if (selectedRow == -1) {  throw new ClientNotFoundException("Please select a client to delete!");  } |

Обработчик кнопки Edit

|  |
| --- |
| if (selectedRow == -1) {  throw new ClientNotFoundException("Please select a client to edit!");  } |

Обработчик кнопки Search

|  |
| --- |
| if (searchText.isEmpty()) {  throw new MyException("Please enter text to search!");  }  if (!found) {  throw new ClientNotFoundException("No clients found for query: '" + searchText + "'");  } |

3. **Методы, где обрабатываются исключительные ситуации**

Обработка в методе Add

|  |
| --- |
| try {  validateClientData(name, phone, address, newspaper);  // код добавления  } catch (MyException ex) {  JOptionPane.showMessageDialog(parent, ex.getMessage(), "Input Error", JOptionPane.ERROR\_MESSAGE);  } |

Обработка в методе Edit

|  |
| --- |
| try {  // код редактирования  } catch (ClientNotFoundException ex) {  JOptionPane.showMessageDialog(parent, ex.getMessage(), "Edit Error", JOptionPane.WARNING\_MESSAGE);  } catch (MyException ex) {  JOptionPane.showMessageDialog(parent, ex.getMessage(), "Input Error", JOptionPane.ERROR\_MESSAGE);  } |

Обработка в методе Delete

|  |
| --- |
| try {  // код удаления  } catch (ClientNotFoundException ex) {  JOptionPane.showMessageDialog(parent, ex.getMessage(), "Delete Error", JOptionPane.WARNING\_MESSAGE);  } |

Обработка в методе Search

|  |
| --- |
| try {  // код поиска  } catch (MyException | ClientNotFoundException ex) {  JOptionPane.showMessageDialog(parent, ex.getMessage(), "Search Error", JOptionPane.WARNING\_MESSAGE);  } |

# ВЫВОДЫ

В результате выполнения лабораторной работы я изучил механизм обработки исключений в Java. Освоил создание пользовательских классов исключений и их применение в приложении. Получил практические навыки генерации и обработки исключительных ситуаций. Научился обеспечивать стабильность работы программы при возникновении ошибок.