Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра проектирования информационно-компьютерных систем

ОТЧЁТ

По лабораторной работе №2

по теме «Обработка исключений»

Выполнил:

Студент ФКП

Группа 114301

Тиунчик Д.Р.

Вариант № 26

Проверил:

Куприянов Н.И.

Минск 2021

**Цель работы**: научиться обрабатывать исключения, используя блоки кода try() {} catch(){} и добавлять собственные исключения.

**Индивидуальное задание:**

**Вариант 26:** Исходя из варианта задания лабораторной работы №1 и исходного кода по этой лабораторной необходимо:

1. Реализовать обработку исключительных ситуаций посредством блоков кода try() {} catch{}.
2. Добавить собственное исключение, унаследовав его от типа Exception, логическая составляющая исключения определяется исходя из ошибок, которые могут возникнуть в исходном коде л.р. №1.

**Реализация:**

**Блоки кода с обработкой исключений:**

**1.**  
 while (flag) {  
 System.*out*.println("1. Добавить автомобиль");  
 System.*out*.println("2. Снять автомобиль(вместе с водителем)");  
 System.*out*.println("3. Показать автопарк");  
 System.*out*.println("4. Показать заявки на ремонт");  
 System.*out*.println("0. Выход");  
 System.*out*.println("Ваш выбор: ");  
 try {  
 choice = scanner.nextInt();  
 switch (choice) {  
 case 1:  
 System.*out*.println("Автомобиль: ");  
 name = scanner.next();  
 System.*out*.println("Состояние: ");  
 status = scanner.nextInt();  
 add(new Auto(name, status));  
 read();  
 break;  
 case 2:  
 show();  
 if (Auto.size() != 0) {  
 System.*out*.println("Ваш выбор: ");  
 id = scanner.nextInt();  
 delete(id - 1);  
 }  
 break;  
 case 3:  
 show();  
 read();  
 break;  
 case 4:  
 System.*out*.print("Имя водителя: ");  
 System.*out*.println(rName);  
 app.show();  
 System.*out*.println("Отправить на заказ?");  
 System.*out*.println("1. Да");  
 System.*out*.println("0. Нет");  
 choice2 = scanner.nextInt();  
 if (choice2 == 0) {  
 app.setOK(false);  
 } else {  
 app.setOK(true);  
 }  
 break;  
 case 0:  
 flag = false;  
 break;  
 default:  
 System.*out*.println("Неверный ввод");  
 break;  
 }  
 }  
 catch (Exception ex)  
 {  
 System.*out*.println("Некорректный ввод, попробуйте еще раз");  
 scanner.nextLine();  
 continue;  
 }  
 }  
 }  
  
**2.**  
 for(Auto p : flight.getAuto())  
 System.*out*.printf("Автомобиль: %s \t Состояние: %d \n", p.getAuto(), p.getStatus());  
  
 while(cycle){  
 System.*out*.println("1. Выбрать автомобиль для рейса");  
 System.*out*.println("2. Отменить рейс");  
 System.*out*.println("3. Показать текущие рейсы");  
 System.*out*.println("4. Сделать отметку о состоянии автомобиля");  
 System.*out*.println("5. Установить дату ремонта");  
 System.*out*.println("0. Выход");  
 System.*out*.println("Ваш выбор: ");  
 try {  
 choice = scanner.nextInt();  
 switch (choice) {  
 case 1:  
 base.show();  
 if (base.getSize() != 0) {  
 System.*out*.println("Ваш выбор: ");  
 id = scanner.nextInt();  
 app.add(base.getFlight(id - 1));  
 }  
 break;  
 case 2:  
 app.show();  
 if (app.getSize() != 0) {  
 System.*out*.println("Ваш выбор: ");  
 id = scanner.nextInt();  
 app.delete(id - 1);  
 }  
 break;  
 case 3:  
 app.show();  
 break;  
 case 4:  
 app.makeStatus();  
 break;  
 case 5:  
 System.*out*.println("День: ");  
 day = scanner.nextInt();  
 System.*out*.println("Месяц: ");  
 month = scanner.nextInt();  
 System.*out*.println("Год: ");  
 year = scanner.nextInt();  
 Date date = new Date(day, month, year);  
 app.setDate(date);  
 break;  
 case 0:  
 cycle = false;  
 break;  
 }  
 }  
 catch (Exception ex)  
 {  
 System.*out*.println("Неверный ввод, войдите заново");  
 scanner.nextLine();  
 continue;  
 }  
  
 }  
 }  
  
**3.**  
 Date(int day, int month, int year)  
 {  
 try  
 {  
 if (year==0)  
 {  
 throw new NotCorrectYear();  
 }  
 this.month = month;  
 this.day = day;  
 this.year = year;  
 }  
 catch(Exception a)  
 {  
 System.*out*.println(a.getMessage()+"\n");  
 }  
 }  
  
**4.**  
 try {  
 throw new NotCorrectYear();  
 }  
 catch(NotCorrectYear a)  
 {  
 System.out.println(a.getMessage()+"\n");  
 }

**5.**  
 *Dispatcher* = new Dispatcher();  
 Driver driver = new Driver();  
 String name = new String();  
 *Dispatcher*.add(new Auto("Волга", 1));  
 *Dispatcher*.add(new Auto("Ауди", 0));  
 *Dispatcher*.add(new Auto("БМВ", 1));  
 try(ObjectOutputStream file = new ObjectOutputStream(new FileOutputStream("Auto.txt")))  
 {  
 file.writeObject(*Dispatcher*);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 try {  
 choice = scanner.nextInt();  
 switch (choice) {  
 case 1:  
 System.*out*.print("Введите своё имя: ");  
 name = scanner.next();  
 driver.setStatus(name);  
 driver.DriverMenu(*Dispatcher*);  
 break;  
 case 2:  
 *Dispatcher*.AutoBaseMenu(driver.getApp(), driver.getStatus());  
 break;  
 case 3:  
 System.*out*.print("Эта функция ещё не добавлена, следите за обновлениями\n");  
 break;  
 case 4:  
 try(ObjectOutputStream a = new ObjectOutputStream(new FileOutputStream("Auto.txt")))  
 {  
 a.writeObject(*Dispatcher*);  
 //a.close(Dispatcher);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 break;  
 case 0:  
 flag = false;  
 break;  
 }  
 }  
 catch (Exception ex)  
 {  
 // System.out.println("Неверный ввод, войдите заново");  
 scanner.nextLine();  
 continue;  
 }  
 }  
}  
  
**Класс NotCorrectYear:**

package com.company;  
  
public class NotCorrectYear extends Exception  
{  
 public NotCorrectYear(){super("Некорректный год");}  
}

**Вывод:** научился обрабатывать исключения с помощью блоков кода try(){} catch() {} , добавлять и обрабатывать собственные исключения.