Частное учреждение образования

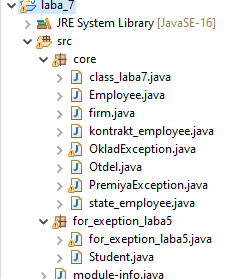
«Колледж бизнеса и права»

Тема: «Обработка исключений»

Учащийся М.Н. Гончаров.

группа: Т-992

2021



Рисунок

Текст задачи 1

Для своего варианта задания лабораторной работы No 5

оптимизировать программу, включив в нее обработку исключительных

ситуаций: ввода данных, операции деления и др.каждого перебора создать свой метод.

// pakage for\_exeption\_laba5

// main

package for\_exeption\_laba5;

import java.util.HashMap;

import java.util.InputMismatchException;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Scanner; // Консольный ввод import java.util.Scanner;

// Т.к они все в одном пакете то не надо импортить их /// ЭТО полная копия проекта laba5 + добавление исключений по заданию

///////////////////////////////////////// НАПРИМЕР РЕАЛИЗОВАЛ ИСКЛЮЧЕНИЯ ВО 2 задании, остальные выпилил задания

////////////////////////////////////////////////////////////////////////////////////////////////////

public class for\_exeption\_laba5 {

// for 2 task

public static void SortStudent(ArrayList<Student> listStudent)

{

// Сначала нахожу средний балл

int sredn = 0;

for (int i = 0; i < listStudent.size(); i++)

{

sredn = 0; // Зануляю для каждого студента

for (int j = 0; j < listStudent.get(i).marks.size(); j++) // listStudent.get(i).marks.size() Проверяю каждый список отметок у каждого студента

{

sredn += listStudent.get(i).marks.get(j);

}

// Посчитал

try

{

sredn /= listStudent.get(i).marks.size();

//sredn /= 0; // Чисто для проверки исключени, среднее делю на ноль типо нету вообще оценок

}

catch(ArithmeticException ex)

{

System.out.println("EXCEPTION 4: " + ex.getMessage());

}

if (sredn < 3)

{

listStudent.remove(i);

}

else

{

// Значит больше или равно 3 // >=3

if (listStudent.get(i).getNumKursa() < 4) // Проверка зачем его отправлять на 5 курс если например всего 4

{

listStudent.get(i).setNumKursa(listStudent.get(i).getNumKursa()+1);

}

}

}

}

// task 2.2

public static void printStudentInList\_vibor\_pocursu(ArrayList<Student> listStudent, int numCourse)

{

for (int i = 0; i < listStudent.size(); i++)

{

if (listStudent.get(i).getNumKursa() == numCourse) // Если совпадает

{

System.out.println("Student name: " + listStudent.get(i).getName());

}

}

}

// main

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

//////////////////////////////////// 2

System.out.println("\nTask 2");

// Список студентов

ArrayList<Student> listStudent = new ArrayList<Student>(2); // МОЖЕТ БЫТЬ ТОЛЬКО 2 студента, но исключение почемуто пропускает

// Для ввода

Scanner in = new Scanner(System.in);

// Первый студент и его оценки

ArrayList<Integer> listMarksMatvey = new ArrayList<Integer>();

// {9, 8, 9, 10};

// ЗАКОМЕНТИЛ ЧТОБЫ ВЫПОЛНИТЬ ЗАДАНИЕ ПРОВЕРКА НА ВВОД

// listMarksMatvey.add(9);

// listMarksMatvey.add(8);

// listMarksMatvey.add(9);

// listMarksMatvey.add(10);

// Второй студент и его оценки

ArrayList<Integer> listMarksNikita = new ArrayList<Integer>();

// {3, 2, 1, 2};

// listMarksNikita.add(3);

// listMarksNikita.add(2);

// listMarksNikita.add(1);

// listMarksNikita.add(2);

// Третий студент и его оценки

ArrayList<Integer> listMarksPeta = new ArrayList<Integer>();

// {10, 5, 8, 9};

// listMarksPeta.add(10);

// listMarksPeta.add(5);

// listMarksPeta.add(8);

// listMarksPeta.add(9);

// Для порверки на ввод валидных оценок

try

{

///// import java.util.Scanner; // Консольный ввод import java.util.Scanner;

int mark;

for(int i = 0; i < 4; i++)

{

System.out.print("Введите число (Оценка для Матвея): ");

mark = in.nextInt();

// Например если оценка отрицательная, а такой не может быть то я генерю свое исключение

if (mark < 0)

{

throw new Exception("EXCEPTION 1: Вы ввели отрицательное число Матвею"); // Для проверки ввести в одну из оценок отрицательное значение

}

listMarksMatvey.add(mark); // Оценки Матвея

System.out.print("Введите число (Оценка для Никиты): ");

mark = in.nextInt();

if (mark < 0)

{

throw new Exception("EXCEPTION 1: Вы ввели отрицательное число Никите");

}

listMarksNikita.add(mark);

System.out.print("Введите число (Оценка для Пети): ");

mark = in.nextInt();

if (mark < 0)

{

throw new Exception("EXCEPTION 1: Вы ввели отрицательное число Пете");

}

listMarksPeta.add(mark);

}

}

catch(InputMismatchException ex) // Например проверка чтобы словить ввести вещественное или символы

{

System.out.println("Вы не смогли заполнить оценки:\nEXCEPTION 2: " + ex.getMessage());

}

Student st1 = new Student("Matvey", 992, 3, listMarksMatvey); // Создал самого студента Матвей со списком оценок

Student st2 = new Student("Nikita", 992, 3, listMarksNikita);

Student st3 = new Student("Peta", 992, 3, listMarksPeta);

// Добавляю самих студентов НО не прокатит т.к пытаюсь запихнуть больше двух студентов. А размерность листа 2

try

{

listStudent.add(st1);

listStudent.add(st2);

listStudent.add(st3);

}

catch(ArrayIndexOutOfBoundsException ex) // Но странно почему-то пропускает

{

System.out.println("EXEPTION 3" + ex.getMessage());

}

// Тест функций

SortStudent(listStudent);

printStudentInList\_vibor\_pocursu(listStudent, 3); // Матвея не распечатает т.к он перевелся на next курс, распечатает Peta (ЭТО ЕСЛИ СМОТРЕТЬ НА ТЕ ДАННЫЕ КОТОРЫЕ ВВОДИЛ СРАЗУ) // Вот эти // {9, 8, 9, 10}; // {3, 2, 1, 2}; // {10, 5, 8, 9};

}

}

// student

**package** for\_exeption\_laba5;

**import** java.util.ArrayList;

**public** **class** Student {

**private** String name;

**private** **int** numGroup;

**private** **int** numKursa;

// Список оценок

**public** ArrayList<Integer> marks = **new** ArrayList<Integer>();

**public** Student(String name, **int** numGroup, **int** numKursa, ArrayList<Integer> marks)

{

**this**.setName(name);

**this**.setNumGroup(numGroup);

**this**.setNumKursa(numKursa);

**this**.marks = marks;

}

**public** String getName() {

**return** name;

}

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

**this**.name = name;

}

**public** **int** getNumKursa() {

**return** numKursa;

}

**public** **void** setNumKursa(**int** numKursa) {

**this**.numKursa = numKursa;

}

**public** **int** getNumGroup() {

**return** numGroup;

}

**public** **void** setNumGroup(**int** numGroup) {

**this**.numGroup = numGroup;

}

}

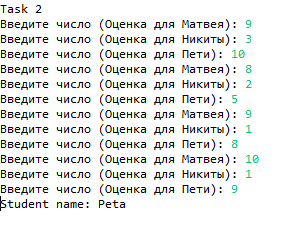


Рисунок 2

**// pakage core**

**// main**

**package** core;

**public** **class** class\_laba7 {

// (Для себя) throws – используется в сигнатуре методов для предупреждения, о том что метод может выбросить исключение.

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

// Для проверки -500, для проверки на исключение

state\_employee employee1 = **new** state\_employee("GoncharovMatveyNicholaevich", 5, 500, 100);

// Все работает но чтобы поломать и выдало исключения в классе state\_employee - метод return\_zarplatu оставил коменты и там можно проверить исключения

// или можно тут пермию сделать отрицательной

// employee1.setPremia(-23.5);

System.***out***.println("Итоговая зарплата: " + employee1.return\_zarplatu());

}

}

// employee

**package** core;

**public** **class** Employee {

**protected** String FIO;

**protected** **int** Dolzhnost = 0; // Например должность от 1 - 5, это будет влиять на оклад

**protected** **double** Oklad;

**public** Employee(){}

**public** Employee(String FIO, **int** Dolzhnost, **double** Oklad)

{

**this**.FIO = FIO;

**this**.Dolzhnost = Dolzhnost;

**try**

{

**if** (Oklad < 0)

{

**throw** **new** OkladException("Exception");

}

**else**

{

**this**.Oklad = Oklad;

}

}

**catch**(OkladException ex)

{

System.***out***.println("«Невозможно создать сотрудника – указан отрицательный оклад: " + Oklad + " " + ex.getMessage());

ex.printStackTrace();

}

}

**public** String getFIO() {

**return** FIO;

}

**public** **void** setFIO(String fIO) {

FIO = fIO;

}

**public** **int** getDolzhnost() {

**return** Dolzhnost;

}

**public** **void** setDolzhnost(**int** dolzhnost) {

Dolzhnost = dolzhnost;

}

**public** **double** getOklad() {

**return** Oklad;

}

**public** **void** setOklad(**double** oklad) {

Oklad = oklad;

}

**public** **double** return\_zarplatu()

{

System.***out***.println("Оклад составляет - " + getOklad());

**int** doklad = getDolzhnost();

System.***out***.println("Должность - " + doklad + " зарплата увеличивается на " + (10\* doklad) + "%");

System.***out***.println("Должность - " + doklad + " зарплата увеличивается на " + (10 \* doklad) + "%");

**double** okl = 0;

**try**

{

// Вызовется исключение т.к Dolzhnost = 0 например надо указать другое

// this.setDolzhnost(5); // раскоментить чтобы не было исключения

okl = Oklad + (Oklad \* (doklad \* 10) / 100); // + процентная ставка от должности

}

**catch**(ArithmeticException ex)

{

System.***out***.println("Exception " + ex.getMessage());

System.*exit*(1);

}

**catch**(Exception ex2)

{

System.***out***.println("Exception " + ex2.getMessage());

System.*exit*(1);

}

**return** okl;

}

}

// firm

**package** core;

**public** **class** firm {

**private** String NameFirm;

**public** String getNameFirm() {

**return** NameFirm;

}

**public** **void** setNameFirm(String nameFirm) {

NameFirm = nameFirm;

}

}

// contract\_employee

**package** core;

**public** **class** kontrakt\_employee **extends** Employee {

**public** kontrakt\_employee()

{

**super**();

}

**public** kontrakt\_employee(String FIO, **int** Dolzhnost, **double** Oklad)

{

**super**(FIO, Dolzhnost, Oklad);

}

**public** **double** return\_zarplatu()

{

System.***out***.println("Оклад составляет - " + getOklad());

**int** doklad = getDolzhnost();

System.***out***.println("Должность - " + doklad + " зарплата увеличивается на " + (10\* doklad) + "%");

System.***out***.println("Должность - " + doklad + " зарплата увеличивается на " + (10 \* doklad) + "%");

**double** okl = 0;

**try**

{

// Вызовется исключение т.к Dolzhnost = 0 например надо указать другое

// this.setDolzhnost(5); // раскоментить чтобы не было исключения

okl = Oklad + (Oklad \* (doklad \* 10) / 100); // + процентная ставка от должности

}

**catch**(ArithmeticException ex)

{

System.***out***.println("Exception " + ex.getMessage());

}

**catch**(Exception ex2)

{

System.***out***.println("Exception " + ex2.getMessage());

}

**return** okl;

}

}

// OkladException

**package** core;

**public** **class** OkladException **extends** Exception {

**public** OkladException(String string) {

// **TODO** Auto-generated constructor stub

**super**(string);

}

}

// otdel

**package** core;

**public** **class** Otdel {

**private** String OtdelName;

**private** **int** ColvoEmployeel; // Кол-во сотрудников

**public** String getOtdelName() {

**return** OtdelName;

}

**public** **void** setOtdelName(String otdelName) {

OtdelName = otdelName;

}

**public** **int** getColvoEmployeel() {

**return** ColvoEmployeel;

}

**public** **void** setColvoEmployeel(**int** colvoEmployeel) {

ColvoEmployeel = colvoEmployeel;

}

}

// premiyaException

**package** core;

**public** **class** PremiyaException **extends** Exception {

**public** PremiyaException(String string) {

// **TODO** Auto-generated constructor stub

**super**(string);

}

}

// state\_employee

**package** core;

**public** **class** state\_employee **extends** Employee{

**private** **double** Premia;

**public** state\_employee()

{

**super**();

}

**public** state\_employee(String FIO, **int** Dolzhnost, **double** Oklad,**double** Premia)

{

**super**(FIO, Dolzhnost, Oklad);

**this**.Premia = Premia;

}

**public** **double** getPremia() {

**return** Premia;

}

**public** **void** setPremia(**double** premia) {

Premia = premia;

}

**public** **double** return\_zarplatu()

{

System.***out***.println("Оклад составляет - " + getOklad());

**int** doklad = getDolzhnost(); // doklad - не оклад

**double** okl = 0;

**try**

{

**if** (getOklad() < 1)

{

**throw** **new** PremiyaException("Ваш оклад < 0"); // Генерю исключение

}

**else**

{

**if** (getPremia() < 0)

{

**throw** **new** PremiyaException("Ваша премия < 0"); // Генерю исключение

}

**else**

{

System.***out***.println("Должность - " + doklad + " зарплата увеличивается на " + (10\* doklad) + "%");

System.***out***.println("Премея составляет = " + getPremia());

okl = Oklad + (Oklad \* (doklad \* 10) / 100) + getPremia(); // + процентная ставка от должности

}

}

}

**catch**(ArithmeticException ex)

{

System.***out***.println("Exception " + ex.getMessage());

ex.printStackTrace();

System.*exit*(1);

}

**catch**(PremiyaException ex2)

{

System.***out***.println("Exception " + ex2.getMessage());

ex2.printStackTrace();

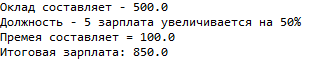
System.*exit*(1);

}

**return** okl;

}

}



Рисунок