|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,**

**обработки и интерпретации больших данных.**

**Отчет**

|  |
| --- |
| **по лабораторной работе № 5** |
|  |
| **Вариант 16** |

**Название:**

Исключения и файлы

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ6-23М | |  |  | В.И. Тарасов |
|  | | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  | |  |  |  |  |
| Преподаватель | |  |  |  | П.В. Степанов |
|  | |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2022

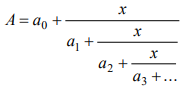
**Цель работы** –получение навыков обработки исключительных ситуаций в языке программирования Java. Изучение основ работы с файлами и файловой системой.

**Ход работы**

**Задание 1:**

Выполнить задания на основе варианта 1 лабораторной работы 3, контролируя состояние потоков ввода/вывода. При возникновении ошибок, связанных с корректностью выполнения математических операций, генерировать и обрабатывать исключительные ситуации. Предусмотреть обработку исключений, возникающих при нехватке памяти, отсутствии требуемой записи (объекта) в файле, недопустимом значении поля и т.д.

1. Определить класс Цепная дробь



Определить методы сложения, вычитания, умножения, деления. Вычислить значение для заданного n, x, a[n].

1. Определить класс Дробь в виде пары (m,n). Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения и деления дробей. Объявить массив из k дробей, ввести/вывести значения для массива дробей. Создать массив объектов и передать его в метод, который изменяет каждый элемент массива с четным индексом путем добавления следующего за ним элемента массива.

Листинг 1 – Код подзадачи 1

package src.lab5\_1;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.InputMismatchException;

import java.util.Scanner;

public class ContinuedFraction {

public int n;

public int x;

public ArrayList<Integer> a;

public double value;

public ContinuedFraction(int n, int x, ArrayList<Integer> a){

this.n = n;

this.x = x;

this.a = a;

this.value = 0.0;

calc();

}

public double sum(ContinuedFraction term){

return this.value + term.value;

}

public double sub (ContinuedFraction term){

return this.value - term.value;

}

public double mul(ContinuedFraction term){

return this.value \* term.value;

}

public double div(ContinuedFraction term){

return this.value / term.value;

}

public void calc(){

double temp = 0.0;

for (int i = n-1; i >= 1; i--) {

temp = x / (temp + this.a.get(i));

}

value = temp;

}

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

System.out.println("Введите n:");

Scanner console = new Scanner(System.in);

try {

int n = console.nextInt();

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

System.out.println("Введите n целых положительных чисел:");

for (int i = 0; i < n; i++){

int a = console.nextInt();

if (a <= 0) throw new ArithmeticException("Число должно быть положительным");

arr.add(a);

}

System.out.println("Введите x:");

int x = console.nextInt();

ContinuedFraction frac = new ContinuedFraction(n, x, arr);

ContinuedFraction frac2 = new ContinuedFraction(3, 2, new ArrayList<Integer>(Arrays.asList(1, 2, 3)));

System.out.println("Значение 1: ");

System.out.println(frac.value);

System.out.println("Значение 2: ");

System.out.println(frac2.value);

System.out.println("Сложение: ");

System.out.println(frac.sum(frac2));

System.out.println("Вычитание: ");

System.out.println(frac.sub(frac2));

System.out.println("Умножение: ");

System.out.println(frac.mul(frac2));

System.out.println("Деление: ");

System.out.println(frac.div(frac2));

} catch (InputMismatchException e) {

System.out.println("Неверный формат ввода");

} catch (OutOfMemoryError e) {

System.out.println("Недостаточно памяти");

} catch (ArithmeticException e){

System.out.println(e.getMessage());

} catch (NegativeArraySizeException e) {

System.out.println("Размерность массива должна быть положительная");

} finally {

console.close();

}

}

}

Результат обработки исключительной ситуации представлен на рисунке 1.

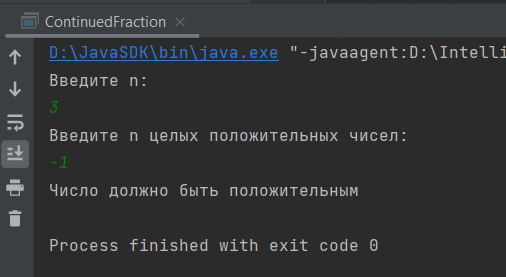


Рисунок 1 – Результат обработки исключения

Листинг 2 – Код класса Fraction

package src.lab5\_1;

public class Fraction {

int m, n;

public Fraction() {

m = 0;

n = 1;

}

public Fraction(int m) {

this.m = m;

n = 1;

}

public Fraction(int m, int n) {

this.m = m;

this.n = n;

if (n == 0) throw new ArithmeticException("Знаменатель равен 0");

}

public Fraction(Fraction a) {

m = a.m;

n = a.n;

if (n == 0) throw new ArithmeticException("Знаменатель равен 0");

}

public Fraction Add(Fraction term) {

Fraction sum = new Fraction();

sum.n = this.n \* term.n;

sum.m = m \* term.n + term.m \* n;

return sum;

}

public Fraction Multiply(Fraction term) {

return new Fraction(

m \* term.m,

n \* term.n

);

}

public Fraction Invert() {

return new Fraction(-m, n);

}

public Fraction Subtract(Fraction term) {

Fraction inverted\_term = new Fraction(term.Invert());

if (inverted\_term.n == 0) throw new ArithmeticException("Знаменатель равен 0");

return this.Add(inverted\_term);

}

public Fraction Switch() {

return new Fraction(n, m);

}

public Fraction Divide(Fraction term) {

Fraction switched\_multiplicator = term.Switch();

return this.Multiply(switched\_multiplicator);

}

public String toString() {

return m + "/" + n;

}

}

Листинг 3 – Код основной программы подзадачи 2

package src.lab5\_1;

import src.lab5\_1.Fraction;

import java.util.InputMismatchException;

import java.util.Scanner;

public class Program {

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

System.out.println("Введите k:");

Scanner console = new Scanner(System.in);

try {

int k = console.nextInt();

Fraction[] fractions = new Fraction[k];

for (int i = 0; i < k; i++) {

System.out.println("Введите m для дроби " + (i + 1) + ":");

int m = console.nextInt();

System.out.println("Введите n для дроби " + (i + 1) + ":");

int n = console.nextInt();

fractions[i] = new Fraction(m, n);

}

PrintArray(fractions);

for (int i = 0; i < k - 1; i += 2) {

fractions[i] = fractions[i].Add(fractions[i + 1]);

}

PrintArray(fractions);

} catch (InputMismatchException e) {

System.out.println("Неверный формат ввода");

} catch (NegativeArraySizeException e) {

System.out.println("Размерность массива должна быть положительная");

} catch (OutOfMemoryError e) {

System.out.println("Недостаточно памяти");

} catch (ArithmeticException e) {

System.out.println(e.getMessage());

} finally {

console.close();

}

}

private static void PrintArray(Fraction[] array) {

System.out.print("[");

for (int i = 0; i < array.length; i++) {

System.out.print(array[i].toString());

if (i != array.length - 1) System.out.print(", ");

}

System.out.print("]\n");

}

}

Результат обработки исключения для подзадачи 2 представлен на рисунке 2.

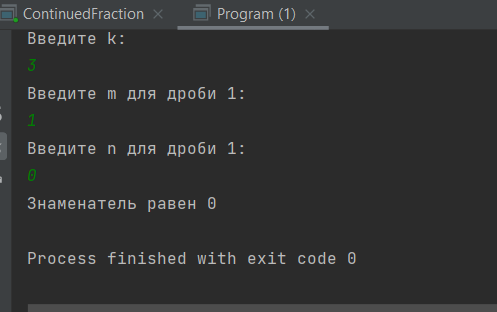


Рисунок 2 – Результат обработки исключения

**Задание 2:**

Выполнить задания из варианта 2 лабораторной работы 3, реализуя собственные обработчики исключений и исключения ввода/вывода.

1. House: id, Номер квартиры, Площадь, Этаж, Количество комнат, Улица, Тип здания, Срок эксплуатации. Создать массив объектов. Вывести: a) список квартир, имеющих заданное число комнат; b) список квартир, имеющих заданное число комнат и расположенных на этаже, который находится в заданном промежутке; c) список квартир, имеющих площадь, превосходящую заданную.
2. Phone: id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Дебет, Кредит, Время городских и междугородных разговоров. Создать массив объектов. Вывести: a) сведения об абонентах, у которых время внутригородских разговоров превышает заданное; b) сведения об абонентах, которые пользовались междугородной связью; c) сведения об абонентах в алфавитном порядке.

Листинг 4 – Код класса House с обработчиками исключений

package src.lab5\_2;

public class House {

int id;

int number;

int square;

int floor;

int room\_count;

String street, building\_type;

int lifetime;

public void setId(int id) {

this.id = id;

}

public void setBuilding\_type(String building\_type) {

this.building\_type = building\_type;

}

public void setFloor(int floor) throws Exception {

if (floor <= 0) throw new Exception("Количество этажей должно быть не менее 1");

this.floor = floor;

}

public void setLifetime(int lifetime) throws Exception {

if (lifetime <= 0) throw new Exception("Время эксплуатации должно быть положительным");

this.lifetime = lifetime;

}

public void setNumber(int number) {

this.number = number;

}

public void setRoom\_count(int room\_count) throws Exception {

if (room\_count <= 0) throw new Exception("Количество комнат должно быть положительным");

this.room\_count = room\_count;

}

public void setSquare(int square) throws Exception {

if (square <= 0) throw new Exception("Площадь должна быть положительной");

this.square = square;

}

public void setStreet(String street) {

this.street = street;

}

public int getId() {

return id;

}

public double getSquare() {

return square;

}

public int getFloor() {

return floor;

}

public int getLifetime() {

return lifetime;

}

public int getNumber() {

return number;

}

public int getRoom\_count() {

return room\_count;

}

public String getBuilding\_type() {

return building\_type;

}

public String getStreet() {

return street;

}

@Override

public String toString() {

return "House{" +

"id=" + id +

", number=" + number +

", square=" + square +

", floor=" + floor +

", room\_count=" + room\_count +

", street='" + street + '\'' +

", building\_type='" + building\_type + '\'' +

", lifetime=" + lifetime +

'}';

}

}

Результаты обработки исключительной ситуации представлен на рисунке 2.

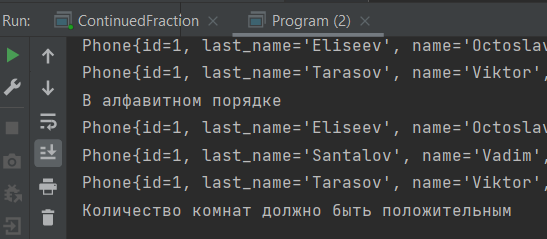


Рисунок 3 – Результат обработки исключительной ситуации

Листинг 5 – Код класса Phone с обработчиками исключений

package src.lab5\_2;

public class Phone {

int id;

String last\_name, name, patronymic, address;

long card\_number;

double debit, credit;

int intercity\_calls, intracity\_calls;

public int getId() {

return id;

}

public String getLast\_name() {

return last\_name;

}

public String getName() {

return name;

}

public String getPatronymic() {

return patronymic;

}

public String getAddress() {

return address;

}

public long getCard\_number() {

return card\_number;

}

public double getDebit() {

return debit;

}

public double getCredit() {

return credit;

}

public int getIntercity\_calls() {

return intercity\_calls;

}

public int getIntracity\_calls() {

return intracity\_calls;

}

public void setAddress(String address) throws Exception {

if (address.isEmpty()) throw new Exception("Некорректная строка");

this.address = address;

}

public void setCard\_number(long card\_number) throws Exception {

int length = (int)(Math.log10(card\_number)+1);

if (length != 16) throw new Exception("Некорректный формат номера карты");

this.card\_number = card\_number;

}

public void setCredit(double credit) {

this.credit = credit;

}

public void setDebit(double debit) {

this.debit = debit;

}

public void setId(int id) {

this.id = id;

}

public void setIntercity\_calls(int intercity\_calls) throws Exception {

if (intercity\_calls < 0) throw new Exception("Длительность звонков не может быть отрицательной");

this.intercity\_calls = intercity\_calls;

}

public void setIntracity\_calls(int intracity\_calls) throws Exception {

if (intracity\_calls < 0) throw new Exception("Длительность звонков не может быть отрицательной");

this.intracity\_calls = intracity\_calls;

}

public void setLast\_name(String last\_name) throws Exception {

if (last\_name.isEmpty()) throw new Exception("Некорректная строка");

this.last\_name = last\_name;

}

public void setName(String name) throws Exception {

if (name.isEmpty()) throw new Exception("Некорректная строка");

this.name = name;

}

public void setPatronymic(String patronymic) throws Exception {

if (patronymic.isEmpty()) throw new Exception("Некорректная строка");

this.patronymic = patronymic;

}

@Override

public String toString() {

return "Phone{" +

"id=" + id +

", last\_name='" + last\_name + '\'' +

", name='" + name + '\'' +

", patronymic='" + patronymic + '\'' +

", address='" + address + '\'' +

", card\_number=" + card\_number +

", debit=" + debit +

", credit=" + credit +

", intercity\_calls=" + intercity\_calls +

", intracity\_calls=" + intracity\_calls +

'}';

}

}

Результаты обработки исключения в подзадаче 2 представлен на рисунке 4.

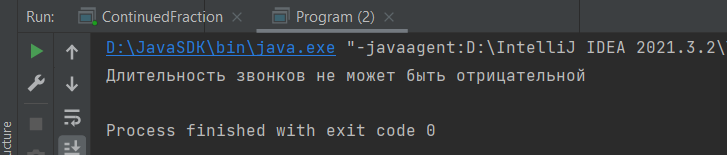


Рисунок 4 – Обработка исключительной ситуации в подзадаче 2

**Задание 3:**

В следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия. При этом могут рассматриваться два варианта:

1. каждая строка состоит из одного слова;
2. каждая строка состоит из нескольких слов.

Имена входного и выходного файлов, а также абсолютный путь к ним могут быть введены как параметры командной строки или храниться в файле.

1. В каждой строке стихотворения Анны Ахматовой подсчитать частоту повторяемости каждого слова из заданного списка и вывести эти слова в порядке возрастания частоты повторяемости.
2. В каждом слове стихотворения Николая Заболоцкого заменить первую букву слова на прописную.

Листинг 6 – Код задания 3

package src.lab5\_3;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.nio.charset.StandardCharsets;

import java.util.\*;

public class Program {

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

try {

TreeMap<String, Integer> wordsList = new TreeMap<>();

TreeMap<Integer, String> sorted = new TreeMap<>();

wordsList.put("мне", 0);

wordsList.put("как", 0);

wordsList.put("у", 0);

System.out.println("Стихотворение Ахматовой");

File poem = new File("src/lab5\_3/data/ahm.txt");

Scanner reader = new Scanner(poem, StandardCharsets.UTF\_8);

while (reader.hasNextLine()) {

String data = reader.nextLine();

String[] words = data.split("\\s");

for(String w:words){

if (wordsList.containsKey(w)) {

int count = wordsList.get(w);

wordsList.put(w, count+1);

}

}

System.out.println();

}

System.out.print("\n");

System.out.println("Стихотворение обработано");

for(Map.Entry<String,Integer> entry : wordsList.entrySet()) {

sorted.put(entry.getValue(), entry.getKey());

}

reader.close();

for(Map.Entry<Integer,String> entry : sorted.entrySet()) {

System.out.println(entry.getValue() + ' ' + entry.getKey());

}

} catch (FileNotFoundException e) {

System.out.println("Файл не найден");

e.printStackTrace();

} catch (IOException e) {

System.out.println("Произошла ошибка ввода/вывода");

e.printStackTrace();

}

try {

System.out.println("Стихотворение Н. Заболоцкого");

File poem = new File("src/lab5\_3/data/poem\_orig.txt");

String output\_poem = "src/lab5\_3/data/poem\_fin.txt";

FileWriter writer = new FileWriter(output\_poem, StandardCharsets.UTF\_8);

Scanner reader = new Scanner(poem, StandardCharsets.UTF\_8);

while (reader.hasNextLine()) {

System.out.print(".");

String data = reader.nextLine();

writer.write(capitalizeWords(data));

writer.write("\n");

}

System.out.print("\n");

System.out.println("Стихотворение обработано");

writer.close();

reader.close();

} catch (FileNotFoundException e) {

System.out.println("Файл не найден");

e.printStackTrace();

} catch (IOException e) {

System.out.println("Произошла ошибка ввода/вывода");

e.printStackTrace();

}

}

static String capitalizeWords(String str) {

if (str.isEmpty()) return "";

String[] words = str.split("\\s");

StringBuilder capitalizeWord = new StringBuilder();

for(String w:words){

String first = w.substring(0,1);

String others = w.substring(1);

capitalizeWord.append(first.toUpperCase()).append(others).append(" ");

}

return capitalizeWord.toString().trim();

}

}

Результаты выполнения подзадачи 1 представлены на рисунке 5.

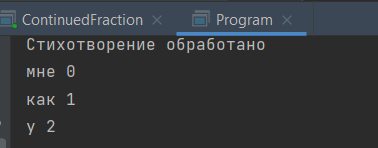


Рисунок 5 – Результат подсчета частоты слов

Результаты выполнения подзадачи 2 представлены на рисунке 6.

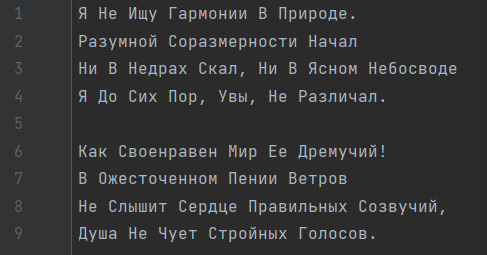


Рисунок 6 – Результирующий файл

**Задание 4:**

1. Из файла удалить все слова, содержащие от трех до пяти символов, но при этом из каждой строки должно быть удалено только максимальное четное количество таких слов.
2. Прочитать текст Java-программы и удалить из него все “лишние” пробелы и табуляции, оставив только необходимые для разделения операторов.

Листинг 7 – Код задания 4

package src.lab5\_4;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.nio.charset.StandardCharsets;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.util.ArrayList;

import java.util.List;

import java.util.stream.Stream;

public class Program {

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

try {

Path inputPath = Paths.get("src/lab5\_4/data/Prog.java");

List<String> strings = Files.readAllLines(inputPath, StandardCharsets.UTF\_8);

ArrayList<String> outList = new ArrayList<>();

for (String s : strings) {

String changed = s.trim().replaceAll(" +", " ");

outList.add(changed);

}

Files.createDirectories(Paths.get("src/lab5\_4/output"));

Path outPath = Paths.get("src/lab5\_4/output/Hello\_linted.txt");

Files.write(outPath, outList, StandardCharsets.UTF\_8);

} catch (FileNotFoundException e) {

System.out.println("Файл не найден");

e.printStackTrace();

} catch (IOException e) {

System.out.println("Произошла ошибка I/O");

e.printStackTrace();

}

try {

Path inputPath = Paths.get("src/lab5\_4/data/Prog.java");

ArrayList<String> outList = new ArrayList<>();

Stream<String> stream = Files.lines(inputPath);

stream.forEach(line -> deleteWords(line, outList));

Files.createDirectories(Paths.get("src/lab5\_4/output"));

Path outPath = Paths.get("src/lab5\_4/output/Hello\_trimmed.txt");

Files.write(outPath, outList, StandardCharsets.UTF\_8);

} catch (FileNotFoundException e) {

System.out.println("Файл не найден");

e.printStackTrace();

} catch (IOException e) {

System.out.println("Произошла ошибка I/O");

e.printStackTrace();

}

}

static void deleteWords(String str, ArrayList<String> outList) {

int deleted = 0;

int max\_deleted = 0;

str = str.trim().replaceAll(" +", " ");

String result = "";

String[] words = str.split("\\s");

if (words.length % 2 == 0) {

max\_deleted = words.length;

}

else {

max\_deleted = words.length - 1;

}

for(String w:words){

if (deleted == max\_deleted) break;

if (w.length() >= 3 && w.length() <= 5) {

deleted += 1;

}

else {

result = result.concat(w + ' ');

}

}

outList.add(result);

}

}

Результаты выполнения подзадачи 1 представлены на рисунке 7.

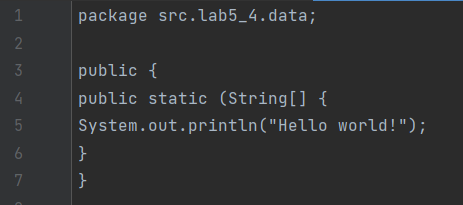


Рисунок 7 – Результирующий файл подзадачи 1

Результаты выполнения подзадачи 2 представлены на рисунке 8.

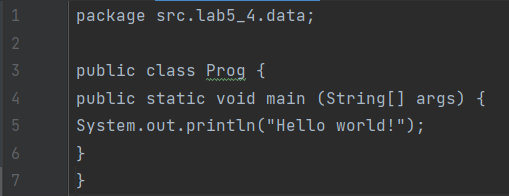


Рисунок 8 – Результирующий файл подзадачи 2

**Местоположение проектных файлов** –файлы проекта расположены в репозитории GitHub по следующим адресам:

<https://github.com/Fregatty/BigDataLanguages/tree/main/src/lab5_1>

<https://github.com/Fregatty/BigDataLanguages/tree/main/src/lab5_2>

<https://github.com/Fregatty/BigDataLanguages/tree/main/src/lab5_3>

<https://github.com/Fregatty/BigDataLanguages/tree/main/src/lab5_4>

**Вывод** – в результате выполнения лабораторной работы были получены навыки обработки исключительных ситуаций, а также работы с файлами в языке Java.