|  |  |
| --- | --- |
| **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 **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений**

**Отчет**

**по лабораторной работе № 5**

**Название:** Исключения. Файлы

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

Студент гр. ИУ6-23М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_** Г. Е. Горский **\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**П.В. Степанов **\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2023

**Цель:** ознакомиться с базовыми принципами языка Java для работы с большими данными

**Вариант 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.

**Решение:**

import java.util.Random;  
  
public class Drob {  
 private float m;  
 private float n;  
  
 Drob(){  
 this.m = 2; //числитель  
 this.n = 1; // знаменатель  
 }  
 Drob(int m\_, int n\_){  
 this.m=m\_;  
 this.n=n\_;  
 }  
 Drob(int m\_){  
 this.m=m\_;  
 this.n = 1;  
 }  
  
 public float getM() {  
 return m;  
 }  
  
 public float getN() {  
 return n;  
 }  
  
 public void setM(float m) {  
 this.m = m;  
 }  
  
 public void setN(float n) {  
 this.n = n;  
 }  
  
 @Override  
 public String toString() {  
 return "m=" + m +  
 ", n=" + n;  
 }  
  
/\* 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. \*/  
 public static void main(String[] args) {  
 Random rnd = new Random();  
 Drob dr1 = new Drob(rnd.nextInt(1,5),rnd.nextInt(1,5) );  
 Drob dr2 = new Drob(rnd.nextInt(1,5));  
 System.out.println(dr1.toString() + "\n" + dr2.toString());  
 System.out.println((Plus(dr1, dr2)));  
 System.out.println((Minus(dr1, dr2)));  
 System.out.println((Multiply(dr1, dr2)));  
 System.out.println((Delete(dr1, dr2)));  
  
  
  
 int k=rnd.nextInt(2,7);  
 Drob[] dr3 = new Drob[k];  
 try {  
 for (int i = 0; i<k; i++){  
 dr3[i] = new Drob(rnd.nextInt(1,5),rnd.nextInt(1,5) );  
 }  
 } catch (OutOfMemoryError e){  
 throw new OutOfMemoryError("виртуальная машина Java не может выделить объект из-за нехватки памяти");  
 }  
  
 try {  
 for (int i = 0; i<k; i++){  
 // dr3[i] = new Drob(rnd.nextFloat(1,5),rnd.nextFloat(1,5) );  
 System.out.println(dr3[i].toString());  
 }  
 } catch (IndexOutOfBoundsException e){  
 throw new IndexOutOfBoundsException("Выход за пределы масисива " + k);  
 }  
  
 Array\_Drob(k, dr3);  
 System.out.println("new array");  
 for (int i = 0; i<k; i++) {  
 System.out.println(dr3[i].toString());  
 }  
 }  
 public static Drob[] Array\_Drob(int k, Drob[] a){  
  
 for (int i=0; i<k-1; i++){  
 if(i%2==0){  
 float tmp1=a[i].getM() + a[i+1].getM();  
 float tmp2=a[i].getN() + a[i+1].getN();  
 a[i].setM(tmp1);  
 a[i].setN(tmp2);  
 }  
 }  
 return a;  
 }  
 public static float Change(Drob a) throws ArithmeticException{  
 float first\_ch = a.getM();  
 float first\_zn = a.getN();  
 float tmp1 =0;  
 // first\_zn = 0;  
  
 tmp1 = first\_ch / first\_zn;  
 if (tmp1 == Float.POSITIVE\_INFINITY || tmp1 == Float.NEGATIVE\_INFINITY) {  
 throw new ArithmeticException("Деление на ноль");  
 }  
 return tmp1;  
 }  
 public static float Plus(Drob a, Drob b){  
 float tmp1 =Change(a);  
 float tmp2 = Change(b);  
 return tmp1 + tmp2;  
 }  
 public static float Minus(Drob a, Drob b){  
 float tmp1 =Change(a);  
 float tmp2 = Change(b);  
 return tmp1 - tmp2;  
 }  
 public static float Multiply(Drob a, Drob b){  
 float tmp1 =Change(a);  
 float tmp2 = Change(b);  
 return tmp1 \* tmp2;  
 }  
 public static float Delete(Drob a, Drob b) throws ArithmeticException {  
 float tmp1 =Change(a);  
 float tmp2 = Change(b);  
 if (tmp2 != 0) {  
 return tmp1 / tmp2;  
 }  
 else {  
 throw new ArithmeticException("Деление на ноль");  
 }  
 }  
}

Результат выполнения представлен на рисунке 1.

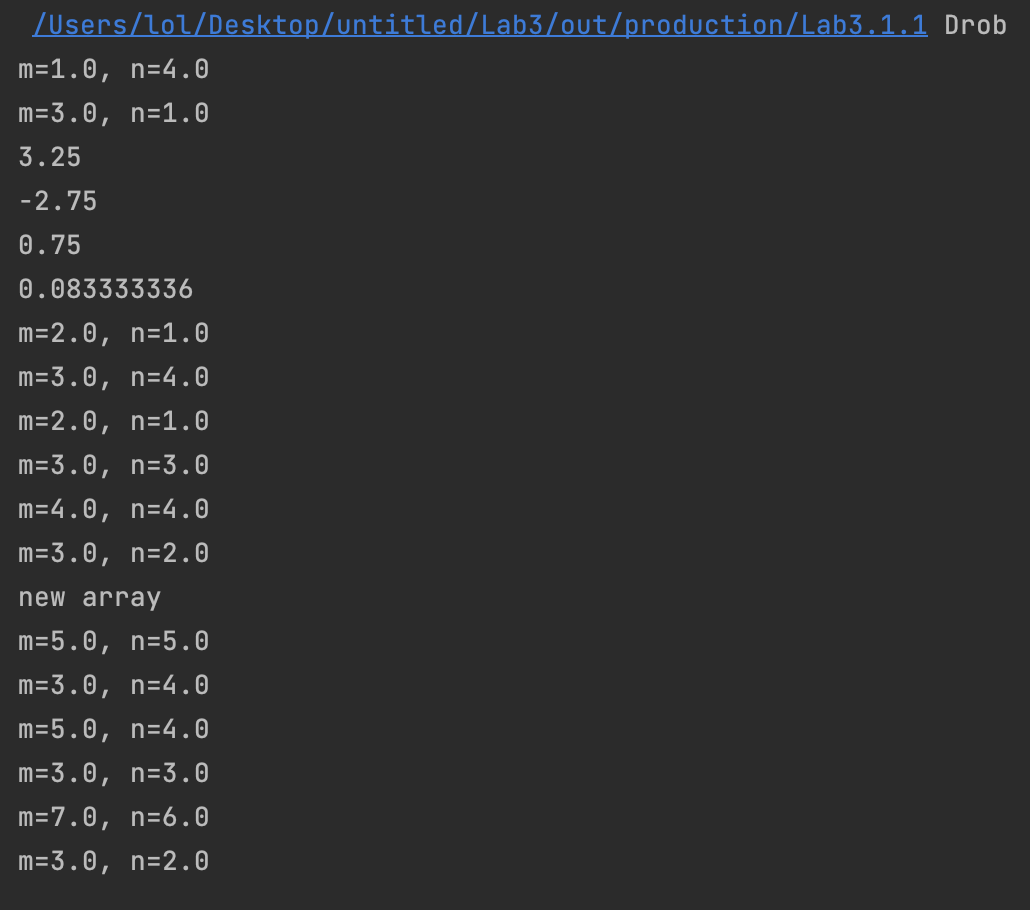
****

Рисунок 1 — пример выполнения

import java.util.Random;  
import java.util.Vector;  
  
public class Complex {  
 private float re;  
 private float im;  
  
 public Complex(float real, float imag) {  
 this.re = real;  
 this.im = imag;  
 }  
 public Complex(float real) {  
 this.re = real;  
 this.im = 0;  
 }  
 public Complex() {  
 this.re = 2;  
 this.im = 1;  
 }  
 @Override  
 public String toString() {  
 if (im == 0) return re + "";  
 if (re == 0) return im + "i";  
 if (im < 0) return re + " - " + (-im) + "i";  
 return re + " + " + im + "i";  
 }  
 public Complex Plus(Complex b) {  
 Complex a = this;  
 float real = a.re + b.re;  
 float imag = a.im + b.im;  
 return new Complex(real, imag);  
 }  
 public Complex Minus(Complex b) {  
 Complex a = this;  
 float real = a.re - b.re;  
 float imag = a.im - b.im;  
 return new Complex(real, imag);  
 }  
 public Complex Multiply(Complex b) {  
 Complex a = this;  
 float real = a.re \* b.re - a.im \* b.im;  
 float imag = a.re \* b.im + a.im \* b.re;  
 return new Complex(real, imag);  
 }  
  
 public Complex reciprocal() throws ArithmeticException{  
 float scale = re \* re + im \* im;  
 if (scale == Float.POSITIVE\_INFINITY || scale == Float.NEGATIVE\_INFINITY) {  
 throw new ArithmeticException("Деление на ноль");  
 }  
 return new Complex(re / scale, -im / scale);  
 }  
 public Complex Delete(Complex b) {  
 Complex a = this;  
 return a.Multiply(b.reciprocal());  
 }  
 public static Vector<Complex> Vect(Vector<Complex> v1, Vector<Complex> v2){  
  
 for (int i=0; i< v1.capacity(); i++){  
 v1.set(i,v1.get(i).Plus(v2.get(i)));  
 }  
  
 return v1;  
 }  
/\* Определить класс Комплекс. Класс должен содержать несколько конструкторов.  
 Реализовать методы для сложения, вычитания, умножения, деления, присваивания комплексных чисел.  
 Создать два вектора размерности n из комплексных координат.  
 Передать их в метод, который выполнит их сложение. \*/  
 public static void main(String[] args) {  
 Random rnd = new Random();  
 Complex a = new Complex(rnd.nextInt(-5, 5), rnd.nextInt(-5, 5));  
 Complex b = new Complex(rnd.nextInt(-5, 5), rnd.nextInt(-5, 5));  
  
 System.out.println("a = " + a);  
 System.out.println("b = " + b);  
 System.out.println("b + a = " + b.Plus(a));  
 System.out.println("a - b = " + a.Minus(b));  
 System.out.println("a \* b = " + a.Multiply(b));  
 System.out.println("a / b = " + a.Delete(b));  
  
 int n=rnd.nextInt(2,7);  
 Vector<Complex> v1 = new Vector<Complex>(n);  
 Vector<Complex> v2 = new Vector<Complex>(n);  
 try {  
 for (int i = 0; i<n; i++) {  
 v1.add(new Complex(rnd.nextInt(-5, 5), rnd.nextInt(-5, 5)));  
 v2.add(new Complex(rnd.nextInt(-5, 5), rnd.nextInt(-5, 5)));  
 }  
 }catch (OutOfMemoryError e){  
 throw new OutOfMemoryError("виртуальная машина Java не может выделить объект из-за нехватки памяти");  
 }catch (Exception e){  
 System.out.println("Что-то пошло не так");  
 }  
 System.out.println(v1.toString());  
 System.out.println(v2.toString());  
 Vect(v1,v2);  
 System.out.println("new Vector");  
 System.out.println(v1.toString());  
 }

}

Результат выполнения представлен на рисунке 2.

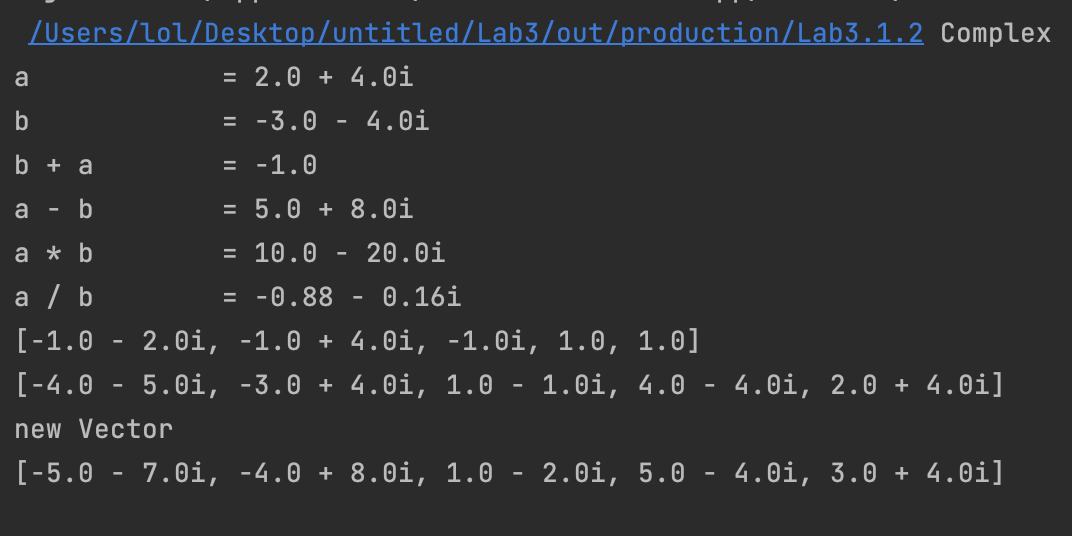


Рисунок 2 — пример выполнения

**Вариант 2:**

1. 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.
2. Car: id, Марка, Модель, Год выпуска, Цвет, Цена, Регистрационный номер. Создать массив объектов. Вывести: a) список автомобилей заданной марки; b) список автомобилей заданной модели, которые эксплуатируются больше n лет; c) список автомобилей заданного года выпуска, цена которых больше указанной.

**Решение:**

public class ExcClass extends Exception {  
 private String someString;  
 public ExcClass(String string) {  
 this.someString = string;  
 // System.out.println("Exception ExcClass");  
 }  
 public void myOwnExceptionMsg() {  
 System.err.println("недостаточно элементов массива: " + someString);  
 }  
}

public static void main(String[] args) throws IOException {  
 Random rnd = new Random();  
 Scanner myObj = new Scanner(System.in);  
 //int k=rnd.nextInt(2,4);  
 int k=0;  
 System.out.println("k=");  
 try {  
 k =myObj.nextInt();  
 if (k<=0){  
 throw new ExcClass("k должно быть больше 0");  
 }  
 }catch (ExcClass e){  
 e.myOwnExceptionMsg();  
 }  
 Phone[] ph = new Phone[k];  
 for (int i = 0; i<k; i++){  
 ph[i] = new Phone();  
 ph[i].ID=i;  
 System.out.println(" new user");  
 myObj.nextLine();  
 System.out.println("Input surname: ");  
 ph[i].surname = myObj.nextLine();  
 System.out.println("Input name: ");  
 ph[i].name = myObj.nextLine();  
 System.out.println("Input otch: ");  
 ph[i].otch = myObj.nextLine();  
 System.out.println("Input address: ");  
 ph[i].address = myObj.nextLine();  
 System.out.println("Input number credit card: ");  
 try {  
 ph[i].debet = myObj.nextInt();  
  
 // ph[i].number\_credit = myObj.nextInt();  
 System.out.println("Input debet: ");  
 ph[i].debet = myObj.nextInt();  
 System.out.println("Input credit: ");  
 ph[i].credit = myObj.nextInt();  
 System.out.println("Input debet: ");  
 ph[i].debet = myObj.nextInt();  
 System.out.println("Input localtime: ");  
 ph[i].localtime = myObj.nextInt();  
 System.out.println("Input globaltime: ");  
 ph[i].globaltime = myObj.nextInt();  
 }catch (InputMismatchException e){  
 throw new InputMismatchException("NAN " + myObj.next());  
 }  
 }  
 System.out.println("-------------------------------------------");  
 for (int i = 0; i<k; i++){  
 System.out.println(ph[i].toString());  
 }  
 System.out.println("-------------------------------------------");  
 for (int i = 0; i<k; i++){  
 ph[i].Show();  
 }  
 System.out.println("-------------------------------------------");  
 Arrays.sort(ph, (a, b) -> a.name.compareTo(b.name));  
 for (int i = 0; i<k; i++){  
 System.out.println(ph[i].toString());  
 }  
 }  
}

Результат выполнения представлен на рисунке 3.

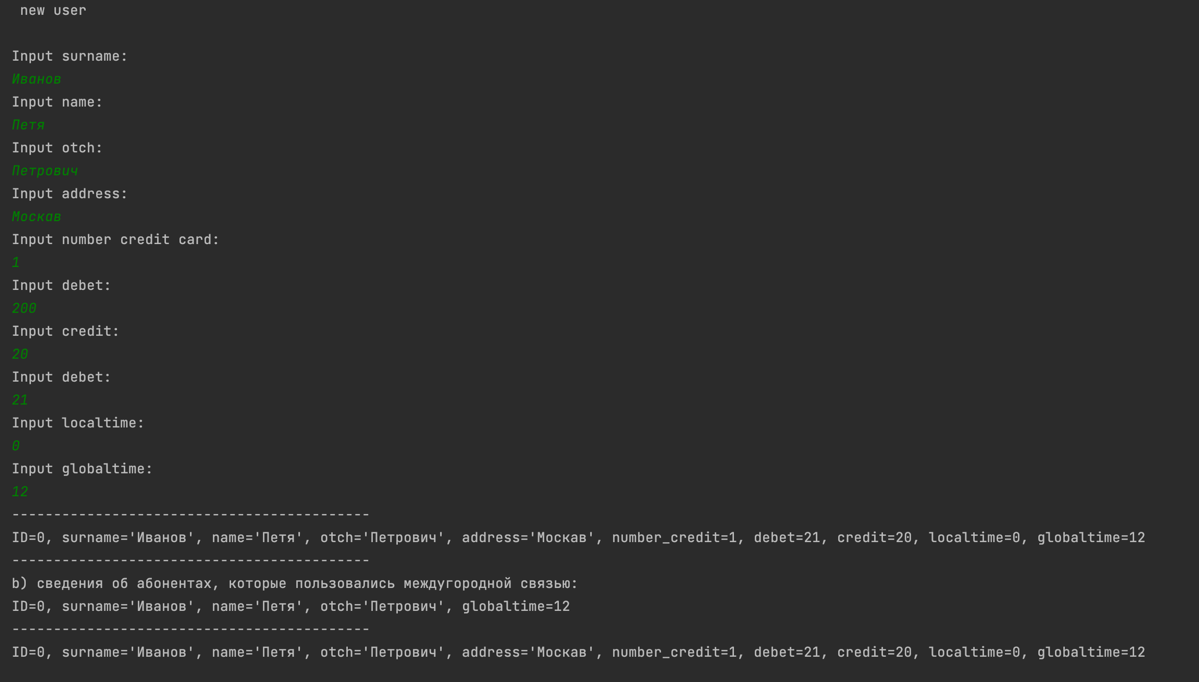


Рисунок 3 — пример выполнения

public static void main(String[] args) {  
 Random rnd = new Random();  
 Scanner myObj = new Scanner(System.in);  
 //int k=rnd.nextInt(2,4);  
 //int k =2;  
 int k=0;  
 System.out.println("k=");  
 try {  
 k =myObj.nextInt();  
 if (k<=0){  
 throw new ExcClass("k должно быть больше 0");  
 }  
 }catch (ExcClass e){  
 e.myOwnExceptionMsg();  
 }  
 Car[] ph = new Car[k];  
 for (int i = 0; i<k; i++){  
 ph[i] = new Car();  
 ph[i].ID=i;  
 System.out.println(" new car ");  
 System.out.println("Input mark: ");  
 ph[i].mark = myObj.nextLine() + myObj.nextLine();  
 System.out.println("Input model: ");  
 ph[i].model = myObj.nextLine() ;  
 System.out.println("Input color: ");  
 ph[i].color = myObj.nextLine() ;  
 System.out.println("Input reg number: ");  
 ph[i].reg\_num = myObj.nextLine() ;  
 try {  
 System.out.println("Input year: ");  
 ph[i].year = myObj.nextInt();  
 System.out.println("Input price: ");  
 ph[i].price = myObj.nextInt();  
 }catch (InputMismatchException e){  
 throw new InputMismatchException("NAN: " + myObj.next());  
 }  
  
 myObj.nextLine();  
 }  
  
 String mark = "BMW";  
 String model = "3";  
 int n = 7;  
 int m = 2010;  
 int price = 100;  
 System.out.println("-------------------------------------------");  
 for (int i = 0; i<k; i++){  
 System.out.println(ph[i].toString());  
 }  
 System.out.println("-------------------------------------------");  
 for (int i = 0; i<k; i++){  
 ph[i].Show(mark, model, n, m, price);  
 }  
 }  
}

Результат выполнения представлен на рисунке 4.

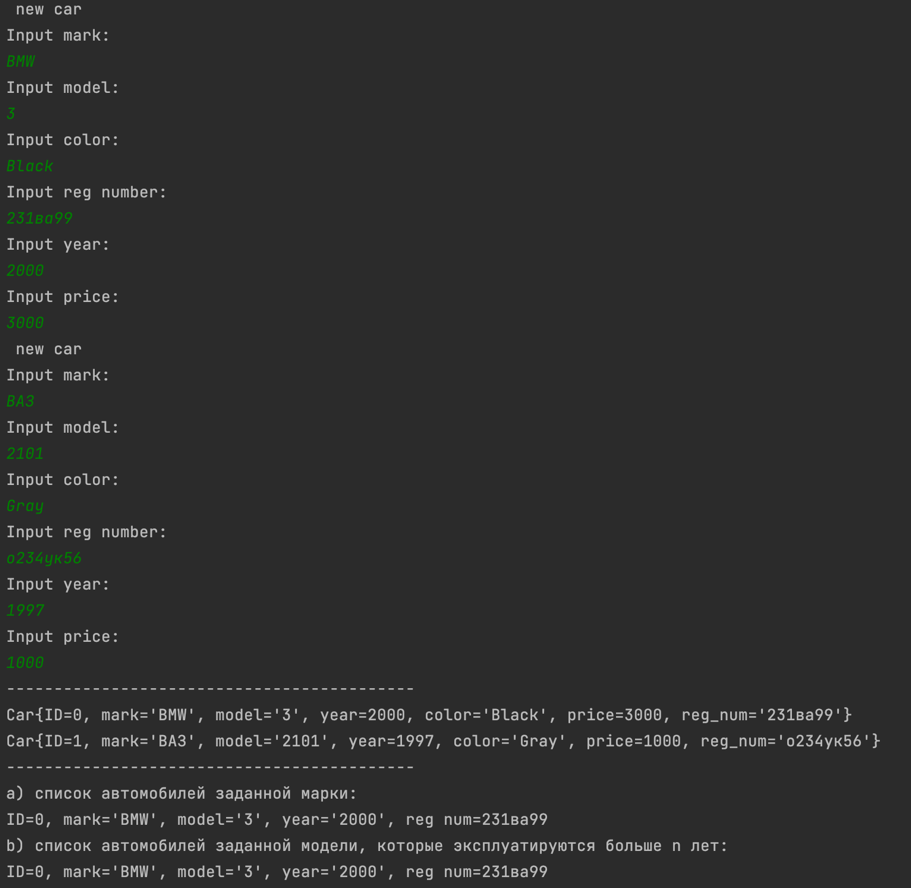


Рисунок 4 — пример выполнения

**Вариант 3:** в следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия. При этом могут рассматриваться два варианта:

• каждая строка состоит из одного слова;

• 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 каждом слове стихотворения Николая Заболоцкого заменить первую букву слова на прописную.
2. Определить частоту повторяемости букв и слов в стихотворении Александра Пушкина.

**Решение:**

import java.util.\*;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 ArrayList<String> words = new ArrayList<>();  
 String s = "";  
 StringBuilder stringBuilder = new StringBuilder();  
 while ( true ) {  
 s = scanner.nextLine();  
 if ( s.isEmpty() )  
 break;  
 stringBuilder.append(s.toLowerCase()+ "\n");  
 System.out.println(stringBuilder);  
 }

Результат выполнения представлен на рисунке 5.

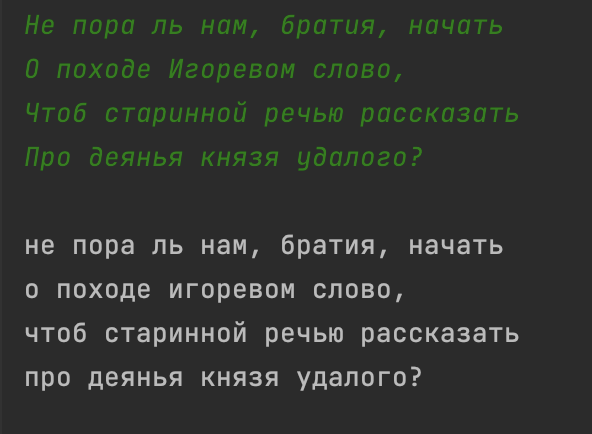


Рисунок 5 — пример выполнения

import java.util.\*;  
public class Main {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 ArrayList<String> words = new ArrayList<>();  
 String s = "";  
 String cleanedString = "";  
 StringBuilder stringBuilder =new StringBuilder();  
 HashMap<String, Integer> wordFrequencyMap = new HashMap<>();  
 while ( true ) {  
 s = scanner.nextLine();  
 if ( s.isEmpty() )  
 break;  
 cleanedString = s.replaceAll("[!?.,—;]", "").toLowerCase();  
 stringBuilder.append(cleanedString);  
 words.addAll(Arrays.asList(cleanedString.split(" ")));  
 }  
 for (int i=0; i<words.size(); i++){  
 if (words.get(i).isEmpty()){  
 words.remove(i);  
 }  
 }  
 for (String word : words) {  
 if (wordFrequencyMap.containsKey(word)) {  
 wordFrequencyMap.put(word, wordFrequencyMap.get(word) + 1);  
 } else {  
 wordFrequencyMap.put(word, 1);  
for (String key : wordFrequencyMap.keySet()) {  
 System.out.println(key + " : " + wordFrequencyMap.get(key));  
 }  
 //повторяеомсть букв  
 HashMap<Character, Integer> charFrequencyMap = new HashMap<>();  
 for (int i=0; i<stringBuilder.length(); i++){  
 char curChar = stringBuilder.charAt(i);  
 if (charFrequencyMap.containsKey(curChar)) {  
 charFrequencyMap.put(curChar, charFrequencyMap.get(curChar) + 1);  
 } else {  
 charFrequencyMap.put(curChar, 1);  
 }  
 }  
 for (char key : charFrequencyMap.keySet()) {  
 System.out.println(key + " : " + charFrequencyMap.get(key));  
 }  
 scanner.close();  
 }

Результат выполнения представлен на рисунке 6.

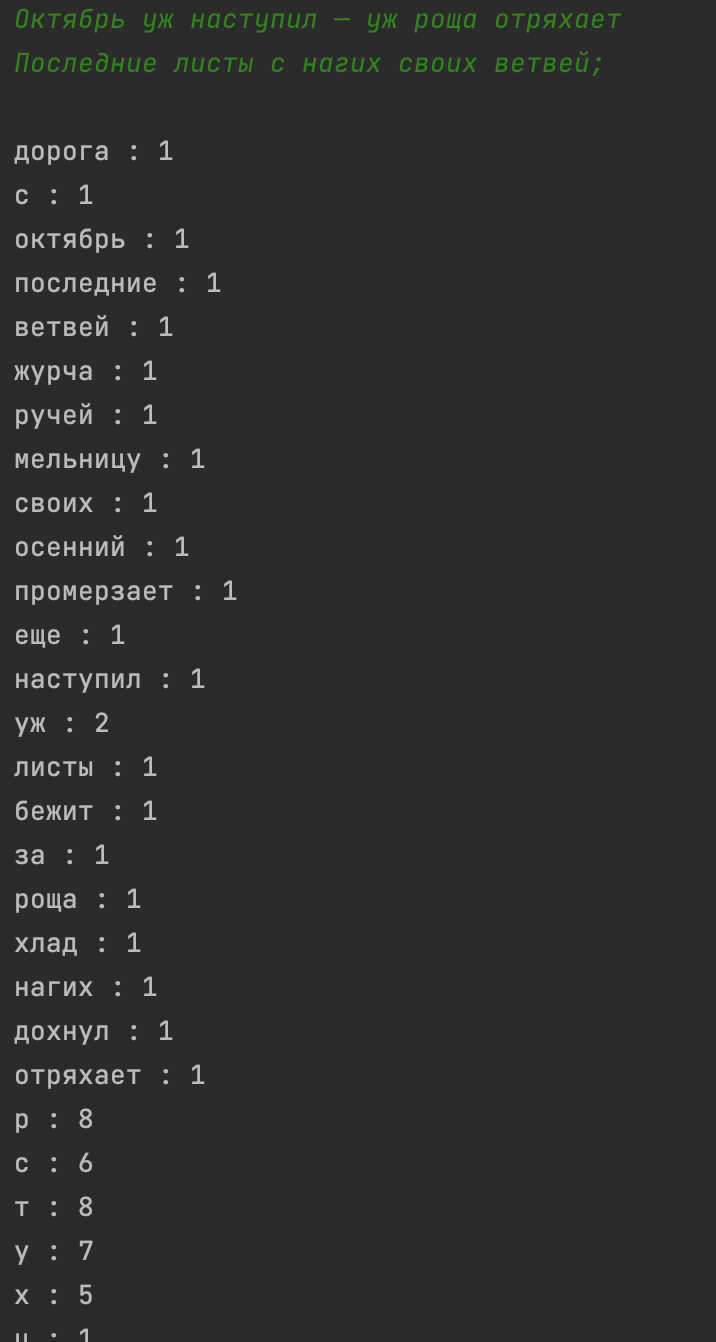


Рисунок 6 — пример выполнения

**Вариант 4:**

1. Прочитать текст Java-программы и удалить из него все “лишние” пробелы и табуляции, оставив только необходимые для разделения операторов.
2. Из текста Java-программы удалить все виды комментариев.

**Решение:**

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 System.out.println("Hello world!");  
 StringBuilder stringBuilder = new StringBuilder();  
 try {  
 File myObj = new File("/Users/lol/Desktop/untitled/Lab5/Lab5.4.1/src/1.txt");  
 Scanner myReader = new Scanner(myObj);  
 while (myReader.hasNextLine()) {  
 String data = myReader.nextLine();  
 System.out.println(data.replaceAll("\\s{2,}", " "));  
  
 stringBuilder.append(data);  
 stringBuilder.append("\n");  
 }  
 myReader.close();  
 } catch (FileNotFoundException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 System.out.println("--------------------------------------------------");  
 System.out.println("второе задание");  
 int l = stringBuilder.length();  
 while (l>=0) {  
 if(stringBuilder.indexOf("/\*")>0 && stringBuilder.indexOf("\*/") >0){  
 stringBuilder.delete(stringBuilder.indexOf("/\*"), stringBuilder.indexOf("\*/", stringBuilder.indexOf("/\*")) + 2);  
 }  
 if (stringBuilder.indexOf("//")>0 && stringBuilder.indexOf("\n", stringBuilder.indexOf("//"))>0){  
 stringBuilder.delete(stringBuilder.indexOf("//"), stringBuilder.indexOf("\n", stringBuilder.indexOf("//")));  
 }  
 l = l - stringBuilder.indexOf("\n");  
 }  
 System.out.println(stringBuilder);  
 }  
}

Результат выполнения представлен на рисунке 7.

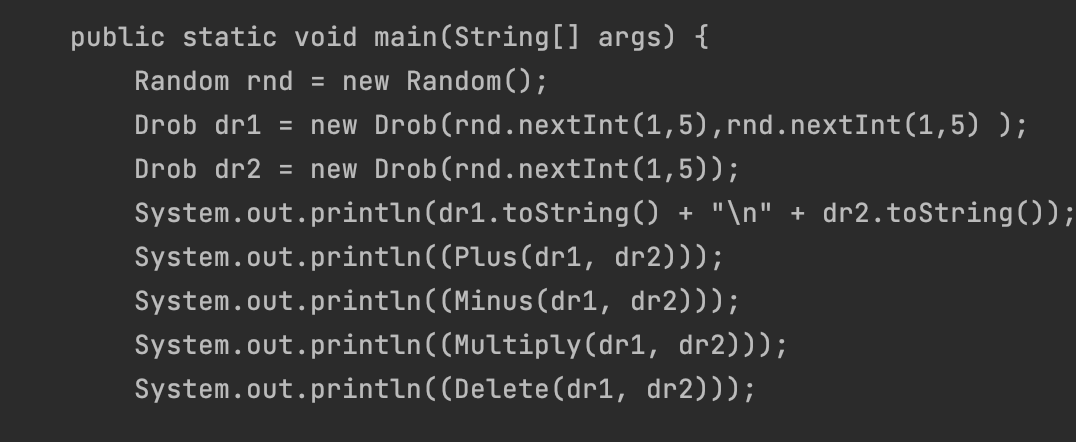
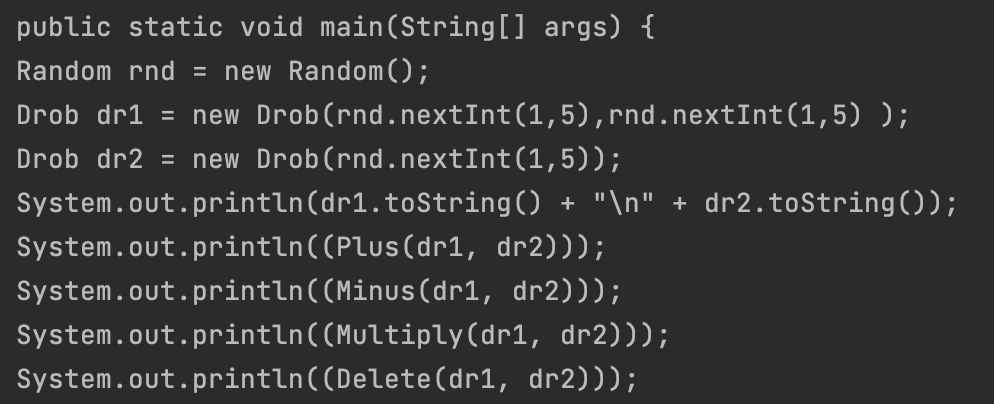


Рисунок 7 — пример выполнения

**Вывод:** были разработаны классы и методы согласно вариантам.