|  |  |
| --- | --- |
|  | **Министерство образования и науки 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

НАПРАВЛЕНИЕ ПОДГОТОВКИ: 09.04.01 Информатика и вычислительная техника

**О т ч е т**

**по лабораторной работе №5**

**Дисциплина: Языки программирование при работе с большими данными**

**Название:**

студент группы ИУ6-21М  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Коваленко М.В.**

(Подпись, дата) (Фамилия И.О.)

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

(Подпись, дата) (Фамилия И.О.)

Москва, 2020

**Задание 1**

Выполнить задания на основе варианта 1 лабораторной работы 4, 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.

**Программа**

**Основной файл**

|  |
| --- |
| import java.io.IOException;  //Создать класс City (город) с внутренним классом,  // с помощью объектов которого можно хранить информацию о проспектах, улицах, площадях.  public class Main {  public static void main(String[] args) {  City ct = new City();  ct.info.print();  }  } |

**Класс City**

|  |
| --- |
| import java.util.InputMismatchException;  import java.util.Scanner;  public class City {  public static Scanner ob = new Scanner(System.in);  public String city;  public info info;  public City() {  System.out.println("Введите название города");  setCity (ob.next());  info = new info();  }  public String getCity() {  return city;  }  public void setCity(String city) {  this.city=city;  }  public class info {  int pr,st,sq;  public info() {  System.out.println("Количество проспектов в городе:");  try {  setPr(ob.nextInt());  }  catch (InputMismatchException e) {  System.out.println("Количество проспектов надо написать числом");  }  System.out.println("Количество улиц в городе:");  try {  setSt(ob.nextInt());  }  catch (InputMismatchException e) {  System.out.println("Количество улиц надо написать числом");  }  System.out.println("Количество площадей в городе:");  try {  setSq(ob.nextInt());  }  catch (InputMismatchException e) {  System.out.println("Количество площадей надо написать числом");  }  }  public int getPr() {  return pr;  }  public void setPr(int pr) {  this.pr=pr;  }  public int getSt() {  return st;  }  public void setSt(int st) {  this.st=st;  }  public int getSq() {  return sq;  }  public void setSq(int sq) {  this.sq=sq;  }  public void print() {  System.out.print("В городе "+getCity()+" "+getPr()+" проспектов,"+getSt()+" улиц и "+getSq()+" площадей");  }  }  } |

**Задание 2**

Выполнить задания из варианта 2 лабораторной работы 4, реализуя собственные обработчики исключений и исключения ввода/вывода.

interface Mobile <- abstract class Siemens Mobile <- class Model.

**Программа**

|  |
| --- |
| public interface Mobile {  void startCall (String number);  String searchForCall ();  void acceptCall (String number);  void breakCall ();  public static void main(String[] args) {  SiemensMobile mob = new Model();  mob.connectToSiemensOnline();  try {  String s = "1000";  mob.startCall(s);  if (Integer.parseInt(s) < 1000)  {  throw new MyException("телефонный номер должен быть длиннее");  }  System.out.println(mob.searchForCall());  mob.acceptCall("8111111119");  mob.breakCall();  }  catch (MyException e) {  System.out.println("телефонный номер должен быть длиннее");  }  }  class MyException extends Throwable {  public MyException(String s) {  }  }  }  abstract class SiemensMobile implements Mobile {  Boolean incomingCall=true;  Boolean isTalkingNow;  abstract void connectToSiemensOnline();  }  class Model extends SiemensMobile {  @Override  void connectToSiemensOnline() {  System.out.println("Подсоединились к сервису Сименс Онлайн");  }  @Override  public void startCall(String number) {  isTalkingNow = true;  }  @Override  public String searchForCall() {  while (true){  if (incomingCall){  if (isTalkingNow){  return "Линяя занята";  } else return "Вызов принят";  }  }  }  @Override  public void acceptCall(String number) {  isTalkingNow = true;  }  @Override  public void breakCall() {  isTalkingNow = false;  System.out.println("Вызов завершен");  }  } |

**Задание 3**

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

В каждой строке найти слова, начинающиеся с гласной буквы.

**Программа**

|  |
| --- |
| import java.util.Arrays;  import java.util.HashSet;  import java.util.Scanner;  import java.util.Set;  public class Lab5\_3 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  Set<String> words = new HashSet<String>();  char[] alphabet = "aeijouуеэоаыяию".toCharArray();  System.out.println("Enter some strings ");  String s = null;  while (true) {  s = scanner.nextLine();  if (s.isEmpty())  break;  words.addAll(Arrays.asList(s.split("\\s+")));  }  for (int i=0; i<alphabet.length; i++) {  for (String head : words) {  if (head.charAt(0) == alphabet[i])  System.out.println("Слово, начинающееся с гласной: " + head);  }  }  scanner.close();  }  } |

**Задание 4**

При выполнении следующих заданий для вывода результатов создавать новую директорию и файл средствами класса File.

Прочитать текст Java-программы и все слова public в объявлении атрибутов и методов класса заменить на слово private.

**Программа**

|  |
| --- |
| import java.io.\*;  public class Lab5\_4 {  public static void main(String[] args) throws IOException {  BufferedReader br = new BufferedReader(new FileReader("C:/replace.txt"));  String line = "";  String FileContent = "";  String NL = System.getProperty("line.separator");  while ((line = br.readLine()) != null) {  FileContent = FileContent + line + NL;  }  br.close();  FileContent = FileContent.replaceAll("public","private");  System.out.println(FileContent);  File file = new File("replace\_new.txt");  if(file.createNewFile()) {  BufferedWriter bw = new BufferedWriter(new FileWriter(file));  bw.write(FileContent);  bw.close();  }  }  } |

**Выводы**

В результате выполнения данной лабораторной работы разобралась с исключениями и файлами на языке Java.