*Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«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

ДИСЦИПЛИНА Языки программирования для работы с большими данными

**Отчет**

**по лабораторной работе № 5**

**Вариант 5**

Студент гр. ИУ6-21М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_**Дейкин И. И.\_\_\_

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2020

**ЗАДАНИЕ 1**

*Условие*

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

Задание: Создать класс Календарь с внутренним классом, с помощью объектов которого можно хранить информацию о выходных и праздничных днях.

*Решение*

Класс Календарь с внутренним классом приведен ниже:

|  |
| --- |
| package com.company;  import java.util.ArrayList;  import java.util.List;  public class Calendar {  private List<Holiday> holidays = new ArrayList<Holiday>();  private char[] map = new char[366]; //здесь для каждого дня w - раб день, h - вых  //Изначально все рабочие дни, даже выходные не делал  public Calendar () {  for (int i = 0; i < 366; i++) {  map[i] = 'w';  }  }  public class Holiday {  private String name;  private int date\_strt; //Решил не заморачиваться с датами, сделал интами  private int date\_end;  public Holiday(String s, int d1, int d2) {  this.name = s;  this.date\_end = d2;  this.date\_strt = d1;  }  }  public void AddHoliday(String s, int d1, int d2) {  Holiday h = new Holiday(s, d1, d2);  holidays.add(h);  for (int i = d1; i < d2; i++) {  map[i] = 'h';  }  }  public void Show() {  System.out.println(map);  }  public void IsHol(int i) { //Новый метод - выходной ли день?  try {  System.out.println(map[i]);  } catch (IndexOutOfBoundsException e1) {  System.out.println("Wrong year!");  }  }  } |

Основная программа приведена ниже:

|  |
| --- |
| package com.company;  import java.util.InputMismatchException;  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  // write your code here  Scanner scan = new Scanner(System.in);  Calendar cal = new Calendar();  cal.Show();  System.out.println("Adding new holiday:");  String s = scan.nextLine();  int d1 = 0, d2 = 0;  try {  d1 = scan.nextInt();  d2 = scan.nextInt();  } catch (InputMismatchException e1) {  System.out.println("O-la-la! You've put something somewhere where it doesn't belong!");  }  cal.AddHoliday(s, d1, d2);  cal.Show();  int k = 0;  try {  k = scan.nextInt();  } catch (InputMismatchException e1) {  System.out.println("O-la-la! You've put something somewhere where it doesn't belong!");  }  cal.IsHol(k); //Этот метод показывает, выходной ли день k, в методе обрабатывается выход за границы  }  } |

**ЗАДАНИЕ 2**

*Условие*

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

Задание: Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов: interface Mobile <- abstract class Siemens Mobile <- class Model.

*Решение*

Основной метод, содержащий обработку исключений, приведен ниже:

|  |
| --- |
| package com.company;  import java.util.InputMismatchException;  import java.util.Scanner;  public class Main {  public static void main(String[] args) throws NoFreeStuff {  // write your code here  Scanner scan = new Scanner(System.in);  char c1, c2;  String s = scan.nextLine();  String cs = scan.nextLine();  c1 = cs.charAt(0);  c2 = cs.charAt(1);  int p = 0;  try { //Обработка стандартного исключения  p = scan.nextInt();  } catch (InputMismatchException e1) {  System.out.println("Wrong input!");  }  if (p == 0) throw new NoFreeStuff(1); //Свое исключение без обработки  Model me\_phone = new Model(s, c1, c2, p);  me\_phone.Info();  me\_phone.Ring();  System.out.println("Let's decrease the price!");  int k = scan.nextInt();  try {  me\_phone.Discount(k); //Метод понижения цены выбрасывает исключение NoFreeStuff  } catch (NoFreeStuff e2) { //его и обрабатываем  System.out.println("Hey, buster, no free stuff here!");  }  me\_phone.Info();  }  } |

**ЗАДАНИЕ 3**

*Условие*

В следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия. Найти в строке наибольшее число цифр, идущих подряд.

*Решение*

Решение приведено на листинге ниже:

|  |
| --- |
| package com.company;  import java.io.File;  import java.io.FileNotFoundException;  import java.util.Scanner;  public class Main {  public static void main(String[] args) throws FileNotFoundException {  File input = new File("input.txt");  try (Scanner scan = new Scanner(input)) {  // Ищем наибольшее число цифр идущих подряд  int res = 0;  while (scan.hasNextLine()) {  String s = scan.nextLine();  int b = 0;  for (int i = 0; i < s.length(); i++) {  if (Character.isDigit(s.charAt(i))) {  // Если цифра, наращиваем счетчик  b++;  }  else {  // Если нет - проверяем, больше ли результат, и сбрасываем  if (b > res) res = b;  b = 0;  }  }  }  System.out.println("Most digits - " + res);  }  catch (FileNotFoundException exp) {  System.out.println("No such file found!");  }  }  } |

**ЗАДАНИЕ 1**

*Условие*

При выполнении следующих заданий для вывода результатов создавать новую директорию и файл средствами класса File. Файл содержит символы, слова, целые числа и числа с плавающей запятой. Определить все данные, тип которых вводится из командной строки.

*Решение*

Приведем решение на листинге ниже:

|  |
| --- |
| package com.company;  import java.io.File;  import java.io.FileNotFoundException;  import java.io.FileWriter;  import java.io.IOException;  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  File input = new File("input.txt");  Scanner scan\_ = new Scanner(System.in);  System.out.println("Enter the desired type:");  String type = scan\_.nextLine();  File dir = new File("outputs");  Boolean dir\_cr = dir.mkdirs();  String path = "outputs\\output\_" + type + ".txt";  File output = new File(path);  try {  FileWriter myWriter = new FileWriter(path);  try (Scanner scan = new Scanner(input)) {  while (scan.hasNextLine()) {  String s = scan.nextLine();  // Определяем тип  String b\_type = "Int";  for (int i = 0; i < s.length(); i++) {  if (s.charAt(i) == '.' && b\_type.equals("Int")) {  b\_type = "Float";  }  if (!Character.isDigit(s.charAt(i)) && s.charAt(i) != '.') {  b\_type = "Word";  }  }  if (b\_type.equals("Word") && s.length() == 1) {  b\_type = "Char";  }  // Тип определен  if (b\_type.equals(type)) {  // Выводим подобранные строки в файл  myWriter.write(s + '\n');  }  }  }  catch (FileNotFoundException exp) {  System.out.println("No input file found!");  }  myWriter.close();  } catch (IOException e) {  System.out.println("Output file exception!");  }  }  } |

**ВЫВОД**

В ходе выполнения данной работы было получено представление об обработке исключений, также были использованы операции работы с файлами: считывание файла, запись в файл, обработка исключений файла, создание новой директории.