|  |  |
| --- | --- |
| 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**

**Отчет**

**по лабораторной работе № 4**

**Название:** Внутренние классы, интерфейсы

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-21М |  |  | Д.Ю. Ермошин |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

**Задание**

Научиться работать с внутренними классами и интерфейсами в Java.

**Выполнение работы**

**Вариант 1**

Ввести строки из файла, записать в список ArrayList. Выполнить сортировку строк.

Mobile.java

|  |
| --- |
| package lab4;  import com.sun.tools.internal.ws.wsdl.document.Output;  import java.io.BufferedReader;  import java.io.BufferedWriter;  import java.io.FileWriter;  import java.io.IOException;  import java.io.InputStream;  import java.io.InputStreamReader;  import java.io.OutputStream;  import java.io.OutputStreamWriter;  import java.util.ArrayList;  import java.util.Arrays;  import java.util.List;  public class Mobile {  private final List<MobileInfo> mobileInfos;  public Mobile(InputStream inputStream) throws IOException {  this.mobileInfos = Arrays.asList(  new MobileInfo(inputStream),  new MobileInfo(inputStream),  new MobileInfo(inputStream)  );  }  public Mobile(List<MobileInfo> mobileInfos) {  this.mobileInfos = mobileInfos;  }  public Mobile() {  this.mobileInfos = Arrays.asList(  new MobileInfo(),  new MobileInfo(),  new MobileInfo()  );  }  public List<MobileInfo> getMobileInfos() {  return mobileInfos;  }  public static class MobileInfo {  private final List<MobileModel> mobiles;  public MobileInfo(InputStream inputStream) throws IOException {  this.mobiles = Arrays.asList(  new MobileModel(inputStream),  new MobileModel(inputStream),  new MobileModel(inputStream)  );  }  public MobileInfo(List<MobileModel> mobiles) {  this.mobiles = mobiles;  }  public MobileInfo() {  this.mobiles = Arrays.asList(  new MobileModel(),  new MobileModel(),  new MobileModel()  );  }  public List<MobileModel> getMobiles() {  return mobiles;  }  public static class MobileModel {  private final List<String> properties;  public MobileModel(InputStream inputStream) throws IOException {  this.properties = readFromInputStream(inputStream);  }  public MobileModel(List<String> properties) {  this.properties = properties;  }  public MobileModel() {  this.properties = Arrays.asList("property1", "property2", "property3");  }  public List<String> getProperties() {  return properties;  }  private List<String> readFromInputStream(final InputStream inputStream) throws IOException {  final List<String> result = new ArrayList<>();  final BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));  String line = br.readLine();  while (!line.equals("END")) {  result.add(line);  line = br.readLine();  }  return result;  }  public void writeToOutputStream(final OutputStream outputStream) throws IOException {  try (final BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(outputStream))) {  for (final String property : properties) {  bw.write(property);  bw.newLine();  }  }  }  }  }  } |

Variant1.java

|  |
| --- |
| package lab4;  /\*\*  \* 3. Создать класс Mobile с внутренним классом,  \* с помощью объектов которого можно хранить информацию о моделях телефонов и их свойствах.  \*/  public class Variant1 {  public void work() {  final Mobile mobile = new Mobile();  mobile.getMobileInfos().forEach(mobileInfo ->  mobileInfo.getMobiles()  .forEach(mobileModel-> mobileModel.getProperties() .forEach(System.out::println)  ));  }  } |

**Вариант 2**

Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов

interface Фильм <- class Отечественный Фильм <- class Комедия.

Film.java

|  |
| --- |
| package lab4;  public interface Film {  void watch();  int getRating();  void setRating(final int rating);  } |

SovietFilm.java

|  |
| --- |
| package lab4;  public class SovietFilm implements Film {  private int rating;  @Override  public void watch() {  System.out.println("Watching soviet film");  }  @Override  public int getRating() {  return rating;  }  @Override  public void setRating(final int rating) {  watch();  this.rating = rating;  }  } |

Comedy.java

|  |
| --- |
| package lab4;  public class Comedy extends SovietFilm {  @Override  public void watch() {  System.out.println("Watching soviet comedy");  }  } |

Variant2.java

|  |
| --- |
| package lab4;  /\*\*  \* Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов  \* 10. interface Фильм <- class Отечественный Фильм <- class Комедия.  \*/  public class Variant2 {  public void work() {  final Film sovietFilm = new SovietFilm();  sovietFilm.setRating(7);  System.out.println("Rating of soviet film is " + sovietFilm.getRating());  final Film comedy = new Comedy();  comedy.setRating(9);  System.out.println("Rating of soviet comedy is " + comedy.getRating());  }  } |

**Заключение**

В данной лабораторной работе мы познакомились ближе с внутренними классами и интерфейсами в Java.