

**Course Experiment Report**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Course:** | Java Programming Language | | | | | | |
|  |  | | | | | | |
| **Semester:** | 1-18th | **week** | 2nd | **year** | | 1st | **term** |
|  |  |  |  |  | |  |  |
| **Major:** | Software Engineering Class | | | | | **Class:** | 2018 |
|  |  | | | | |  |  |
| **Student name:** | Song Xingjian | | **Student No.:** | | 222018321062006 | | |
|  |  | |  | |  | | |
| **Teacher:** | Wang Xiaomeng | | | | | | |

College of Computer and Information Science

|  |  |  |  |
| --- | --- | --- | --- |
| Project | Exp 6 I/O Operation | | |
| Time | 2019.12.6 | Type | □Verification □Design □Synthetical |
| **1. Objective**  Through this training, to master binary I/O operation and text I/O operation.  **2. Requirement**  Programming the following exercise  **Problem 1:** Extract all words which contain the sub-string “\*\*” from the text file word.txt, and store them into a new file \*\*\_word.txt.  **Problem 2:** Randomly generate circles and rectangles and sort by area.  **3. Content and design of experiment(main content, operations, algorithm description or code of program)**  **Problem 1:**  **Codes:**  **import** java.util.Scanner;  **import** java.io.File;  **import** java.io.FileNotFoundException;  **import** java.io.PrintWriter;  **public** **class** FindSubString {  **public** **static** **void** main(String[] args)**throws** FileNotFoundException {  //设置要寻找的substring  String subString = "ac";    Scanner input = **new** Scanner(**new** File("E:\\Java\\实验课\\Exp6\\words.txt")); //输入文件流  PrintWriter pw = **new** PrintWriter("E:\\Java\\实验课\\Exp6\\words\_out.txt"); //输出文件流  **while**(input.hasNext()) { //如果之后还有文本，则循环  String word = input.nextLine(); //将一行数据读入并存放在word中  **int** index = word.indexOf(subString); //查找word中是否含有substring，如果没有则返回-1  **if**(index != -1) { //如果找到，则将word打印到输出文件流  pw.println(word);  }  }  pw.close(); //关闭输出文件流  }  }  **Results:**    **Summary:**  用Scanner打开输入流，用PrintWriter创建输出流。利用Scanner类中的hasNext()方法判断之后是否还有文本，这个方法作为循环结束的标准特别方便。对输入流逐行查找，利用indexOf方法查找是否含有子字符串，如果有则在输出流打印出来。  **Problem 2:**  **Codes:**   * **Comparable抽象类及compareTo方法的重写：**   **public** **abstract** **class** GeometricObject **implements** Comparable<GeometricObject>{  **public** **abstract** **double** getArea();  @Override //重写compareTo  **public** **int** compareTo(GeometricObject g) {  **if**(**this**.getArea()>g.getArea()) {  **return** 1;  }**else** **if**(**this**.getArea()<g.getArea()) {  **return** -1;  }**else** {  **return** 0;  }  }  }   * **Circle和Rectangle图形子类的创建：**   **public** **class** Circle **extends** GeometricObject{  **private** **double** r;  //初始化圆形  **public** Circle(**double** r){  **this**.r=r;  }  @Override //重写获得圆形面积函数，实现接口中的方法  **public** **double** getArea() {  **return** r\*r\*Math.***PI***;  }  **public** String toString(){  **return** "Circle," + r;  }  }  **public** **class** Rectangle **extends** GeometricObject{  **private** **double** h; //矩形的高  **private** **double** w; //矩形的宽  //初始化矩形  **public** Rectangle(**double** h, **double** w){  **this**.h=h;  **this**.w=w;  }  @Override //重写获得矩形面积函数，实现接口中的方法  **public** **double** getArea() {  **return** h\*w;  }  **public** String toString(){  **return** "Rectangle," + h + "," + w;  }  }   * **随机生成数据方法：**   //随机生成图形数据  **public** **static** **void** generate(String path, **int** num) {  **try** {  PrintWriter pw = **new** PrintWriter(path);  **for** (**int** i = 0; i < num; i++) {  **int** rand=(**int**)(Math.*random*()\*2+1);//随机生成1或2  **if**(rand%2==0) {  Circle c=**new** Circle(Math.*random*());  pw.println(c.toString());  }**else** {  Rectangle r=**new** Rectangle(Math.*random*(), Math.*random*());  pw.println(r.toString());  }  }  pw.close();  } **catch** (IOException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }  }   * **排序并输出：**   //排序  **public** **static** **void** sortByArea(String srcPath, String sortPath){  //创建一个图像类型的List  List<GeometricObject> list = **new** ArrayList<GeometricObject>();  **try** {  //输入流文件  Scanner input = **new** Scanner(**new** FileInputStream(srcPath));  //先将输入流读出的文件存到list中  **while**(input.hasNext()) {  String str = input.nextLine();  String[] items = str.split(","); //以“，”为标志将字符串分开  **if**(items[0].equals("Rectangle")) {  list.add(**new** Rectangle(Double.*parseDouble*(items[1]), Double.*parseDouble*(items[2])));  }  **if**(items[0].equals("Circle")) {  list.add(**new** Circle(Double.*parseDouble*(items[1])));  }  }  //对list进行排序,因为在GeometricObject类中重写了compareTo方法，所以这里排序是按照面积排序  Collections.*sort*(list);  //输出流文件  PrintWriter pw=**new** PrintWriter(sortPath);  **for**(GeometricObject g:list) {  pw.println(g.toString()+"\t"+g.getArea());  }  pw.close();  input.close();  } **catch** (IOException e1) {  // **TODO** Auto-generated catch block  e1.printStackTrace();  }  }  }   * **主函数：**   **public** **static** **void** main(String[] args) {  // 随机生成图形数据  String loc\_rand = "E:\\Java\\实验课\\Exp6\\OutRandom.txt";  *generate*(loc\_rand,1000);  *sortByArea*("E:\\Java\\实验课\\Exp6\\OutRandom.txt","E:\\Java\\实验课\\Exp6\\OutSort.txt");  System.***out***.print("Finish!");  }  **Results:**      **Summary:**  为circle和rectangle创建抽象类GeometricObject，以便提取他们的共同特征。分别在子类中重写面积计算方法getArea和toString方法。  在读取分析输入流字符串时，利用String的split方法将字符串切分，将图形的参数数据存入List中，然后进行排序操作。  库函数中的sort调用了Comparable接口进行比较排序，而在本次实验中我们对Comparable抽象类中的compareTo方法进行了重写，所以在排序时调用sort方法时才会按照面积对图形进行排序。  在随机生成圆形或矩形时，我利用随机数生成器random方法生成1~3的数，然后强制转化为int类型，即统一为了1或2，分别指代一种图形即可。 | | | |

|  |  |  |
| --- | --- | --- |
| Teacher’s  comments | content and design of experiment（A-E）： |  |
| operations, algorithm description or code of program（A-E）： |  |
| results（A-E）： |  |
| summary and analysis of experiment（A-E）： |  |
| Grade（A-E）： | |