高级语言程序设计I实验报告

**实验十 集合**

班 级： 22软件工程

姓 名： 王玉升

学 号： 22206091012

成 绩：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验名称 | | **实验十 泛型与集合** | | | |
| 实验日期 | | |  | 实验课时 | 2 |
| 实  验  目  的  及  要  求 | （给出本次实验内容及要求的具体描述）   1. 实验目的： 2. 了解java.awt包和javax.swing包 3. 了解常用的容器、常用的布局、常用的组件 4. 掌握事件处理的模式 5. 掌握常用事件的处理 6. 了解对话框的使用 7. 实验要求： 8. 安JDK及Ｅclipse开发环境，并使用Ｅclipse开发平台进行java语言程序的编辑、编译和运行。 9. 在D盘或E盘以自己的学号末尾四位数字为名创建文件夹，并在其下面为每个实验项目创建子文件夹，程序相关文件存储在相应的文件夹中（如学号末尾是”1023”，实验六则应创建目录结构“E:\1023\sy07”，实验六程序文件放在“…\sy07”）。 10. 实验报告中附上程序清单和程序运行结果截图（控制台中输出结果后用自己的真实姓名注释）。 11. 每题程序代码中至少有两处用自己的真实姓名进行注释。   5．实验结束后进行总结。 | | | | |
| 实  验  环  境 | （列出本次实验所使用的平台和相关软件）  地点：南6207  开发环境：JDK1.8、eclipse | | | | |
| 实  验  过  程  及  实  验  结  果 | 1. **创建一个ArrayList集合，用来管理学生的名单。将【代码X】替换为Java程序代码（注释不能删除）：**   **/\*本题重点是掌握ArrayList集合的创建、元素的添加、修改、删除等操作\*/**   * 1. 程序清单：   package 一**;** import java.util.ArrayList**;** public class StudentName {  public static void main(String argvs[]) {  ArrayList<String> list= new ArrayList<>()**;**//【代码1】创建ArrayList集合list  list.add("王玉升")**;** list.add("张小宝")**;**//【代码2】将“张小宝”加入名单  list.add("陈娟")**;** list.add("华国庆")**;** list.add("孙佳")**;** list.add(**0,**"陈红叶")**;** //【代码3】将“陈红叶”插入到名单最前面  int index = list.indexOf("陈娟")**;** if (index != -**1**) {  list.set(index**,** "陈娟娟")**;** }  // 王玉升   //【代码4】将“陈娟”姓名修改为““陈娟娟”  list.remove("华国庆")**;** //【代码5】将“华国庆”从名单中删除  System.*out*.println("人数："+list.size())**;**//【代码6】输出总人数  System.*out*.println("第三个人是："+list.get(**2**))**;**//【代码7】输出名单中第3个人姓名  System.*out*.println(list)**;** } }   * 1. 运行程序和运行结果（截图）：      1. **使用堆栈结构实现输出数列的前10项，其中an=2an-1+2an-2,a1=3,a2=8。**   **/\*本题重点是掌握Stack堆栈类的使用\*/**  **提示：**   * **Stack类的主要方法的方法有：**  |  |  | | --- | --- | | **方法** | **功能描述** | | **public E push(E item)** | **实现入栈操作** | | **public E pop()** | **实现出栈操作。** | | **public boolean empty()** | **判断堆栈是否还有数据。** | | **public E peek()** | **读取堆栈顶端的数据，但不删除该数据。** | | **public int search(Object data)** | **获取数据在堆栈中的位置。** |  1. 程序清单：   package 二**;** import java.util.Stack**;** public class Sy10\_2{  public static void main(String[] args) {  Stack<Integer> stack = new Stack<>()**;** stack.push(**3**)**;** stack.push(**8**)**;** // 王玉升  while (stack.size() > **1**) {  int a = stack.pop()**;** int b = stack.pop()**;** int c = a + b \* **2;** stack.push(c)**;** } // 王玉升     while (!stack.isEmpty()) {  System.*out*.print(stack.pop() + " ")**;** }  } }  （2） 运行程序和运行结果（截图）：     1. **使用TreeSet集合管理学生信息，id相同认为是同一学生，要求按年龄升序排列，年龄相同再按id升序排列**   **/\*本题重点是掌握TreeSet使用、元素排序\*/**   * 1. 程序清单：   package 三**;** import java.util.Comparator**;** import java.util.TreeSet**;** import java.util.Iterator**;** class Student{  private String id**;** //王玉升  private String name**;** private int age**;** public Student(String id**,** String name**,**int age) {  this.id = id**;** this.name = name**;** this.age=age**;** }    */\*\*\*  \* 王玉升  \** ***@return*** *\*/* public int getAge() {  return age**;** }  public void setAge(int age) {  this.age = age**;** }  public String getId() {  return id**;** }  public void setId(String id) {  this.id = id**;** }  public String getName() {  return name**;** }  public void setName(String name) {  this.name = name**;** }  // 重写toString()方法  public String toString() {  return id + ":" + name+":"+age**;** } }  package 三**;** import java.util.Comparator**;** import java.util.TreeSet**;** import java.util.Iterator**;** public abstract class Sy7\_3 implements Comparator {  public static void main(String[] args) {  TreeSet ts = new TreeSet(new Comparator<Student>() {  @Override  //【代码1】实现Comparator接口   public int compare(Student s1**,** Student s2) { //【代码2】重写compare方法  int m = s1.getId().compareTo(s2.getId())**;**//【代码3】比较s1、s2 id大小  int n = s1.getAge() - s2.getAge()**;** //比较s1、s2年龄大小（差值） 22206091012王玉升  return m == **0** ? m : n != **0** ? n : m**;**//id相同表示同一个同学，否则按年龄升序排列，年龄相同按ID长序排列   }  })**;** //王玉升  ts.add(new Student("1001"**,** "苏超"**, 20**))**;** ts.add(new Student("1002"**,** "李彬"**, 19**))**;** ts.add(new Student("1003"**,** "李进玉"**, 21**))**;** ts.add(new Student("1012"**,** "王玉升"**, 20**))**;** ts.add(new Student("1005"**,** "冯明宝"**, 18**))**;** Iterator<Student> it = ts.iterator()**;** // 生成迭代器//【代码4】生成迭代器  while (it.hasNext()) { // 遍历迭代器中的元素  //【代码5】遍历  System.*out*.println(it.next())**;** }  } }   * 1. 运行程序和运行结果（截图）：      1. **已知某学校的课程表如下**：  |  |  | | --- | --- | | **课程号** | **课程名** | | Kc01 | C程序设计 | | Kc02 | 数据结构 | | Kc03 | 数字逻辑 | | Kc04 | 数据库原理 | | Kc05 | 面向对象程序设计 |   **完成下列要求：**  **1） 使用一个TreeMap，以课程号作为键，以课程名作为值，表示上述课程安排。**  **2） 增加了一门新课程Kc06，计算机网络**  **3） Kc04课程名改为数据库概论**  **4） 删除数字逻辑这门课程**  **5）遍历课程表，输出所有的课程号及课程名**  **/\*本题重点是掌握TreeMap集合的创建、操作与遍历\*/**   * 1. 程序清单：   package 四**;** import java.util.TreeMap**;** public class CourseSchedule {  public static void main(String[] args) {    TreeMap<String**,** String> courseMap = new TreeMap<>()**;** courseMap.put("Kc01"**,** "C程序设计")**;** courseMap.put("Kc02"**,** "数据结构")**;** courseMap.put("Kc03"**,** "数字逻辑")**;** courseMap.put("Kc04"**,** "数据库原理")**;** // 王玉升   courseMap.put("Kc05"**,** "面向对象程序设计")**;** courseMap.put("Kc06"**,** "计算机网络")**;** courseMap.put("Kc04"**,** "数据库概论")**;** courseMap.remove("Kc03")**;** for (String courseId : courseMap.keySet()) {  System.*out*.println("课程号: " + courseId + ", 课程名: " + courseMap.get(courseId))**;** }  } }   * 1. 运行程序和运行结果（截图）： | | | | |
| 总  结 | 实验心得体会： | | | | |