高级语言程序设计I实验报告

**实验六 常用实用类**

班 级： 22软件工程

姓 名： 王玉升

学 号： 22206091012

成 绩：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验名称 | | **实验六 常用实用类** | | | |
| 实验日期 | | | 2023.11.20 | 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\sy06”，实验六程序文件放在“…\sy06”）。 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. **提取个人简历中重要信息并输出。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String类主要方法的使用，如字符定位、截取子串等\*/**   * 1. 程序清单：   **FindMess.java：**  import java.lang.String; public class FindMess {  public static void main(String args[]) {  String mess = "姓名:张三 出生时间:1989.10.16。个人网站:http://www.zhang.com。身高:185 cm,体重:72 kg";  int index = mess.indexOf(":"); //【代码1】mess调用indexOf(String s)方法返回字符串中首次出现冒号的位置  String name = mess.substring(index+1);  if(name.startsWith("张")) {  System.*out*.println("简历中的姓名姓\"张\"");  }  index = mess.indexOf(":", 2);//【代码2】mess调用indexOf(String s,int start)方法返回字符串中第2次出现冒号的位置  String date = mess.substring(index+1,index+11);  System.*out*.println("出生日期："+date);  index = mess.indexOf(":",index+1);  int heightPosition = mess.indexOf("身高");// 【代码3】mess调用indexOf(String s)方法返回字符串中首次出现"身高"的位置  String personNet = mess.substring(mess.indexOf("h", 1), mess.indexOf("m" ) + 1);//【代码4】获取个人网站  System.*out*.println("个人网站:"+personNet);  index = mess.lastIndexOf(":", 5);//【代码5】mess调用indexOf(String s,int start)方法返回字符串中"身高"后面的冒号位置  int cmPosition = mess.indexOf("cm");  String height = mess.substring(index+1,cmPosition);  height = height.trim();  int h = Integer.*parseInt*(height);  if(h>=180) {  System.*out*.println("简历中的身高"+height+",大于或等于180 cm");  }  else {  System.*out*.println("简历中的身高"+height+",小于180 cm");  }  index = mess.lastIndexOf(":");//【代码6】mess调用lastIndexOf(String s)返回字符串中最后一个冒号位置  int kgPosition = mess.indexOf("kg");  String weight = mess.substring(index+1,kgPosition);  weight = weight.trim();  int w = Integer.*parseInt*(weight);  if(w>=75) {  System.*out*.println("简历中的体重"+weight+",大于或等于75 kg");  }  else {  System.*out*.println("简历中的体重"+weight+",小于75 kg");  }  } }   * 1. 运行程序和运行结果（截图）：      1. **利用数组随机抽取幸运观众**，**在电视节目中经常看到随机抽取幸运观众，现在要抽取3名幸运观众。如果观众抽取得范围较少（假设10人），可以让程序使用数组实现，而且效率很高。方法是：首先将根据所有观众姓名生成数组name，然后获得数组元素的总数量，再创建与name同等长度的数组flag，表示对应观众是否抽取为幸运观众，初值为false,然后由随机抽取数组元素的下标，根据抽取的下标将数组flag对应置true,并输出幸运观众姓名。**   **/\*本题重点是掌握数组和随机函数的使用\*/**   * 1. 程序清单：   public class Audience {  String name;   public Audience(String name) {  this.name = name;  }     }  import java.lang.String; import java.util.Random;  public class Chou {           private static int getRandomIndex(int max){  Random Random = new Random();  return Random.nextInt(max); //获得随机下标    }   public static void main(String[] args) {     Audience A = new Audience("A");  Audience B = new Audience("B");  Audience C = new Audience("C");  Audience D = new Audience("D");  Audience E = new Audience("E");  Audience F = new Audience("F"); //王玉升  Audience G = new Audience("G"); //王玉升  Audience H = new Audience("H");  Audience I = new Audience("I");  Audience J = new Audience("J");      String name[] = {A.name, B.name, C.name, D.name, E.name, F.name, G.name, H.name , I.name, J.name};  boolean[] Flags = new boolean[name.length];   for(int i = 0; i < 3; i++){   int chosenIndex = *getRandomIndex*(name.length); //王玉升   while (Flags[chosenIndex]){ //如果这个人已经被选中  chosenIndex = *getRandomIndex*(name.length); //重新选一个  }   Flags[chosenIndex] = true; //王玉升   String chosenAudience = name[chosenIndex];    System.*out*.println("幸运观众是" + chosenAudience);           }   }   }   * 1. 运行程序和运行结果（截图）：      1. **字符中子串查找与替换:在给定的字符中查找“登陆”出现的位置，然后将它替换为“登录”，并将“惊慌失错”替换为“惊慌失措”。将【代码X】替换为Java程序代码（注释不能删除）：**   **/\* 本题重点是字符串中子串的查找、定位与替换\*/**   * 1. 程序清单：   import java.util.regex.\*; public class ReplaceErrorWord {  public static void main(String args[ ]) {  String str = "忘记密码，不要惊慌失错，请登陆www.yy.cn或登陆www.tt.cc";  Pattern pattern;  Matcher matcher;  String regex = "登陆";  pattern = Pattern.*compile*("登陆"); //使用regex初试化模式对象pattern  matcher = pattern.matcher(regex); //得到检索str的匹配对象matcher  while(matcher.find()) {  String s = matcher.group();//【代码3】获取匹配的子串  System.*out*.print(matcher.start()+"位置出现:");  System.*out*.println(s);  }  System.*out*.println("将\"登陆\"替换为\"登录\"的字符串:");  String result = matcher.replaceAll("登录");//【代码4】将"登陆"替换为"登录"  System.*out*.println(result);  pattern= Pattern.*compile*("惊慌失错");  matcher = pattern.matcher(result);  System.*out*.println("将\"惊慌失错\"替换为\"惊慌失措\"的字符串:");  result = matcher.replaceAll("惊慌失措");  System.*out*.println(result);  } }   * 1. }运行程序和运行结果（截图）：      1. **用户从键盘输入一行文本，程序输出其中的单词。**   **/\* 本题重点掌握字符串的分解，即split（）方法的使用，单词分隔符可能是空格、数字、标点符号等 \*/**  **输入：**Hello, Tom. Nice to meet you!  **输出：**Hello、Tom、Nice、to、meet、you   * 1. 程序清单：   2. 运行程序和运行结果（截图）：  1. **根据标点符号对字符串进行分行,即在标点符号后面插入换行符**：**首先需要理解一个概念——String字符串是不可改变的对象，也就是说，字符串在创建以后，就不会被改变，当使用字符串对象的replace()、split()等方法操作字符串时，实际上是产生了一个新的字符串对象，原有的字符串如果没有被引用，将会被垃圾收集器回收。如果频繁的使用字符串中的方法对字符串进行操作，会产生大量的没有被引用的字符串对象，这会增加垃圾收集的压力，造成系统资源的浪费。如果需要大量的操作字符串，怎样操作才合理呢？可以使用StringBuilder类有效的解决上面出现的问题，使用StringBuilder类操作字符串不会产生新的字符串对象，这样处理才更加方便效率。本题的操作流程是，首先创建StringBuilder对象，使用StringBuilder对象对字符串进行分行操作，要求输入一行字符串，然后根据分界符分行输出**   **/\*本题重点掌握StringBuilder和StringBuffer类的使用， 可用与上题类似的进行模式匹配，确定标点符号位置，在其后插入换行符”\n”\*/**  **程序运行结果：**  **输入：**  **today is a nice day today, We went on a spring outing together, Everyone had a good time!**  **输出：**  **today is a nice day today,**  **We went on a spring outing together,**  **Everyone had a good time!**   * 1. 程序清单：   2. 运行程序和运行结果（截图）： | | | | |
| 总  结 | 实验心得体会： | | | | |