《 — JAVA高级编程—》

**实 验 报 告**

陈四贵

182210710119

2021年 6月

**实验一 九九乘法口诀表的打印**

实验学时：2学时

实验类型：验证

实验要求：必修

**一、实验内容**

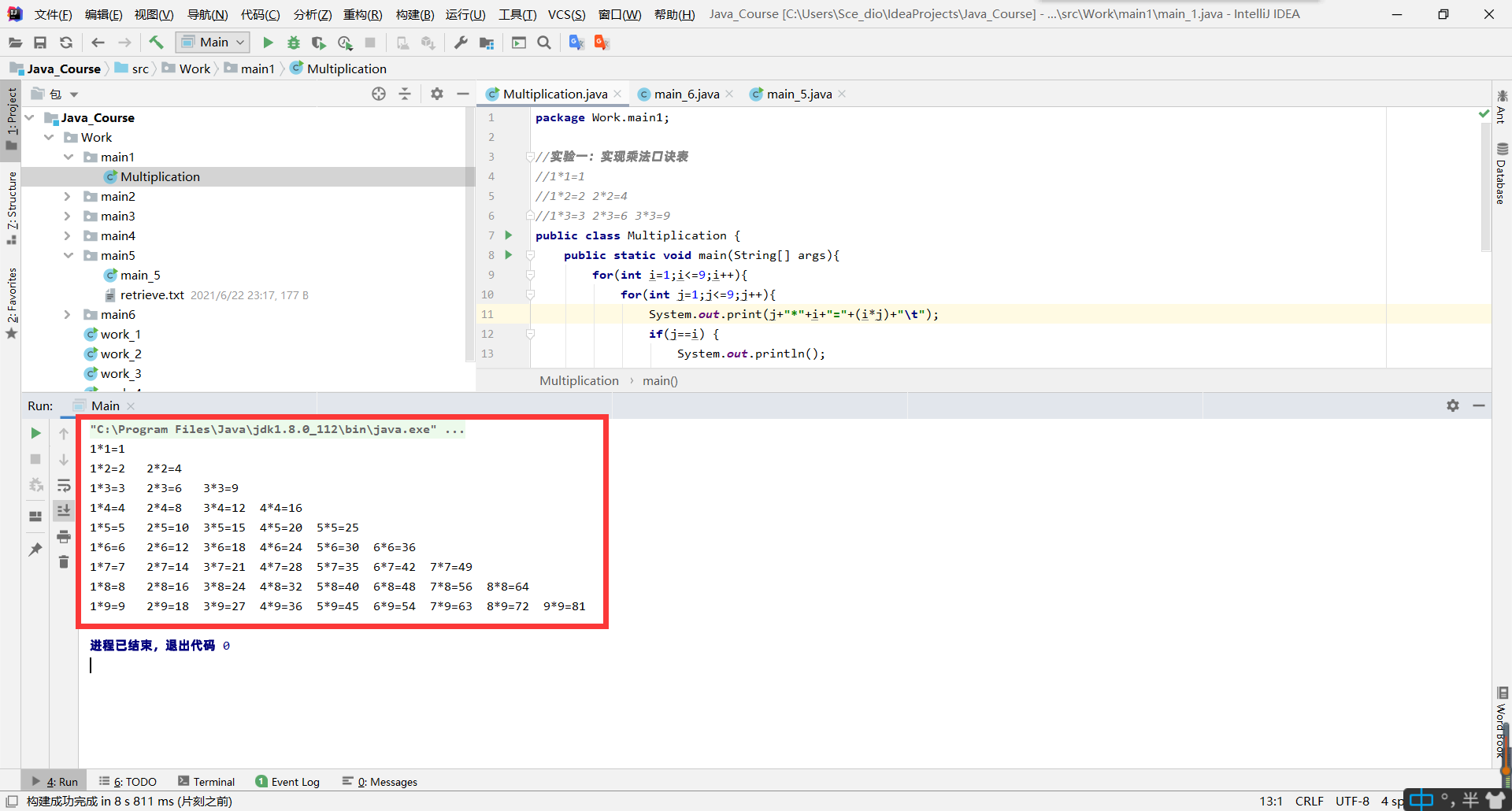
使用JAVA实现打印乘法口诀表

**二、实验组织运行要求**

每个同学独立完成上述实验要求。

**三、程序**

1. **package** Work.main1;
3. //实验一：实现乘法口诀表
4. //1\*1=1
5. //1\*2=2 2\*2=4
6. //1\*3=3 2\*3=6 3\*3=9
7. //文件名 Multiplication.java
8. **public** **class** Multiplication {
9. **public** **static** **void** main(String[] args){
10. **for**(**int** i=1;i<=9;i++){
11. **for**(**int** j=1;j<=9;j++){
12. System.out.print(j+"\*"+i+"="+(i\*j)+"\t");
13. **if**(j==i) {
14. System.out.println();
15. **break**;
16. }
17. }
18. }
19. }
20. }

**四、实验结果**

**实验二 基于数组类的二叉排序算法**

实验学时：2学时

实验类型：验证

实验要求：必修

**一、实验内容**

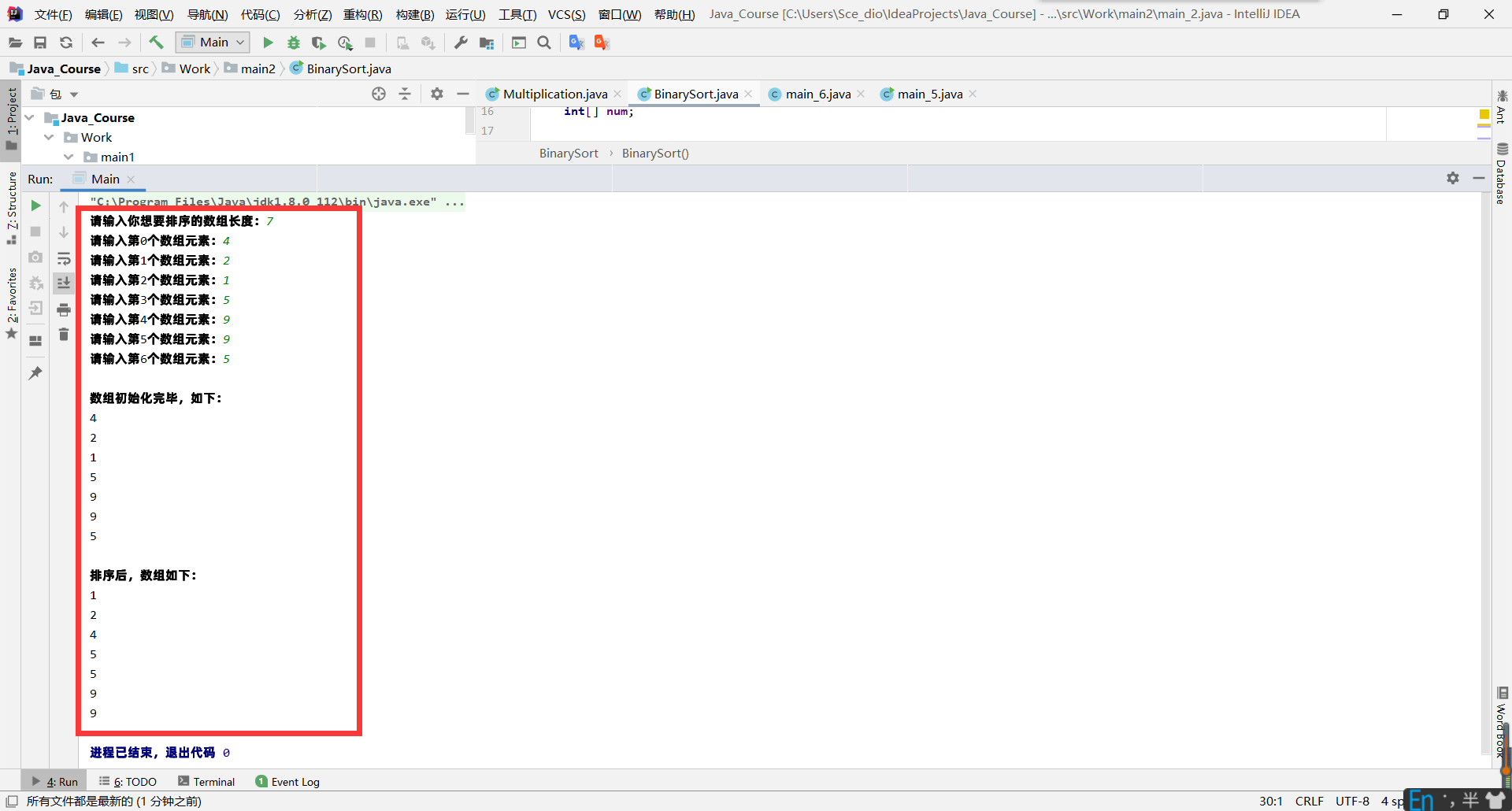
使用JAVA实现基于数组类的二叉排序算法

**二、实验组织运行要求**

每个同学独立完成上述实验要求。

**三、程序**

1. **package** Work.main2;
3. //文件名 Binarysort.java
4. **import** java.util.Scanner;
6. //定义结构体二叉排序，用于排序。
7. **class** BiTree{
8. **int** value;
9. BiTree lnext;
10. BiTree rnext;
11. };
13. //实现数组类
14. **public** **class** BinarySort {
15. BiTree Head;
16. **int** n=1;
17. **int**[] num;
19. BinarySort(){//构造函数，传参，确定需要排序的数组
20. System.out.print("请输入你想要排序的数组长度：");
21. Scanner scanner=**new** Scanner(System.in);
22. n=scanner.nextInt();
23. num=**new** **int**[n];
24. **int** i=0;
25. **while**(i<n){
26. System.out.print("请输入第"+i+"个数组元素：");
27. num[i]=scanner.nextInt();
28. i++;
29. }
30. System.out.println("\n数组初始化完毕，如下：");
31. **for**(i=0;i<n;i++){
32. System.out.println(num[i]+"\t");
33. }
34. }
36. **public** **void** createTree(){//二叉排序树的非递归实现
37. Head=**new** BiTree();
38. Head.value=num[0];
39. Head.lnext=**null**;
40. Head.rnext=**null**;
41. **for** (**int** i = 1; i < n; i++){
42. InsertNode(Head,num[i]);
43. }
44. }
46. **public** **void** InsertNode(BiTree p,**int** num) {//二叉排序树的非递归实现
47. **if**(num<=p.value){
48. **if**(p.lnext==**null**){
49. BiTree q=**new** BiTree();
50. q.value=num;
51. q.lnext=**null**;
52. q.rnext=**null**;
53. p.lnext=q;
54. }
55. **else** InsertNode(p.lnext,num);
56. }
57. **if**(num>p.value){
58. **if**(p.rnext==**null**){
59. BiTree q=**new** BiTree();
60. q.value=num;
61. q.lnext=**null**;
62. q.rnext=**null**;
63. p.rnext=q;
64. }
65. **else** InsertNode(p.rnext,num);
66. }
67. }
69. **public** **void** printStatus(BiTree p){//中序遍历
70. **if**(p.lnext!=**null**){
71. printStatus(p.lnext);
72. }
73. System.out.println(p.value);
74. **if**(p.rnext!=**null**){
75. printStatus(p.rnext);
76. }
77. }
79. **public** **static** **void** main(String[] args){
80. BinarySort A=**new** BinarySort();
81. A.createTree();
82. System.out.println("\n排序后，数组如下：");
83. A.printStatus(A.Head);
84. }
85. }

**四、实验结果**

**实验三 windows 10计算器仿真**

实验学时：2学时

实验类型：验证

实验要求：必修

**一、实验内容**

尽可能多地实现windows10计算器的功能与外观，实现仿真。

**二、实验组织运行要求**

每个同学独立完成上述实验要求。

**三、程序**

1. **package** Work.main3;
3. //Calculator.java
4. //实现计算器
5. //优势：同时实现键盘监听与Button点击监听
6. **import** javax.swing.\*;
7. **import** java.awt.\*;
8. **import** java.awt.event.ActionEvent;
9. **import** java.awt.event.ActionListener;
10. **import** java.awt.event.KeyAdapter;
11. **import** java.awt.event.KeyEvent;
12. **import** java.text.DecimalFormat;
14. **public** **class** Calculator **implements** ActionListener {
15. **private** JFrame jf=**new** JFrame();
16. **private** String[] temp = {"%","CE","C","Back","1⁄x","X²","√x","÷","7","8","9","\*","4","5","6","-","1","2","3","+","+/-","0",".","="};
17. **private** JButton buttons[] = **new** JButton[temp.length];  //创建Button按钮
18. **private** JTextField resultText = **new** JTextField("0");  //显示计算结果文本框，初始状态为0
20. **private** **boolean** firstDigit = **true**;  // 标志用户按的是否是整个表达式的第一个数字,或者是运算符后的第一个数字
21. **private** **double** resultNum = 0.0000;   // 计算的中间结果
22. **private** String operator = "=";   // 当前运算的运算符（按键"C"时需要将其还原为"="）
23. **private** **boolean** operateValidFlag = **true**;   // 判断操作是否合法
25. Calculator(){//控制生成计算器
26. init();
27. }
29. **public** **void** init(){//初始化计算器界面
30. jf.setSize(500,700);
31. jf.setTitle("Calculaor designed by CSG");
32. jf.setLocation(300,100);
33. jf.setMinimumSize(**new** Dimension(500, 700));
34. Image icon=Toolkit.getDefaultToolkit().getImage("C://Users/Sce\_dio/IdeaProjects/Java\_Course/src/Work/main3/1.jpg");
35. jf.setIconImage(icon);
36. jf.setResizable(**true**);
37. jf.setVisible(**true**);
38. jf.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
40. //设置文本框格式，并将其加入jp\_text面板中
41. resultText.setFont(**new** Font("宋体",Font.BOLD,43));  //设置文本框中文字的字体以及大小，加粗
42. resultText.setHorizontalAlignment(JTextField.RIGHT);  //文本框中的内容采用右对齐方式
43. resultText.setEditable(**true**);  //不能修改结果文本框
44. resultText.setBorder(**null**);  //删除文本框的边框
45. Color color1 = **new** Color(195, 238, 242);    //功能键颜色
46. resultText.setBackground(color1);
47. JPanel jp\_text=**new** JPanel();
48. jp\_text.setLayout(**new** BorderLayout());
49. jp\_text.add(resultText);
51. //设置各按键格式，并将其加入jp\_key面板中
52. JPanel jp\_key = **new** JPanel();
53. jp\_key.setLayout(**new** GridLayout(6, 4, 2, 2));
54. Color color3 = **new** Color(232, 232, 232);  //背景颜色
55. **for**(**int** i = 0; i < 8; i++) {//初始化功能按钮
56. buttons[i] = **new** JButton(temp[i]);
57. buttons[i].setBackground(color3);
58. buttons[i].setForeground(Color.black);
59. buttons[i].setFont(**new** Font(Font.SERIF,Font.PLAIN,18));
60. buttons[i].setBorderPainted(**false**);  //去除按钮的边框
61. jp\_key.add(buttons[i]);
62. }
63. **for**(**int** i = 8; i < temp.length; i++) {//初始化运算符及数字键按钮
64. buttons[i] = **new** JButton(temp[i]);
65. jp\_key.add(buttons[i]);
66. **if**((i+1)%4==0) buttons[i].setBackground(color3);
67. **else** buttons[i].setBackground(Color.white);
68. buttons[i].setForeground(Color.black);
69. buttons[i].setFont(**new** Font(Font.SERIF,Font.PLAIN,18));
70. buttons[i].setBorderPainted(**false**);  //去除按钮的边框
71. }
72. Color color2 = **new** Color(126, 192, 238);  //等于号专属颜色
73. buttons[23].setBackground(color2);  // '='符键用特殊颜色
74. jp\_key.setBackground(color1);
76. // 为各按钮添加事件监听器，都使用同一个事件监听器。
77. **for** (**int** i = 0; i < temp.length; i++) {
78. buttons[i].addActionListener(**this**);
79. }
81. //将面板JPanel加入到JFrame中，jp\_text在上，jp\_key在下
82. jf.getContentPane().add("North", jp\_text);
83. jf.getContentPane().add("Center",jp\_key);
85. //设置两个面板的边框，尽量还原win10计算器
86. jp\_text.setBorder(BorderFactory.createMatteBorder(25,3,1,3,color1));
87. jp\_key.setBorder(BorderFactory.createMatteBorder(6,3,3,3,color1));
89. resultText.addKeyListener(**new** KeyAdapter() {//向文本域添加监听器，以实现监听键盘输入
90. @Override
91. **public** **void** keyReleased(KeyEvent e) {
92. **super**.keyTyped(e);
93. **char** string=e.getKeyChar();
94. String key=String.valueOf(string);
95. System.out.println(key);
96. **if** ("0123456789.".indexOf(key) >= 0) {// 用户按了数字键或者小数点键
97. doNumber(key,**false**);
98. } **else** **if**(key.equals(temp[7])||key.equals(temp[11])||key.equals(temp[15])||key.equals(temp[19])){
99. doOperator2(key,**true**);
100. }
101. }
102. });
103. }
105. @Override
106. **public** **void** actionPerformed(ActionEvent e) {//对button按钮添加监听器
107. String key = e.getActionCommand();
108. **if** (key.equals(temp[3])) {//Back键
109. doBackspace();
110. } **else** **if** (key.equals(temp[1])) {//CE键
111. resultText.setText("0");
112. } **else** **if** (key.equals(temp[2])) {//C键
113. do\_Clear();
114. } **else** **if** ("0123456789.".indexOf(key) >= 0) {// 用户按了数字键或者小数点键
115. doNumber(key,**true**);
116. } **else** **if**(key.equals(temp[0]) || key.equals(temp[4]) || key.equals(temp[5]) ||
117. key.equals(temp[6]) || key.equals(temp[20])) {// 用户按了只需一个数的运算键（求倒数，%，开方，平方，取正负数）
118. doOperator1(key,**false**);
119. } **else** {
120. doOperator2(key,**false**);
121. }
122. }
124. **private** **void** doBackspace() {
125. String text = resultText.getText();
126. **int** i = text.length();
127. **if** (i > 0) {
128. text = text.substring(0, i - 1);  // 退格
129. **if** (text.length() == 0) {//初始化
130. resultText.setText("0");
131. firstDigit = **true**;
132. operator = "=";
133. } **else** {//text已经减少一个字符，赋值，看起来就像退格
134. resultText.setText(text);
135. }
136. }
137. }
139. **private** **void** do\_Clear() {
140. resultText.setText("0");
141. firstDigit = **true**;
142. operator = "=";
143. }
145. **private** **void** doNumber(String key,**boolean** control) {
146. //firstDigit用于判断是否是第一个数，是的话覆盖，不是的话连接
147. **if** (firstDigit) {// 输入的为第一个数，直接设置文本，覆盖0
148. resultText.setText(key);
149. } **else** **if** ((key.equals(".")) && (resultText.getText().indexOf(".") < 0)) {
150. // 输入的是小数点，并且之前没有小数点，则将小数点附在结果文本框的后面
151. resultText.setText(resultText.getText() + ".");
152. } **else** **if** (!key.equals(".")&&control) {
153. // 如果输入的不是小数点，则将数字附在结果文本框的后面。键盘输入的就不用继续附了。
154. resultText.setText(resultText.getText() + key);
155. }
156. firstDigit = **false**;
157. }
159. **private** **void** doOperator1(String command,**boolean** control) {
160. //只有一个操作数类型的操作符
161. **if**(control){//如果是键盘输入的，先把运算符覆盖再说。
162. resultText.setText(resultText.getText());
163. }
164. operator = command;  // 运算符为用户按的按钮
165. **if** (operator.equals("1⁄x")) {   // 倒数运算
166. **if** (resultNum == 0) {
167. operateValidFlag = **false**;
168. resultText.setText("零没有倒数");
169. } **else** {
170. resultNum = 1 / getNumberFromText();
171. }
172. } **else** **if** (operator.equals("√x")) {// 平方根运算
173. **if** (resultNum < 0) {
174. operateValidFlag = **false**;
175. resultText.setText("根号内不能为负");
176. } **else** {
177. resultNum = Math.sqrt(getNumberFromText());
178. }
179. } **else** **if** (operator.equals("X²")) {// 平方运算
180. resultNum = getNumberFromText()\*getNumberFromText();
181. } **else** **if** (operator.equals("%")) {// 百分号运算，除以100
182. resultNum = getNumberFromText() / 100;
183. } **else** **if** (operator.equals("+/-")) {// 正数负数运算
184. resultNum = getNumberFromText() \* (-1);
185. **if** (operateValidFlag) {// 操作合法的情况下，才正常输出数值到屏幕上
186. **long** t1;
187. **double** t2;
188. t1 = (**long**) resultNum;
189. t2 = resultNum - t1;
190. **if** (t2 == 0) {
191. resultText.setText(String.valueOf(t1));
192. } **else** {
193. resultText.setText(String.valueOf(**new** DecimalFormat("0.00").format(resultNum)));
194. }
195. }
196. firstDigit = **true**;
197. operateValidFlag = **true**;
198. }
199. }
201. **private** **void** doOperator2(String command,**boolean** control) {
202. ////有两个操作数类型的操作符
203. **if**(control) doBackspace();
204. **if** (operator.equals("÷")) { // 除法运算
205. // 如果当前结果文本框中的值等于0
206. **if** (getNumberFromText() == 0.0) {
207. operateValidFlag = **false**;  //操作不合法
208. resultText.setText("除数不能为零");
209. } **else** {
210. resultNum /= getNumberFromText();
211. }
212. } **else** **if** (operator.equals("+")) {
213. // 加法运算
214. resultNum += getNumberFromText();
215. } **else** **if** (operator.equals("-")) {
216. // 减法运算
217. resultNum -= getNumberFromText();
218. } **else** **if** (operator.equals("\*")) {
219. // 乘法运算
220. resultNum \*= getNumberFromText();
221. } **else** **if** (operator.equals("=")) {
222. // 赋值运算
223. resultNum = getNumberFromText();
224. }
225. **if** (operateValidFlag) {
226. // 操作合法的情况下，结果为小数保留小数点后4位，整数正常输出
227. **long** t1;
228. **double** t2;
229. t1 = (**long**) resultNum;
230. //判断resultNum是否小数部分为0，若小数部分为0，当整数输出；若小数部分不为0，那就按8位小数输出resultNum
231. t2 = resultNum - t1;
232. **if** (t2 == 0) {
233. resultText.setText(String.valueOf(t1));
234. } **else** {
235. resultText.setText(String.valueOf(**new** DecimalFormat("0.00000000").format(resultNum)));
236. }
237. }
238. operator = command;  //运算符为用户按的按钮
239. firstDigit = **true**;
240. operateValidFlag = **true**;
241. }
243. **private** **double** getNumberFromText() {
244. **double** result = 0;
245. **try** {
246. result = Double.valueOf(resultText.getText()).doubleValue();//textField本来是字符串，将其转为double数值
247. } **catch** (NumberFormatException e) {
248. }
249. **return** result;
250. }
252. **public** **static** **void** main(String[] args){
253. **new** Calculator();
254. }
255. }

**四、实验结果**

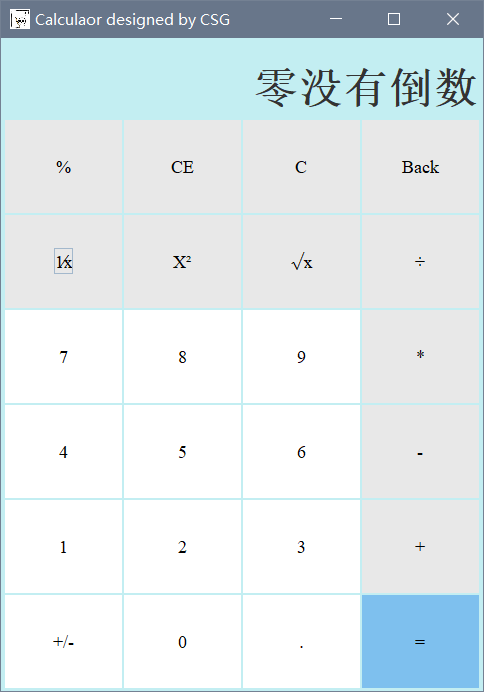
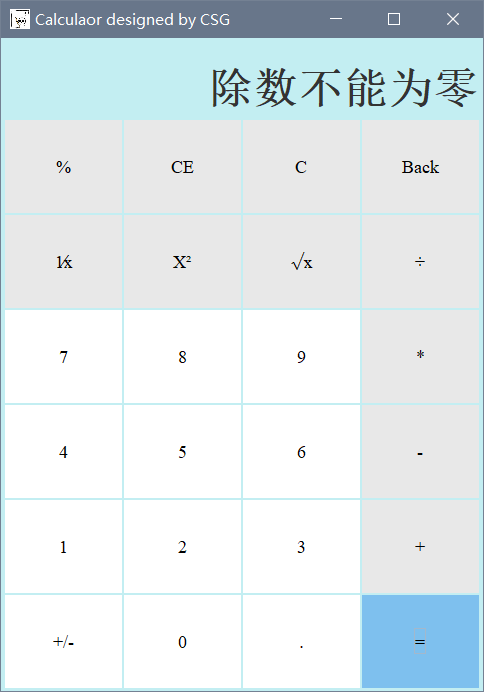
本计算器最大程度实现了对windows 10自带计算器的仿真，其功能特色主要包括：

1)配色模式与设计与windows相仿；

2)存在窗口最小尺寸控制，避免样式怪异；

3)实现了键盘与鼠标点击监听，与windows输入模式相似；

4)能区分较多类型的报错。

主要运行截图如下所示：

**实验四 JAVA网络编程**

实验学时：2学时

实验类型：验证

实验要求：必修

**一、实验内容**

使用JAVA实现发送邮件

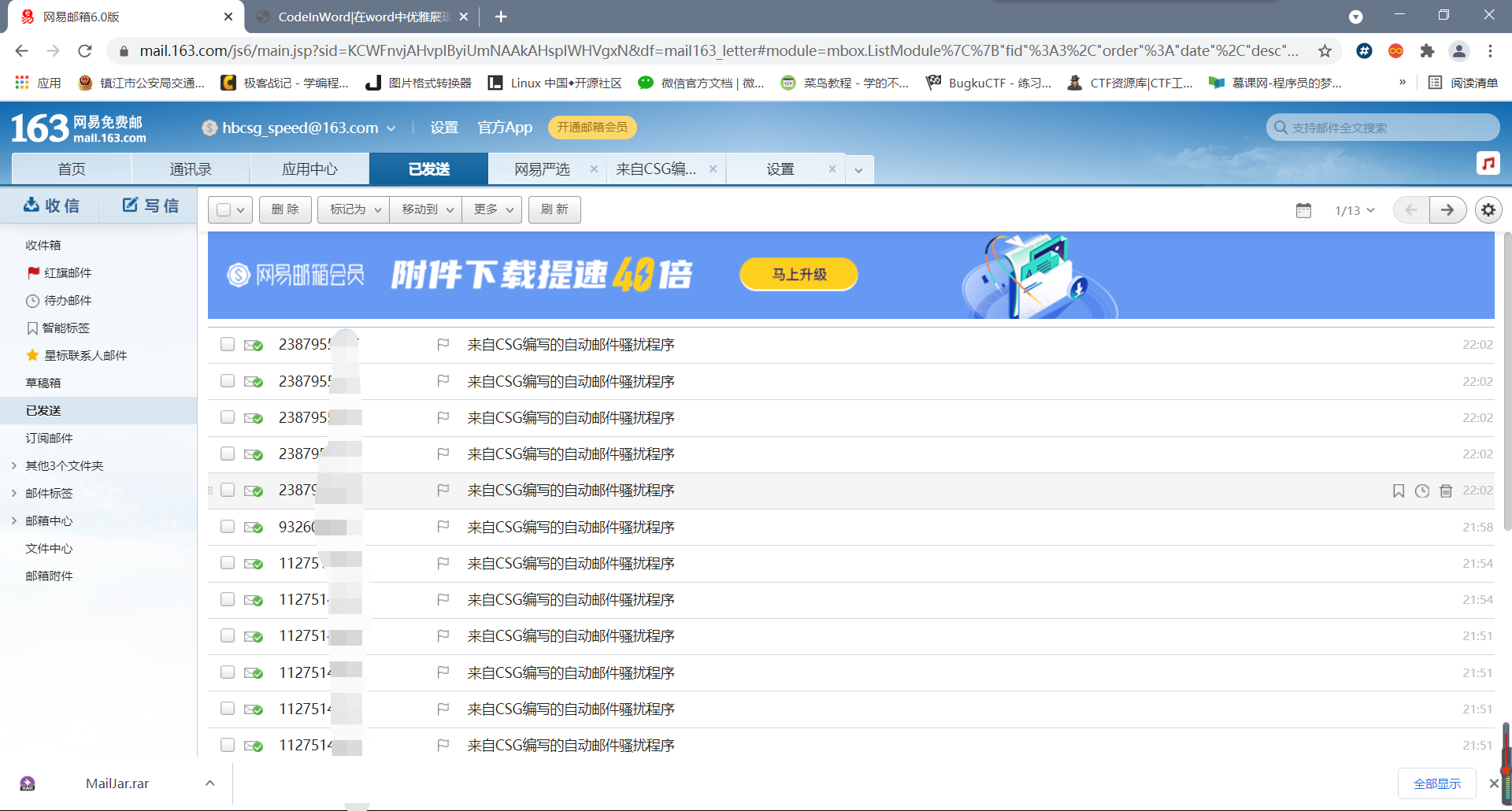
**二、实验组织运行要求**

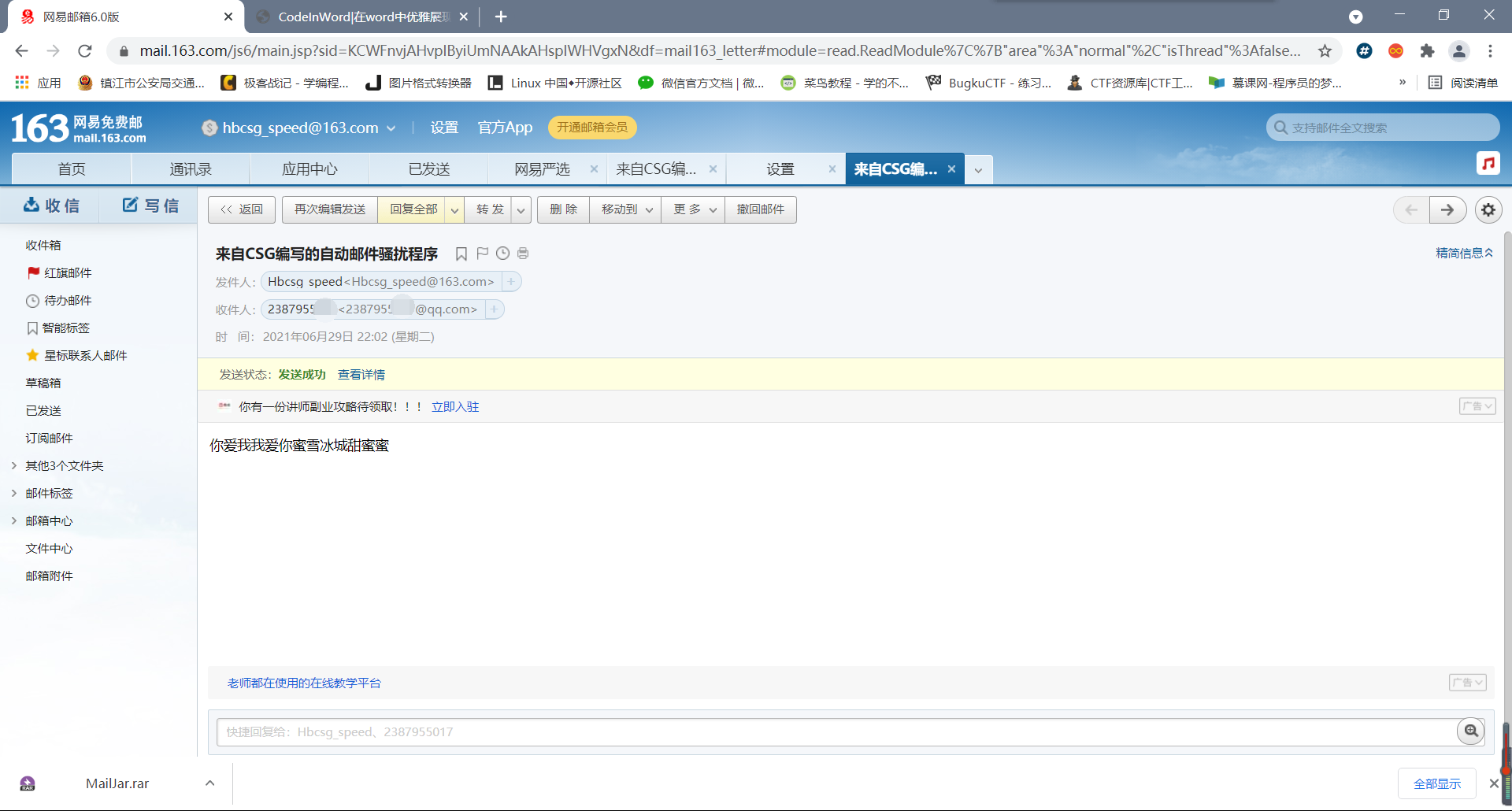
每个同学独立完成上述实验要求。

**三、程序**

1. **package** Work.main4;
3. //SendMail.java
4. //向服务器发送一条信息，实现java发送邮件
6. **import** java.util.Properties;
7. **import** javax.mail.Authenticator;
8. **import** javax.mail.Message;
9. **import** javax.mail.MessagingException;
10. **import** javax.mail.PasswordAuthentication;
11. **import** javax.mail.Session;
12. **import** javax.mail.Transport;
13. **import** javax.mail.internet.InternetAddress;
14. **import** javax.mail.internet.MimeMessage;
16. **public** **class** SendEmail {
17. **public** **static** **void** main(String[] args) {
18. String to="2387955017@qq.com";//收件人邮箱
19. String title="来自CSG编写的自动邮件骚扰程序";//邮件主题
20. String content="你爱我我爱你蜜雪冰城甜蜜蜜";//邮件内容
21. **boolean** flag =**true**;
23. //类Properties（Java.util.Properties），主要用于读取Java的配置文件，方便用户，让用户能够脱离程序本身去修改相关的变量设置。
24. // Step1，创建Properties属性对象，并设置一些邮件的属性。创建键值对，类似于Python字典
25. Properties props = **new** Properties();
26. props.setProperty("mail.host", "smtp.163.com"); // 设置邮箱服务器，利用smtp协议发送到smtp服务器
27. props.setProperty("mail.transport.protocol", "SMTP"); // 设置邮箱发送邮件所使用的协议
28. props.setProperty("mail.smtp.auth", "true"); // 设置认证方式（smtp协议认证方式）
29. props.put("mail.smtp.starttls.enable", "true"); //设置使用ssl连接
31. //Step2，创建认证对象authenticator。其参数是邮件和账号及相应的授权码，以便认证
32. Authenticator authenticator = **new** Authenticator() {
33. **public** PasswordAuthentication getPasswordAuthentication() {
34. **return** **new** PasswordAuthentication("Hbcsg\_speed@163.com", "JYRYQUYDQVFEIDRJ");
35. }
36. };
38. //Step3，创建会话对象session。需要认证与发信息相关字典类型传参
39. Session session = Session.getInstance(props, authenticator);
41. //Step4，创建邮件消息对象，设置发送人、接收人、邮件主题、邮件内容
42. MimeMessage mess = **new** MimeMessage(session);
43. **try** {
44. mess.setFrom(**new** InternetAddress("Hbcsg\_speed@163.com")); // 设置邮件的发件人
45. mess.setRecipients(Message.RecipientType.TO, to); // 设置收件人
46. mess.setSubject(title); // 设置邮件标题
47. mess.setContent(content, "text/html;charset=utf-8"); // 设置邮件内容和格式
48. // Step5，发送邮件
49. **for**(**int** i=0;i<5;i++) {
50. Transport.send(mess);
51. }
52. } **catch** (MessagingException e) {
53. e.printStackTrace();
54. flag=**false**;
55. System.out.println("发送邮件失败, 原因是:" + e.getMessage());
56. }
57. **if**(flag)
58. System.out.println("Congratulations！发送邮件成功！接收人：" + to+"\n主题为："+title+"\n内容为："+content);
59. }
61. }

**四、实验结果**





**实验五 单词的检索**

实验学时：2学时

实验类型：验证

实验要求：必修

**一、实验内容**

使用JAVA检索一段英文单词，粗略统计英文单词个数。

**二、实验组织运行要求**

每个同学独立完成上述实验要求。

**三、程序**

1. **package** Work.main5;
3. //WordCount.java
4. **import** java.io.File;
5. **import** java.util.Scanner;
7. **public** **class** Wordcount {
8. **private** File file;
9. **public** Wordcount(File file) {
10. **this**.file=file;
11. }
13. **public** **int** retrieve(){ //检索单词数量
14. **int** num = 0;
15. **try** {
16. Scanner in = **new** Scanner(file);
17. **while** (in.hasNextLine()) {//逐行读取信息
18. String temp = in.nextLine();
20. String[] row;
21. row = temp.split(" |!|,");//以!、,或空格为分隔符，划分单词
22. **for** (**int** i = 0; i < row.length; i++) {//去掉换行符，将.替换为空格
23. row[i]=row[i].replaceAll("['\r\n']]","");
24. row[i]=row[i].replace('.',' ' );
25. }
26. **for**(**int** i=0;i<row.length;i++) {
27. System.out.println(row[i]);
28. }
29. num += row.length; //逐行累加记录单词数量
30. }
31. in.close();
32. } **catch** (Exception e) {
33. e.printStackTrace();
34. }
35. **return** num;
36. }
38. **public** **static** **void** main(String[] args) {
39. **try** {
40. File file = **new** File("C:\\Users\\Sce\_dio\\IdeaProjects\\Java\_Course\\src\\Work\\main5\\infor.txt");
41. System.out.println("目标文档中的单词数量为：" + **new** Wordcount(file).retrieve());
42. }**catch** (Exception e){
43. e.printStackTrace();
44. }
45. }
46. }

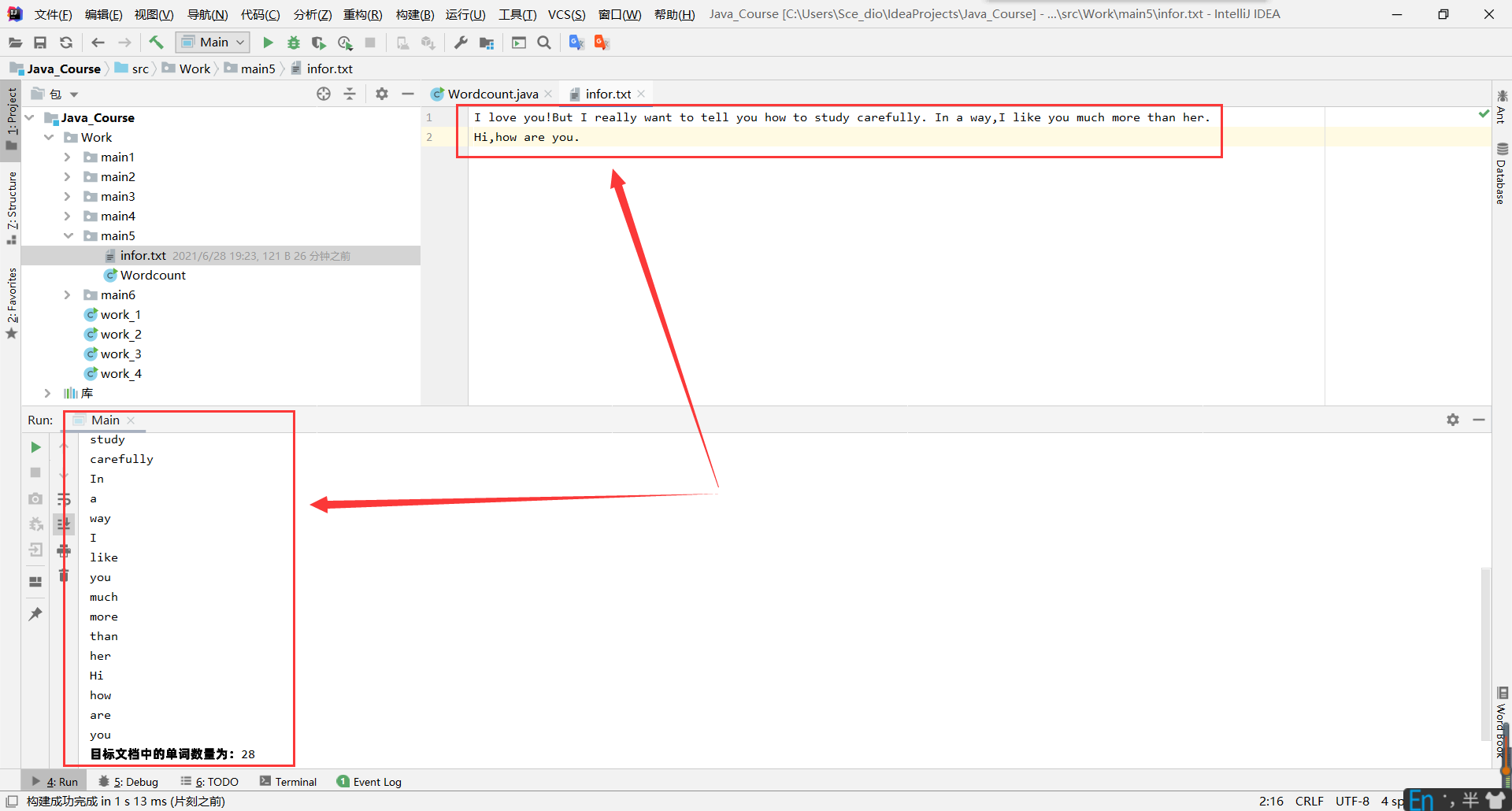
**四、实验结果**

实验所用txt文件内容及检索程序运行情况如下：

所用文本：

I love you!But I really want to tell you how to study carefully. In a way,I like you much more than her.  
Hi,how are you.

实验情况：



**实验六 数据库的连接**

实验学时：2学时

实验类型：验证

实验要求：必修

**一、实验内容**

使用JAVA连接数据库，并实现增删查改两个操作

**二、实验组织运行要求**

每个同学独立完成上述实验要求。

**三、程序**

**1、UserService\_main.java**

1. **package** Work.main6;
3. //写一个sql数据库，增删改查四个操作实现两个即可
4. //UserService\_main.java
5. **import** java.util.List;
6. **import** java.util.Scanner;
8. **public** **class** UserService\_main {
9. **public** **static** **void** main(String[] args) {
10. //初始化一个数据库对象
11. UserDatebase userDatebase = **new** UserDatebase();
12. String name,id;
13. **boolean** flag=**true**;
14. User user;
15. Scanner in =**new** Scanner(System.in);
16. System.out.println("尊敬的用户您好，欢迎来到学籍管理系统。在这里，您可以管理学生信息（仅支持学号、姓名）。");
17. //控制台输出循环
18. **while**(flag){
19. System.out.print("\n功能选项如下所示：\n1.插入学生信息" +
20. "\n2.查询所有学生信息\n3.删除学生信息" +
21. "\n4.修改学生信息\n-1.退出系统\n请选择操作选项：");
22. **int** num =in.nextInt();
23. **switch** (num){
24. //插入
25. **case** 1:
26. System.out.println("请输入添加学生的信息：");
27. System.out.print("学号：");
28. id=in.next();
29. **while**(id.length()!=12){
30. System.out.print("兄弟，学号非法，只能是12位.请重新输入：");
31. id=in.next();
32. }
33. System.out.print("姓名：");
34. name=in.next();
35. user= **new** User(id,name);
36. userDatebase.add(user);
37. **break**;
38. //查询所有信息
39. **case** 2:
40. List<User> list= userDatebase.findAll();
41. **for** (User u : list) {
42. System.out.println(u.getUserId()+u.getUserName());
43. }
44. **break**;
45. //删除
46. **case** 3:
47. System.out.println("请输入删除的学生信息：");
48. System.out.print("学号：");
49. id=in.next();
50. userDatebase.delete(id);
51. System.out.print("删除成功");
52. **break**;
53. **case** 4:
54. System.out.println("请输入修改的学生信息：");
55. System.out.print("学号：");
56. id=in.next();
57. System.out.print("修改后姓名：");
58. name=in.next();
59. user= **new** User(id,name);
60. userDatebase.update(user);
61. **break**;
62. **case** -1:
63. **break**;
64. }
65. }
66. }
67. }

**2、User.java**

1. **package** Work.main6;
3. //User.java
4. **public** **class** User {
5. **private** String userId;
6. **private** String userName;
7. **public** User(String userId, String userName) {
8. **this**.userId = userId;
9. **this**.userName = userName;
10. }
12. **public** String getUserId() {
13. **return** userId;
14. }
16. **public** **void** setUserId(String userId) {
17. **this**.userId = userId;
18. }
20. **public** String getUserName() {
21. **return** userName;
22. }
24. **public** **void** setUserName(String userName) {
25. **this**.userName = userName;
26. }
27. }

**3、JdbcUtil.java**

1. **package** Work.main6;
3. //JdbcUtil.java
4. **import** java.sql.\*;
6. **public** **class** JdbcUtil {
7. **static**{
8. **try** {
9. //加载驱动
10. Class.forName("com.mysql.cj.jdbc.Driver");
11. } **catch** (ClassNotFoundException e) {
12. e.printStackTrace();
13. }
14. }
16. //获取连接
17. **public** **static** Connection getConnection(){
18. Connection connection = **null**;
19. **try** {
20. //获取连接——信息与自己的数据库内容对应
21. connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mysql?useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai", "root", "123456");
22. } **catch** (SQLException e) {
23. e.printStackTrace();
24. }
25. **return** connection;
26. }
28. //关闭数据库对象
29. **public** **static** **void** close(Connection connection, PreparedStatement pstmt, ResultSet rs){
30. **try** {
31. **if**(rs != **null**){
32. rs.close();
33. }
34. **if**(pstmt != **null**){
35. pstmt.close();
36. }
37. **if**(connection != **null**){
38. connection.close();
39. }
40. } **catch** (SQLException e) {
41. e.printStackTrace();
42. }
43. }
44. }

**4、UserDatebase.java**

1. **package** Work.main6;
3. //UserDatebase.java
4. **import** java.sql.\*;
5. **import** java.util.ArrayList;
6. **import** java.util.List;
8. **public** **class** UserDatebase {
10. //添加一个用户信息
11. **public** **void** add(User user){
12. //声明JDBC对象
13. Connection connection = **null**;
14. PreparedStatement pstmt = **null**;
15. **try** {
16. //获取连接
17. connection = JdbcUtil.getConnection();
18. //获取处理器
19. pstmt = connection.prepareStatement("insert into BroFan values(?,?)");
20. //注入参数
21. pstmt.setString(1, user.getUserId());
22. pstmt.setString(2, user.getUserName());
23. //执行新增
24. pstmt.executeUpdate();
25. } **catch** (SQLException e) {
26. e.printStackTrace();
27. } **finally** {
28. JdbcUtil.close(connection, pstmt, **null**);
29. }
30. }
32. **public** **void** update(User user){
33. //声明JDBC对象
34. Connection connection = **null**;
35. PreparedStatement pstmt = **null**;
36. **try** {
37. //获取连接
38. connection = JdbcUtil.getConnection();
39. //获取处理器
40. pstmt = connection.prepareStatement("update BroFan set name=? where userid=?");
41. //注入参数
42. pstmt.setString(2, user.getUserId());
43. pstmt.setString(1, user.getUserName());
44. //执行新增
45. pstmt.executeUpdate();
46. } **catch** (SQLException e) {
47. e.printStackTrace();
48. } **finally** {
49. JdbcUtil.close(connection, pstmt, **null**);
50. }
51. }
53. //查询用户信息
54. **public** User findById(String userId){
55. //声明返回值
56. User user = **null**;
57. Connection connection = **null**;
58. PreparedStatement pstmt = **null**;
59. ResultSet rs = **null**;
60. **try** {
61. //获取连接
62. connection = JdbcUtil.getConnection();
63. //获取处理器
64. pstmt = connection.prepareStatement("select \* from BroFan where userid=?");
65. pstmt.setString(1, userId);
66. //执行查询
67. rs = pstmt.executeQuery();
68. **while**(rs.next()){
69. //创建user对象，用于存储查询的数据
70. user = **new** User(rs.getString(1),rs.getString(2));
71. }
72. } **catch** (SQLException e) {
73. e.printStackTrace();
74. } **finally** {
75. JdbcUtil.close(connection, pstmt, rs);
76. }
77. **return** user;
78. }
80. //查询用户信息
81. **public** List<User> findAll(){
82. //声明返回值
83. List list = **new** ArrayList<User>();
84. User user = **null**;
85. Connection connection = **null**;
86. PreparedStatement pstmt = **null**;
87. ResultSet rs = **null**;
88. **try** {
89. connection = JdbcUtil.getConnection();     //获取连接
90. pstmt = connection.prepareStatement("select \* from BroFan");  //获取处理器
91. rs = pstmt.executeQuery();   //执行查询
92. **while**(rs.next()){//创建user对象并放入list
93. user = **new** User(rs.getString(1),rs.getString(2));
94. list.add(user);
95. }
96. } **catch** (SQLException e) {
97. e.printStackTrace();
98. } **finally** {
99. JdbcUtil.close(connection, pstmt, rs);
100. }
101. **return** list;
102. }
104. //根据编号删除用户
105. **public** String delete(String userId){
106. User user = **null**;
107. Connection connection = **null**;
108. PreparedStatement pstmt = **null**;
109. String result="";
110. **try** {
112. connection = JdbcUtil.getConnection();
113. pstmt = connection.prepareStatement("delete from BroFan where userid=?");
114. pstmt.setString(1, userId);
115. //执行查询
116. result=String.valueOf(pstmt.executeUpdate());
117. } **catch** (SQLException e) {
118. e.printStackTrace();
119. } **finally** {
120. JdbcUtil.close(connection, pstmt,**null**);
121. }
122. **return** result;
123. }
124. }

**四、实验结果**

