****

Java程序设计

实验报告

**学 院：计算机科学与工程学院**

**班 级：信息安全**

**学生姓名：黄佳俊**

**学 号：201836600130**

**指导老师：布社辉**

**提交日期： 2016.12.12**

## 实验1 : 二分法求函数的根

#### 题目：

函数

初始值

二分法：

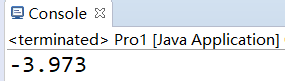
#### 分析：

题目已经给出了二分算法，按照题目要求直接编写即可。

#### 代码

1. **import** java.io.\*;
2. **public** **class** Pro1 {
3. **private** **static** **double** f(**double** x){
4. **return** x \* x \* x - 10 \* x + 23;
5. }
6. **private** **static** **double** my\_abs(**double** x) {
7. **return** x > 0 ? x : -x;
8. }
9. **private** **static** **double** ef(**double** x\_low, **double** x\_high, **double** delta) {
10. **double** x\_c = 0;
11. **do**{
12. x\_c = (x\_high + x\_low) / 2;
13. **if**(f(x\_c) == 0) **return** x\_c;
14. **if**(f(x\_high) \* f(x\_c) < 0) x\_low  = x\_c;
15. **if**(f(x\_low)  \* f(x\_c) < 0) x\_high = x\_c;
16. }**while**(my\_abs(x\_high - x\_low) > delta);
17. **return** x\_c;
18. }
19. **public** **static** **void** main(String argv[]) **throws** IOException{
20. PrintWriter cout = **new** PrintWriter(**new** OutputStreamWriter(System.out));
21. **double** x\_low = -10.0, x\_high = 5.0, delta = 0.001;
22. **if**(f(x\_high) \* f(x\_low) < 0) {
23. **double** x\_c = ef(x\_low, x\_high, delta);
24. cout.printf("%.3f",x\_c);
25. }
26. cout.flush();
27. }
28. }

#### 运行结果



## 实验2 ：求10,000,000内最大的素数

#### 题目：

求10,000,000内最大的素数

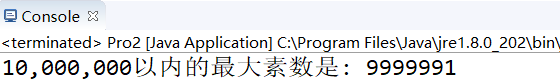
#### 分析：

题目给出了一个参考程序用来求解100以内的全部素数，但是该的算法不便于我们求10,000,000内最大的素数，因此采用欧拉线性筛（每个合数数仅被最小质因数筛去）来获取10,000,000内全部的素数，并获得最大的素数。

#### 代码：

1. **public** **class** Pro2{
2. **public** **static** **void** main(String[] argv) {
3. **final** **int** MAXN = 10000000;
4. **boolean**[] notPrime = **new** **boolean**[MAXN];
5. **int**[] prime = **new** **int**[MAXN];
6. **int** primeCnt = -1;
7. System.out.print("10,000,000以内的最大素数是: ");
8. **for**(**int** i = 2; i < MAXN; i++) {
9. **if**(notPrime[i] == **false**) {
10. prime[++primeCnt] = i;
11. }
12. **for**(**int** j = 0; j <= primeCnt && prime[j] \* i < MAXN; j++) {
13. notPrime[prime[j] \* i] = **true**;
14. **if**(i % prime[j] == 0)
15. **break**;
16. }
17. }
18. System.out.println(prime[primeCnt]);
19. }
20. }

#### 运行结果



## 实验3: K-Means算法实现对数据的聚类分析

#### 题目：

K-Means算法的基本思想是初始随机给定K个簇中心，按照最邻近原则把待分类样本点分到各个簇。

然后按平均法重新计算各个簇的质心，从而确定新的簇心。

一直迭代，直到簇心的移动距离小于某个给定的值。

工作原理是：

选择K个点作为初始质心(随机选择)

Repeat

计算数据集中每个点到各个簇的质心的距离，将数据点分配到距离最近的簇

重新计算每个簇中所有点的均值并将该均值作为质心

Until 簇不发生变化或达到最大迭代次数

- 其中K是用户指定的参数，即所期望的簇的个数。

- 常用的距离度量方法为：欧几里得距离

- 簇的质心都是其均值，即向量各维取平均即可

#### 分析：

题目已经给出了KMeans算法的框架，直接编写成代码即可。

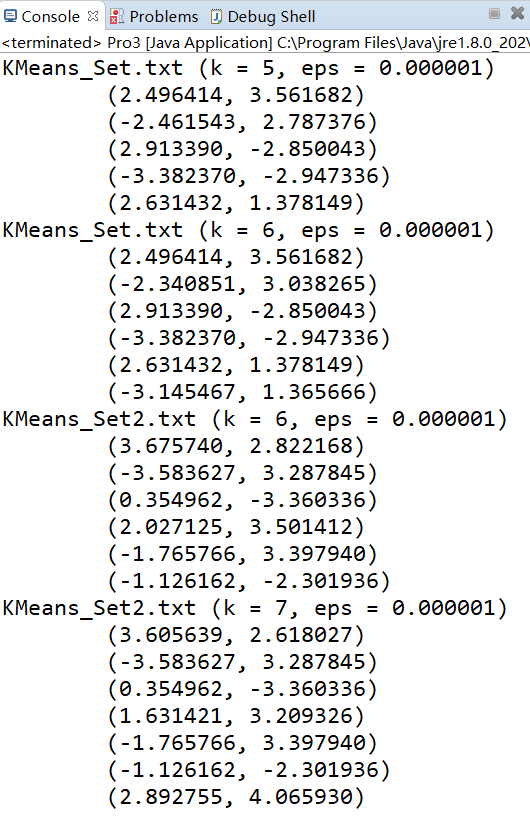
实现时创建了KMeans类负责实现对数据的聚类分析。在构造函数中利用File文件获得数据集； 在dealData中利用指定的k, eps来处理数据。

KMeans类创建了一个内部类Point来存储数据，在本实验中数据都是二维点对，故Poing只包含x,y变量。

#### 代码

1. **import** java.util.\*;
2. **import** java.io.\*;
3. **class** KMeans{
4. **class** Point{
5. Point () { x = 0; y = 0;}
6. Point(**int** x, **int** y) {
7. **this**.x = x;
8. **this**.y = y;
9. }
10. **double** x, y;
11. }
12. **private** String fileName;
13. **private** **int** k;
14. **private** **double** eps;
16. **private** **int** size;
17. **private** Point point[] = **new** Point[100];
18. **private** Point means[] = **new** Point[100];
19. **private** **int** divn[] = **new** **int**[100];
20. **private** **int** div[][] = **new** **int**[100][100];
22. KMeans(String fileName) **throws** FileNotFoundException{
23. **this**.fileName = fileName;
24. File file = **new** File(fileName);
25. Scanner cin = **new** Scanner(file);
27. size = 0;
28. **while**(cin.hasNextDouble()) {
29. point[size] = **new** Point();
30. point[size].x = cin.nextDouble();
31. point[size].y = cin.nextDouble();
32. size++;
33. }
34. cin.close();
35. }
36. **public** **void** dealDate(**int** k, **double** eps) {
37. **this**.k = k;
38. **this**.eps = eps;
39. **for**(**int** i = 0; i < **this**.k; i++) {
40. means[i] = **new** Point();
41. means[i].x = point[i].x;
42. means[i].y = point[i].y;
43. }
45. **while**(divide());
47. System.out.printf("%s (k = %d, eps = %f)\n",fileName, k, eps);
48. **for** (**int** i = 0; i < k; i++)
49. System.out.printf("\t(%f, %f)\n", means[i].x, means[i].y);
50. }
52. **private** **static** **double** getDis(Point a, Point b){
53. **return** Math.sqrt(Math.pow(a.x - b.x, 2) + Math.pow(a.y - b.y, 2));
54. }
55. **private** **boolean** divide() {
56. **boolean** ret = **false**;
57. **for**(**int** i = 0; i < k; i++)
58. divn[i] = 0;
59. **for**(**int** i = 0; i < size; i++) {
60. **double** minDis = getDis(point[i], means[0]);
61. **int** minP = 0;
62. **for**(**int** j = 1;j < k; j++) {
63. **double** dis = getDis(point[i], means[j]);
64. **if**(dis < minDis) {
65. minDis = dis;
66. minP = j;
67. }
68. }
69. div[minP][divn[minP]++] = i;
70. }
71. **for** (**int** i = 0; i < k; i++) {
72. Point tmp = **new** Point(0, 0);
73. **for** (**int** j = 0; j < divn[i]; j++) {
74. tmp.x += point[div[i][j]].x;
75. tmp.y += point[div[i][j]].y;
76. }
77. tmp.x = tmp.x / divn[i];
78. tmp.y = tmp.y / divn[i];
79. **if** (getDis(tmp, means[i]) > eps)
80. ret = **true**;
81. means[i].x = tmp.x;
82. means[i].y = tmp.y;
83. }
84. **return** ret;
85. }
86. }
88. **public** **class** Pro3{
89. **public** **static** **void** main(String[] argv) {
90. **try**{
91. KMeans sol1 = **new** KMeans("KMeans\_Set.txt");
92. sol1.dealDate(5, 0.000001);
93. sol1.dealDate(6, 0.000001);
94. KMeans sol2 = **new** KMeans("KMeans\_Set2.txt");
95. sol2.dealDate(6, 0.000001);
96. sol2.dealDate(7, 0.000001);
97. }**catch**(FileNotFoundException e) {
98. System.out.println("Pro3 : " + e.getMessage());
99. }
100. }
101. }

#### 运行结果



## 实验4: 线性回归(Linear Regression)找到最佳拟合直线

#### 题目

在统计学中，线性回归(Linear Regression)是利用称为线性回归方程的最小平方函数对一个或多个自变量和因变量之间关系进行建模的一种回归分析。

设直线方程为

设数据集为： ​

误差均值为：

若

则

即：

最后根据上述两式可以求出A和B

#### 分析：

以上方程式解出来就是最小二乘法的两个系数，直接套公式求解就好了。

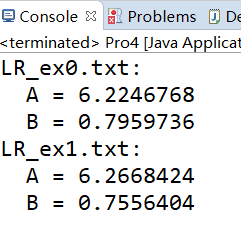
从文件中读取数据，计算相关变量，求解。

但是这道题的测试数据每行给了三个数，其中第一列始终是1，故在程序中取第二列和第三列作为（x,y）

#### 代码：

1. **import** java.io.\*;
2. **public** **class** Pro4{
3. **public** **static** **void** sol(String fileName) **throws** IOException{
4. BufferedReader cin = **new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(fileName)));
5. **int** num = 0;
6. **double** sumXY = 0, sumXX = 0, aveX = 0, aveY = 0;
7. **while**(cin.ready()) {
8. String tmp = cin.readLine();
9. String[] vec = tmp.split("\t");
10. **double** x = Double.valueOf(vec[1]);
11. **double** y = Double.valueOf(vec[2]);
12. ++num;
13. sumXY += x \* y;
14. sumXX += x \* x;
15. aveX += x;
16. aveY += y;
17. }
18. cin.close();
19. aveX /= num;
20. aveY /= num;
21. **double** A = (sumXY - aveX \* aveY) / (sumXX - aveX \* aveX);
22. **double** B = (aveY - A \* aveX);
23. System.out.printf("%s:\n  A = %.7f\n  B = %.7f\n", fileName, A, B);
24. }
25. **public** **static** **void** main(String[] argv) **throws** IOException{
26. sol("LR\_ex0.txt");
27. sol("LR\_ex1.txt");
28. }
29. }

#### 运行结果：



## 实验5：统计学生成绩

#### 题目：

从指定的文本文件读取成绩

获取最高成绩和最低成绩

计算所有学生的平均成绩

分别统计$60​$～$69​$，$70​$～$79​$，$80​$～$89​$，$90～100​$分数段的学生人数

所有结果要输出到一个文本文件

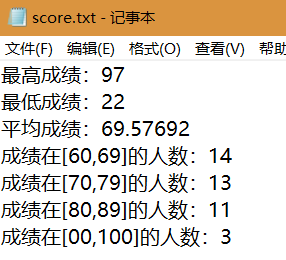
#### 分析：

这题要求我们知道csv文件以”;”作为分隔符。将文件从csv文件读取，并提取有效信息，然后进行处理，完成题目要求，最后用PrintWriter写进score.txt里面。

#### 代码：

1. **import** java.io.\*;
2. **class** Student{
3. **final** **int** MAXN = 100;
4. **int** size;
5. **int** id[], score[];
6. Student(){
7. size = 0;
8. id = **new** **int**[MAXN];
9. score = **new** **int**[MAXN];
10. }
11. **void** append(**int** id,**int** score) {
12. **this**.id[**this**.size] = id;
13. **this**.score[**this**.size] = score;
14. ++**this**.size;
15. }
16. **int** getMax() {
17. **int** maxSorce = score[0];
18. **for**(**int** i = 1; i < size; i++)
19. **if**(score[i] > maxSorce)
20. maxSorce = score[i];
21. **return** maxSorce;
22. }
23. **int** getMin() {
24. **int** minSorce = score[0];
25. **for**(**int** i = 1; i < size; i++)
26. **if**(score[i] < minSorce)
27. minSorce = score[i];
28. **return** minSorce;
29. }
30. **double** getAverage() {
31. **double** ave = 0;
32. **for**(**int** i = 0; i < size; i++)
33. ave += score[i];
34. ave /= size;
35. **return** ave;
36. }
37. **int** getAToB(**int** a,**int** b) {
38. **int** sum = 0;
39. **for**(**int** i = 0; i < size; i++)
40. **if**(a <= score[i] && score[i] <= b)
41. ++sum;
42. **return** sum;
43. }
44. }
45. **public** **class** Pro5{
46. **public** **static** **void** main(String[] agrv) **throws** IOException{
47. BufferedReader cin = **new** BufferedReader(**new** FileReader(**new** File("score.csv")));
48. Student stu = **new** Student();
49. **while**(cin.ready()){
50. String s = cin.readLine();
51. String[] tmp = s.split(";");
52. stu.append(Integer.valueOf(tmp[0]), Integer.valueOf(tmp[1]));
53. }
54. PrintWriter cout = **new** PrintWriter(**new** FileWriter(**new** File("score.txt")));
55. cout.println("最高成绩：" + stu.getMax());
56. cout.println("最低成绩：" + stu.getMin());
57. cout.printf("平均成绩：%.5f" , stu.getAverage()); cout.println("");
58. cout.println("成绩在[60,69]的人数：" + stu.getAToB(60,69));
59. cout.println("成绩在[70,79]的人数：" + stu.getAToB(70,79));
60. cout.println("成绩在[80,89]的人数：" + stu.getAToB(80,89));
61. cout.println("成绩在[00,100]的人数：" + stu.getAToB(90,100));
62. cout.flush();
63. cout.close();
64. }
65. }

#### 运行结果：



## 实验6：图形用户界面

#### 题目：模拟计算器

复习教课书上的模拟计算器的程序例子，并实现该程序

#### 分析：

GUI编程是一项重要技能。Java中awt和swing等可以用于图形用户编程。本实验中主要利用了swing和计算器原理。

calc继承于JFrame，并且使用了Jpanel、JTextField、JButton等类来实现UI。

calcListener实现了ActionEvent接口，用来处理按钮事件的处理，

“0”，“1”“2”“3”“4”“5”“6”“7”“8”“9”“.”按钮触发时屏幕会添加该字符，

“+”“-”“\*”“/”要判断之前位是否也是运算符

“C”清除数据

“CE”去除最后一位，如果没有字符，变为0；

“=”按钮被点击是要对表达式求值。利用了分治思想划分优先级进行表达式求值。

#### 代码:

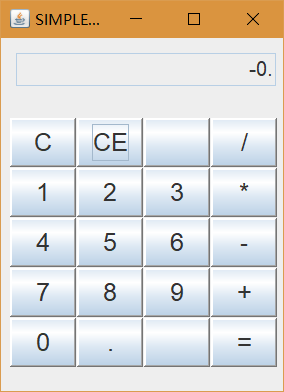
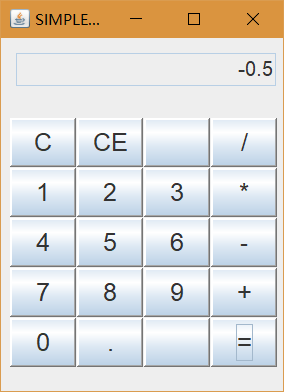
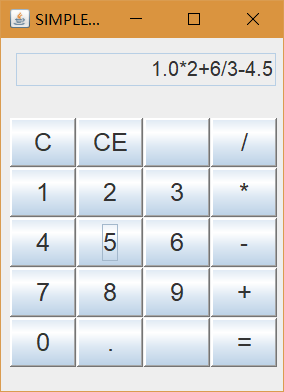
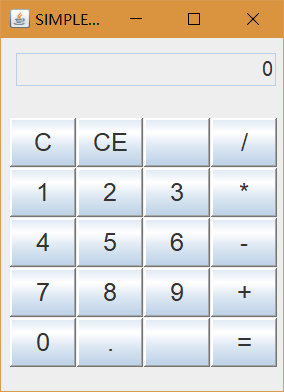
1. **import** java.awt.\*;
2. **import** java.awt.event.ActionEvent;
3. **import** java.awt.event.ActionListener;
5. **import** javax.swing.\*;
6. **class** Calc **extends** JFrame{
7. **private** CalcListener cl;
8. **public** Calc() {
9. **this**.setSize(300,400);
10. **this**.setTitle("SIMPLE\_CALC");
11. **this**.setLocationRelativeTo(**null**);
12. **this**.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
14. JPanel jpanel1 = **new** JPanel();
15. JPanel jpanel2 = **new** JPanel();
16. setLayout(**null**);
17. add(jpanel1);
18. add(jpanel2);
19. layoutPanel1(jpanel1);
20. layoutPanel2(jpanel2);
21. **this**.setVisible(**true**);
22. }
23. **void** layoutPanel1(JPanel panel) {
24. panel.setBounds(-5,10,300,50);
25. JTextField textField = **new** JTextField(16);
26. textField.setHorizontalAlignment(JTextField.RIGHT);
27. textField.setFont(**new** Font("Time\_Roman",Font.PLAIN,20));
28. textField.setEditable(**false**);
29. textField.setText("0");
30. panel.add(textField);
31. cl = **new** CalcListener(textField);
32. }
33. **void** layoutPanel2(JPanel panel) {
34. panel.setBounds(8,80,270,250);
35. panel.setLayout(**new** GridLayout(5,4,1,1));
36. String[] buttonList = {"C","CE"," ","/",
37. "1","2","3","\*",
38. "4","5","6","-",
39. "7","8","9","+",
40. "0","."," ","="};
41. **for** (**int** i = 0; i < buttonList.length; i++) {
42. JButton button = **new** JButton(buttonList[i]);
43. button.setBorder(BorderFactory.createRaisedBevelBorder());
44. button.setFont(**new** Font("Time\_Roman", 0, 25));
45. button.addActionListener(cl);
46. panel.add(button);
47. }
48. }
50. }
51. **class** CalcListener **implements** ActionListener{
52. **private** JTextField textField;
53. **private** String content = "";
54. **public** CalcListener(JTextField textField) {
55. **this**.textField = textField;
56. }
57. @Override
58. **public** **void** actionPerformed(ActionEvent e) {
59. String command = e.getActionCommand();
60. **switch** (command) {
61. **case** "1":**case** "2":**case** "3":
62. **case** "4":**case** "5":**case** "6":
63. **case** "7":**case** "8":**case** "9":
64. **case** "0":**case** ".":
65. content = textField.getText() + command;
66. //System.out.println(content);
67. textField.setText(content);
68. **break**;
69. **case** "+":**case** "-":**case** "\*":**case** "/":
70. content = textField.getText();
71. String lastChar =content.substring(content.length() - 1);
72. //System.out.println(lastChar);
73. **if**(lastChar.compareTo("+") == 0 || lastChar.compareTo("-") ==0 ||
74. lastChar.compareTo("\*") == 0 || lastChar.compareTo("/") == 0) {
75. content = content.substring(0, content.length() - 1) + command;
76. //System.out.println("!");
77. }**else**{
78. content = content + command;
79. //System.out.println("?");
80. }
81. textField.setText(content);
82. **break**;
84. **case** "C":
85. textField.setText("0");
86. **break**;
87. **case** "CE":
88. content = textField.getText();
89. content = content.substring(0, content.length() - 1);
90. **if**(content.length() == 0)
91. content = "0";
92. textField.setText(content);
93. **break**;
95. **case** "=":
96. **double** ans = getAns(**this**.textField.getText());
97. textField.setText(String.valueOf(ans));
98. **break**;
99. **default**:
100. **break**;
101. }
102. }
103. **private** **double** getAns(String s) {
104. System.out.println(s);
105. String[] sAdd = s.split("\\+");
106. **if**(sAdd.length != 1) {
107. **double** sum = 0;
108. **for**(**int** i = 0; i < sAdd.length; i++) {
109. sum += getAns(sAdd[i]);
110. }
111. **return** sum;
112. }**else** {
113. String[] sSub = s.split("\\-");
114. **if**(sSub.length != 1) {
115. **double** sum = getAns(sSub[0]);
116. **for**(**int** i = 1; i < sSub.length; i++) {
117. sum -= getAns(sSub[i]);
118. }
119. **return** sum;
120. }**else** {
121. String[] sMul = s.split("\\\*");
122. **if**(sMul.length != 1) {
123. **double** sum = 1;
124. **for**(**int** i = 0; i < sMul.length; i++) {
125. sum \*= getAns(sMul[i]);
126. }
127. **return** sum;
128. }**else** {
129. String[] sDiv = s.split("\\/");
130. **double** sum = Double.valueOf(sDiv[0]);
131. **for**(**int** i = 1; i < sDiv.length; i++) {
132. sum /= getAns(sDiv[i]);
133. }
134. **return** sum;
135. }
136. }
137. }
138. }
139. }
140. **public** **class** Pro6{
141. **public** **static** **void** main(String[] argv) {
142. Calc calc = **new** Calc();
143. }
144. }

#### 运行结果：

测试“C”按钮

测试：1.0\*2+6/3-4.5

测试“CE”按钮



## 实验7：JDBC方式操作数据库

#### 题目：实现一个简单的信息管理系统

实现如下功能：

- 创建信息表；

- 实现记录的查询，修改，插入，删除；

- 实现记录的统计；

- 纪录数据的导入和导出（Excel文件）

例如

- 以下例子仅供参考，但必须要完成上述所列的基本功能。

- 可根据实际需要作适当的修改，补充。

学生信息管理系统

- 创建student表，包含学生的学号、姓名、年龄信息。

- 根据学号，可以查询到学生的姓名和年龄；

- 给定学生的学号、姓名、年龄，在表中追加一行信息；

- 给定学生的学号，可以从表中删除该学生的信息；

图书信息管理系统

- 创建图书信息表，包含图书的书名、书号、作者、出版日期信息。

- 根据书名，可以查询到图书的书号、作者、出版日期信息；

- 给定图书的书名、书号、作者、出版日期信息，在表中追加一行信息；

- 给定图书的书名，可以从表中删除该图书的信息；

商品信息管理系统

- 创建商品信息表，包含商品的名称、规格、生产厂商、价格信息。

- 根据商品的名称，可以查询到商品的、规格、生产厂商、价格信息；

- 给定商品的名称、规格、生产厂商、价格信息，在表中追加一行信息；

- 给定商品的名称，可以从表中删除该商品的信息；

#### 分析：

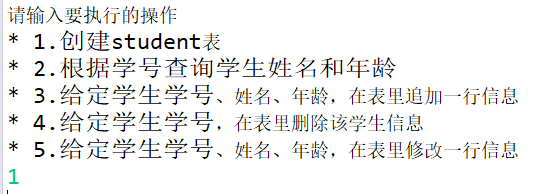
本实验用于巩固数据库处理。选择了第一个学生信息管理系统作为实验内容。

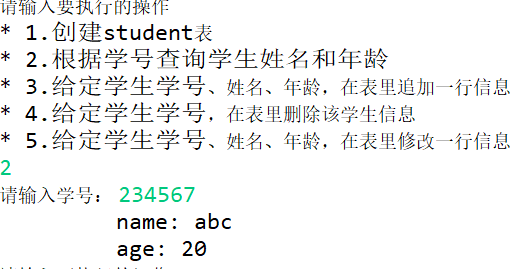
首先程序要连接数据库，这里使用了MySql当做数据源。在网络上下载了“mysql-connector-java-8.0.16.jar”包，利用Class.forName()注册驱动。然后就是利用DriverManager.getConnection()连接数据库生成Connection对象实例，利用createStatement()创建Statement对象实例，利用Statement对象与MySql数据库进行交互，实现实验要求的功能。

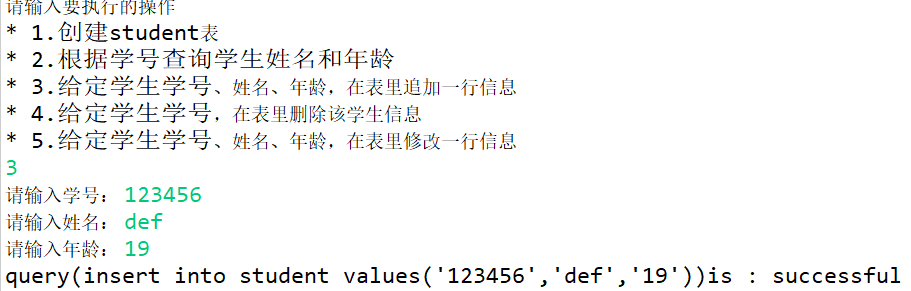
#### 代码：

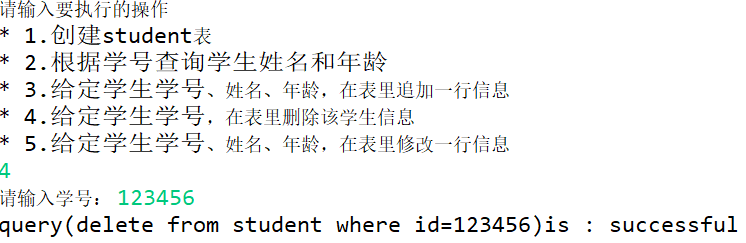
1. **import** java.io.\*;
2. **import** java.sql.\*;
3. **import** java.util.Scanner;
4. **class** MySql{
5. **final** **static** String JDBC\_DRIVER  = "com.mysql.cj.jdbc.Driver";
6. **final** **static** String DB\_URL = "jdbc:mysql://localhost:3306/student?useSSL=false&serverTimezone=UTC";
7. **final** **static** String DB\_USER = "root";
8. **final** **static** String DB\_PASSWD = "";
9. Connection con;
10. Statement stm;
11. **public** MySql() {
12. init();
13. }
14. **public** **void** init() {
15. **try** {
16. Class.forName(JDBC\_DRIVER);
18. con = DriverManager.getConnection(DB\_URL, DB\_USER, DB\_PASSWD);
20. stm = con.createStatement();
22. }**catch**(java.lang.ClassNotFoundException e){
23. System.out.println("class forName is Error:" + e.getMessage());
24. }**catch**(SQLException e) {
25. System.out.println("getConnection is Error: " + e.getMessage());
26. }
27. }
28. **public** **boolean** createStudentTable() {
29. **boolean** fg = **false**;
30. **try**{
31. String exec = "create table student(" +
32. "ID char(12) not null," +
33. "name char(10) not null," +
34. "age int(3) not null)" +
35. "engine=InnoDB default charset=latin1";
36. fg = stm.execute(exec);
37. }**catch**(SQLException e) {
38. System.out.println("createStudentTable:" + e.getMessage());
39. **return** **false**;
40. }
41. **return** fg;
42. }
43. **public** **boolean** query(String ID) {
44. **try** {
45. String exec = "select \* " +
46. "from student " +
47. "where ID=" + ID;
48. ResultSet ret = stm.executeQuery(exec);
49. ret.beforeFirst();
50. String name;
51. **int** age;
52. **for**(;ret.next();) {
53. name = ret.getString("name");
54. age = Integer.valueOf(ret.getString("age"));
55. System.out.printf("\t name: %s\n \t age: %d\n",name,age);
56. }
57. }**catch**(SQLException e) {
58. System.out.println("query is Error:" + e.getMessage());
59. **return** **false**;
60. }
61. **return** **true**;
62. }
63. **public** **boolean** insert(String ID, String name, **int** age) {
64. **try** {
65. String exec = "insert into student values(" +
66. "'" + ID +"'," +
67. "'" + name + "'," +
68. "'" + age + "')";
69. //System.out.println(exec+"\n");
70. stm.executeUpdate(exec);
71. System.out.println("query(" + exec + ")is : successful");
72. }**catch**(SQLException e) {
73. System.out.println("query is Error:" + e.getMessage());
74. **return** **false**;
75. }
77. **return** **true**;
78. }
79. **public** **boolean** update(String ID, String name, **int** age) {
80. **try** {
81. String exec = "update student set " +
82. "name='" + name + "', " +
83. "age='" + age + "' " +
84. "where id=" + ID;
85. //System.out.println(exec+"\n");
86. stm.executeUpdate(exec);
87. System.out.println("query(" + exec + ")is : successful");
88. }**catch**(SQLException e) {
89. System.out.println("query is Error:" + e.getMessage());
90. **return** **false**;
91. }
92. **return** **true**;
93. }
94. **public** **boolean** delete(String ID) {
95. **try** {
96. String exec = "delete from student where id=" + ID;
97. stm.executeUpdate(exec);
98. System.out.println("query(" + exec + ")is : successful");
99. }**catch**(SQLException e) {
100. System.out.println("query is Error:" + e.getMessage());
101. **return** **false**;
102. }
103. **return** **true**;
104. }
105. }
106. **public** **class** Pro7{
107. **public** **static** **void** main(String[] argv) **throws** IOException{
108. MySql sql = **new** MySql();
109. Scanner cin = **new** Scanner(System.in);
110. **for**(;**true**;) {
111. System.out.println("请输入要执行的操作");
112. System.out.println("\* 1.创建student表");
113. System.out.println("\* 2.根据学号查询学生姓名和年龄");
114. System.out.println("\* 3.给定学生学号、姓名、年龄，在表里追加一行信息");
115. System.out.println("\* 4.给定学生学号，在表里删除该学生信息");
116. System.out.println("\* 5.给定学生学号、姓名、年龄，在表里修改一行信息");
117. **int** op = cin.nextInt();
118. String ID, name; **int** age;
119. **switch**(op) {
120. **case** 1: sql.createStudentTable();**break**;
121. **case** 2:
122. System.out.print("请输入学号： ");
123. ID = cin.next();
124. sql.query(ID);
125. **break**;
126. **case** 3:
127. System.out.print("请输入学号： ");
128. ID = cin.next();
129. System.out.print("请输入姓名： ");
130. name = cin.next();
131. System.out.print("请输入年龄： ");
132. age = cin.nextInt();
133. sql.insert(ID,name,age);
134. **break**;
135. **case** 4:
136. System.out.print("请输入学号： ");
137. ID = cin.next();
138. sql.delete(ID);
139. **break**;
140. **case** 5:
141. System.out.print("请输入学号： ");
142. ID = cin.next();
143. System.out.print("请输入姓名： ");
144. name = cin.next();
145. System.out.print("请输入年龄： ");
146. age = cin.nextInt();
147. sql.update(ID,name,age);
148. **break**;
149. **default**:
150. System.out.println("操作参数错误！");
151. }
152. }
154. }
155. }

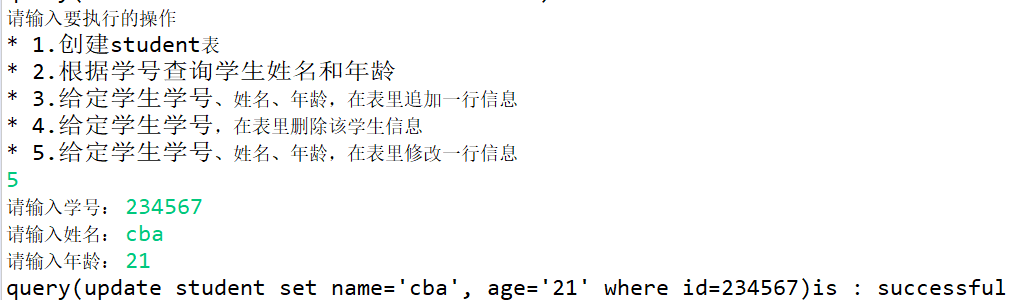
#### 运行结果：

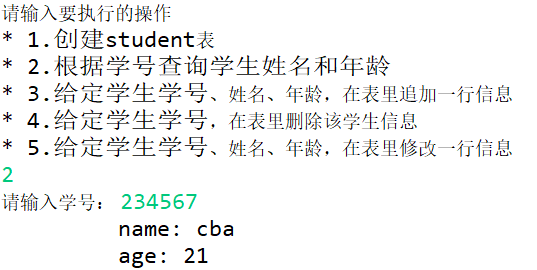












## 实验8 ：网络编程基础

#### 题目：

实现Java爬虫，抓去设定网站的信息

- 设定一家知名的信息网站，例如百度新闻，今日头条，新浪新闻，网易新闻等，

- 使用InetAddress类的方法获取设定网站的主机的IP地址；

- 获取本地机的名称和IP地址。

- 使用URL类下载首页信息，

- 分析首页信息，进行信息过滤，将新闻报道的正文进行分门别类的整理。

#### 分析：

本实验选取了新浪新闻作为数据来源。

这本题中利用了InetAddress类的方法获取设定网站的主机的IP地址和本机IP地址。

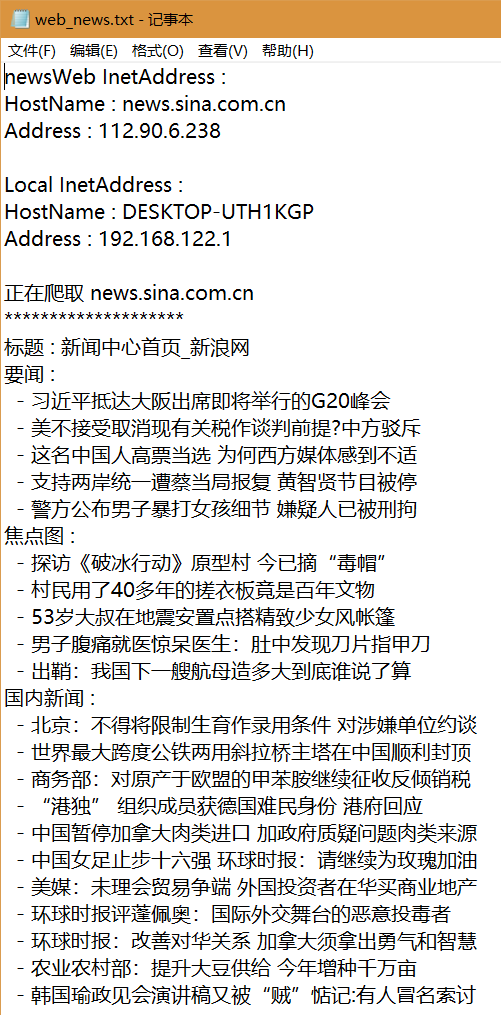
本实验爬虫只需要盘区一个页面，即不用对页面链接做进一步爬取。利用URL类进行网络通讯。利用openStream获得一个InputStream流得到新浪新闻首页的信息。注意协议的填写，不然容易爆302错误。

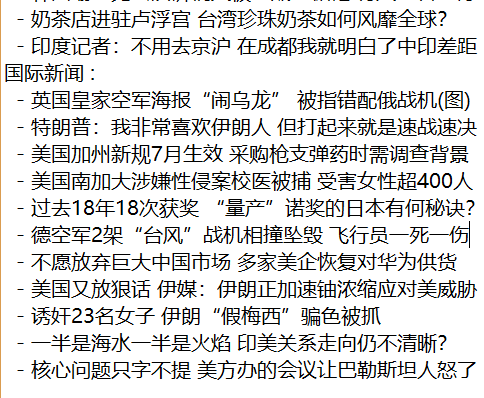
爬取后获得一个页面的html数据保存在String中。利用Jsoup对对象进行解析。依照新浪网页html文件，新闻报道已经分门别类整理过了，我们要将其提取出来。新闻报道的正文的标题都是<h1>,且有超链接。getElementById等方法使用和JS的用法极其类似，在html源码中找到相应的ID名称，利用getElementById获取一个Element对象，然后利用getElementsByTag找<h1>对象，然后把链接<a></a>中的信息输出。

#### 代码：

1. **import** java.io.\*;
2. **import** java.net.\*;
3. **import** org.jsoup.Jsoup;
4. **import** org.jsoup.nodes.Document;
5. **import** org.jsoup.nodes.Element;
6. **class** WebCrawler{
7. PrintWriter cout;
8. String newsWeb = "news.sina.com.cn";
9. **public** **void** run() {
10. **try**{
11. cout = **new** PrintWriter(**new** FileWriter(**new** File("web\_news.txt")));
12. }**catch**(FileNotFoundException e) {
13. System.out.println("run FileNotFoundException :" + e.getMessage());
14. }**catch**(IOException e) {
15. System.out.println("run IOException :" + e.getMessage());
16. }
17. getInetAddressInfo();
18. String htmlData = getHtmlData();
19. dealHtmlData(htmlData);
20. cout.close();
21. }
22. **private** **void** getInetAddressInfo() {
23. InetAddress inetAddress;
24. **try**{
25. inetAddress = InetAddress.getByName(newsWeb);
26. cout.println("newsWeb InetAddress : ");
27. cout.println("HostName : "+inetAddress.getHostName());
28. cout.println("Address : "+inetAddress.getHostAddress());
29. cout.println();
30. inetAddress = InetAddress.getLocalHost();
31. cout.println("Local InetAddress : ");
32. cout.println("HostName : "+inetAddress.getHostName());
33. cout.println("Address : "+inetAddress.getHostAddress());
34. cout.println();
35. cout.flush();
36. }**catch** (UnknownHostException e) {
37. System.out.println("run UnknownHostException:" + e.getMessage());
38. }
39. }
40. **private** String getHtmlData() {
41. StringBuffer htmlData = **new** StringBuffer();
42. **try**{
43. cout.println("正在爬取 "+ newsWeb);
44. URL web = **new** URL("https://" + newsWeb );
45. BufferedReader cin = **new** BufferedReader(**new** InputStreamReader(web.openStream(),"utf-8"));
46. String data;
47. **while**((data = cin.readLine()) != **null**)
48. htmlData.append(data + "\n");
49. cin.close();
50. } **catch** (MalformedURLException e){
51. System.out.println("run MalformedURLException:" + e.getMessage());
52. } **catch** (IOException e){
53. System.out.println("run IOException:" + e.getMessage());
54. }
55. **return** htmlData.toString();
56. }
57. **private** **void** dealHtmlData(String htmlData) {
58. cout.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");
59. Document doc = Jsoup.parse(htmlData);
60. Element title = doc.getElementsByTag("head").first().getElementsByTag("title").first();
62. cout.println("标题 : "+title.html());
64. Element block;
65. block = doc.getElementById("blk\_yw\_01");
66. block = block.getElementById("syncad\_1");
67. cout.println("要闻 : ");
68. **for** (Element item : block.getElementsByTag("h1"))
69. cout.println("  - "+item.getElementsByTag("a").first().html());
71. block = doc.getElementById("Blk01\_Focus\_Cont");
72. cout.println("焦点图 : ");
73. **for** (Element item : block.getElementsByTag("div"))
74. cout.println("  - "+item.getElementsByTag("a").first().html());
76. cout.println("国内新闻 : ");
77. block = doc.getElementById("blk\_new\_gnxw");
78. **for** (Element item : block.getElementsByTag("li"))
79. cout.println("  - "+item.getElementsByTag("a").first().html());
81. cout.println("国际新闻 : ");
82. block = doc.getElementById("blk\_gjxw\_01");
83. **for** (Element item : block.getElementsByTag("li"))
84. cout.println("  - "+item.getElementsByTag("a").first().html());
85. cout.flush();
86. }
87. }
89. **public** **class** Pro8{
90. **public** **static** **void** main(String[] args){
91. WebCrawler crawler = **new** WebCrawler();
92. crawler.run();
93. }
94. }

#### 运行结果





## 实验9：二维码的编码和解码

#### 题目：

利用开源的QR库，实现一个简单的二维码的编码和解码应用程序。

实现如下基本功能：

- 将输入的一段字符串或URL转换为一幅对应的二维码图片；

- 从一幅输入的二维码图片之中解析出对应的字符串或URL。

#### 分析：

本实验提供了zxing作为作为二维码解码和编码工具。从github上clone zxing文件夹，其中core和javase里面的com文件夹下有我们需要的类。将其打包成一个zxing.jar文件，并在实验中导入。

编写CreateQRCode和ReadQRCode两个类用于将输入的一段字符串或URL转换为一幅对应的二维码图片和从一幅输入的二维码图片之中解析出对应的字符串或URL。

利用BitMatrix和MatrixToImageWriter实现二维码生成和写入png文件

利用ImageIO、BinaryBitmap和formatReader实现读取文件和二维码解析。

#### 代码：

1. **import** java.io.\*;
2. **import** java.nio.file.\*;
3. **import** java.util.\*;
4. **import** java.awt.image.\*;
5. **import** javax.imageio.\*;
6. **import** com.google.zxing.\*;
7. **import** com.google.zxing.client.j2se.\*;
8. **import** com.google.zxing.common.\*;
9. **import** com.google.zxing.qrcode.decoder.\*;
11. **class** CreateQRCode {
12. **static** **int** width = 500;
13. **static** **int** height = 500;
14. **public** **static** **void** run(String content) {
15. String format="png";
16. HashMap hints=**new** HashMap();
17. hints.put(EncodeHintType.CHARACTER\_SET, "utf-8");
18. hints.put(EncodeHintType.ERROR\_CORRECTION, ErrorCorrectionLevel.M);
19. hints.put(EncodeHintType.MARGIN, 2);
20. **try** {
22. BitMatrix bitMatrix = **new** MultiFormatWriter().encode(content, BarcodeFormat.QR\_CODE, width, height, hints);
23. Path file = **new** File("code.png").toPath();
24. MatrixToImageWriter.writeToPath(bitMatrix, format, file);
25. } **catch** (Exception e) {
26. System.out.println("create is Error:" + e.getMessage());
27. }
28. }
29. }
30. **class** ReadQRCode {
31. **public** **void** run(String fileName){
32. MultiFormatReader formatReader=**new** MultiFormatReader();
33. File file=**new** File(fileName);
34. BufferedImage image;
35. **try** {
36. image = ImageIO.read(file);
37. BinaryBitmap binaryBitmap=**new** BinaryBitmap(**new** HybridBinarizer(**new** BufferedImageLuminanceSource(image)));
38. Result result=formatReader.decode(binaryBitmap);
40. System.out.println("解析结果："+result.toString());
41. System.out.println("二维码格式："+result.getBarcodeFormat());
42. System.out.println("二维码文本内容："+result.getText());
43. }**catch** (IOException e) {
44. System.out.println("read IOException is Error:" + e.getMessage());
45. }**catch** (NotFoundException e) {
46. System.out.println("read NotFoundException is Error:" + e.getMessage());
47. }
48. }
49. }
51. **public** **class** Pro9{
52. **public** **static** **void** main(String[] argv) {
53. Scanner cin = **new** Scanner(System.in);
54. CreateQRCode cORCode = **new** CreateQRCode();
55. ReadQRCode rQRCode = **new** ReadQRCode();
56. String fn;
57. **while**(**true**) {
58. fn = cin.next();
59. cORCode.run(fn);
60. rQRCode.run("code.png");
61. }
62. }
63. }

#### 运行结果：

## 实验10：重复文件的查询

#### 题目

实现如下基本功能：

- 设定需要查询的文件目录；

- 在遍历整个文件目录的同时，获取每个文件的MD5 checksum值；

- 若在遍历工程之中若发现有的MD5 checksum值相同的文件时，将两个文件进行对比，确认两个文件是否相同；

- 若判断两个文件相同时，输出结果提示用户两个重复文件的路径以及文件名。

#### 分析：

利用path类和file类实现遍历，利用MD5生成获取MD5串，并且保存在hashMap里面，如果遇到相同的就比较MD5相同的文件。

#### 代码：

1. import java.io.\*;
2. import java.util.\*;
3. import java.util.Map.Entry;
5. import org.apache.commons.codec.digest.DigestUtils;
7. **public** **class** Pro10{
8. **static** HashMap<String, ArrayList<String>> filesMD5= **new** HashMap<String, ArrayList<String>>();
9. **public** **static** **void** main(String[] argv) {
10. getFolderMethod("test");
11. Set<String> keys= filesMD5.keySet();
12. **for**(String key: keys) {
13. System.out.println(key + ":");
15. ArrayList<String> filesArr = filesMD5.get(key);
17. **for**(**int** i = 0; i < filesArr.size(); ++i)
18. **for**(**int** j = i + 1; j < filesArr.size(); ++j) {
19. String fileName1 = filesArr.get(i);
20. String fileName2 = filesArr.get(j);
21. **if**(isSameFile(fileName1, fileName2))
22. System.out.println(fileName1 + " " + fileName2 +"相同'");
23. }
24. }
25. }
26. **public** **static** **void** getFolderMethod(String path) {
27. File file = **new** File(path);
28. **if** (file.exists()) {
29. File[] files = file.listFiles();
30. **if** (files != null) {
31. **for** (File file2 : files) {
32. **if** (file2.isDirectory()) {
33. System.out.println("文件夹:" + file2.getAbsolutePath());
34. getFolderMethod(file2.getAbsolutePath());
35. } **else** {
36. **try**{
37. System.out.println("文件:" + file2.getAbsolutePath());
38. String md5 = DigestUtils.md5Hex(**new** FileInputStream(file2.getAbsolutePath()));
39. **if**(filesMD5.containsKey(md5) == **false**)
40. filesMD5.put(md5, **new** ArrayList<String>());
41. ArrayList<String> arr = filesMD5.get(md5);
42. arr.add(file2.getAbsolutePath());
43. }**catch**(IOException e){
44. System.out.println("IO is Error:" + e.getMessage());
45. }
46. }
47. }
48. }
49. } **else** {
50. System.out.println("文件不存在!");
51. }
52. }
53. **public**  **static** boolean isSameFile(String fileName1,String fileName2){
54. FileInputStream fis1 = null;
55. FileInputStream fis2 = null;
56. **try** {
57. fis1 = **new** FileInputStream(fileName1);
58. fis2 = **new** FileInputStream(fileName2);
60. **int** len1 = fis1.available();
61. **int** len2 = fis2.available();
62. **if** (len1 == len2) {
63. byte[] data1 = **new** byte[len1];
64. byte[] data2 = **new** byte[len2];
66. fis1.read(data1);
67. fis2.read(data2);
69. **for** (**int** i=0; i<len1; i++)
70. **if** (data1[i] != data2[i])
71. **return** **false**;
72. **return** **true**;
73. } **else** {
74. **return** **false**;
75. }
76. } **catch** (FileNotFoundException e) {
77. e.printStackTrace();
78. } **catch** (IOException e) {
79. e.printStackTrace();
80. } finally {
81. **if** (fis1 != null) {
82. **try** {
83. fis1.close();
84. } **catch** (IOException e) {
85. e.printStackTrace();
86. }
87. }
88. **if** (fis2 != null) {
89. **try** {
90. fis2.close();
91. } **catch** (IOException e) {
92. e.printStackTrace();
93. }
94. }
95. }
96. **return** **false**;
97. }
98. }

#### 运行结果

