Java实验报告（一）

姓名：王维

班级：计算机科学与技术二班

学号：320210942931

一：3.3.1（1）

1. 源码：

|  |
| --- |
| public class Person**{**  //属性  private int age**=**0**;**  private String name**=**"noname"**;**//名字  private char sex**=**'M'**;**//性别  public Person**(){}**  public Person**(**String n**,**int a**,**char s**){**  name **=** n**;**  **if(**a**>=**0**&&**a**<**140**)** age**=**a**;** //数据过滤  **else** age**=**0**;**  **if(**s**==**'M'**)** sex**=**s**;** //数据过滤  **else** sex**=**'F'**;**  **}**  //行为或方法  public void introduceme**(){**  System**.**out**.**println**(**"my name is:"**+**name**+**"\tmt age is:"**+**age**);**  **if(**sex**==**'M'**)** System**.**out**.**println**(**"I am woman!"**);**  **else** System**.**out**.**println**(**"I am woman!"**);**  **}**  //查询姓名name  public String getName**(){**  **return** name**;**  **}**  //设置姓名name  public void setName**(**String n**){**  name **=** n**;**  **}**  //查询年龄age  public int getAge**(){**  **return** age**;**  **}**  //设置年龄age  public void setAge**(**int a**){**  **if(**a**>=**0**&&**a**<**140**)** age **=**a**;**  **else** age **=**0**;**  **}**  //查询性别sex  public char getSex**(){return** sex**;}**  //设置性别sex  public void setSex**(**char s**){**  **if(**s**==**'M'**)** sex**=**'M'**;**  **else** sex**=**'F'**;**  **}**  public boolean equals**(**Person a**){**  **if(this.**name**.**equals**(**a**.**name**)&&this.**age**==**a**.**age**&&this.**sex**==**a**.**sex**)**  **return** **true;**  **else**  **return** **false;**  **}**  public String toString**(){**  **return** name**+**"."**+**sex**+**","**+**age**;**  **}**  **}**  class PersonTest**{**  public static void main**(**String args**[]){**  Person p1**,**p2**;**  p1 **=** **new** Person**(**"张三"**,**28**,**'M'**);**  p2 **=** **new** Person**();**  p2**.**setName**(**"陈红"**);**p2**.**setAge**(**38**);**p2**.**setSex**(**'F'**);**  p1**.**introduceme**();**  p2**.**introduceme**();**  **}**  **}** |

2.运行结果：

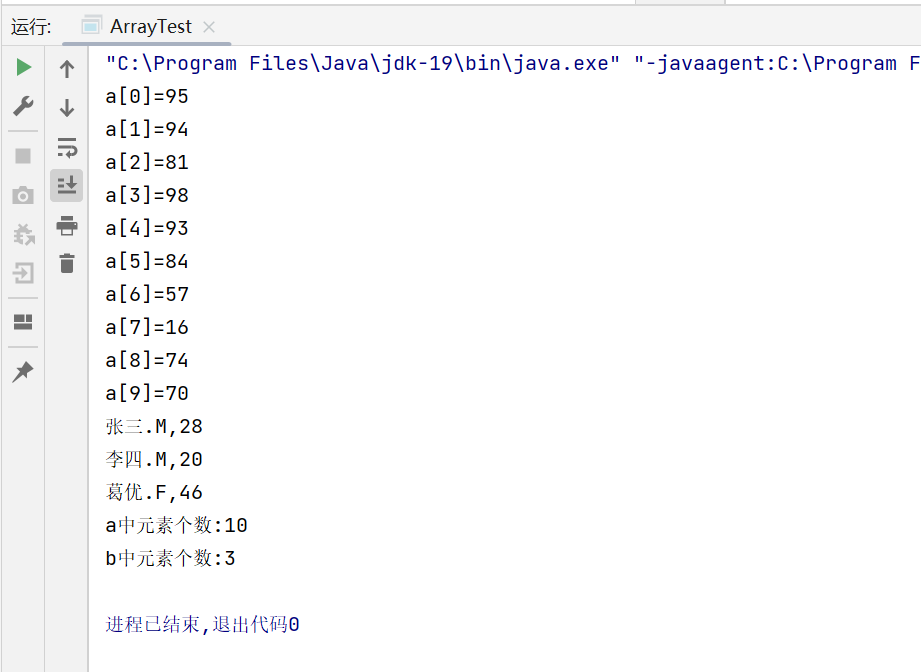


二：3.3.1（2）

1. 源代码：

|  |
| --- |
| public class ArrayTest {  public static void main(String[] args){  int[] a;  Person[] b;  a = new int[10];  b = new Person[3];  for(int i=0;i<10;i++)  {  a[i]=(int)(100 \* Math.*random*());  }   b[0] = new Person("张三",28,'M');  b[1] = new Person("李四",20,'M');  b[2] = new Person();  b[2].setName("葛优");  b[2].setAge(46);  b[2].setSex('F');  for(int i = 0;i<10;i++){  System.*out*.println("a["+i+"]="+a[i]);  }  System.*out*.println(b[0]+"\n"+b[1]+"\n"+b[2]);  System.*out*.println("a中元素个数:"+a.length);  System.*out*.println("b中元素个数:"+b.length);  } } |

1. 运行结果：

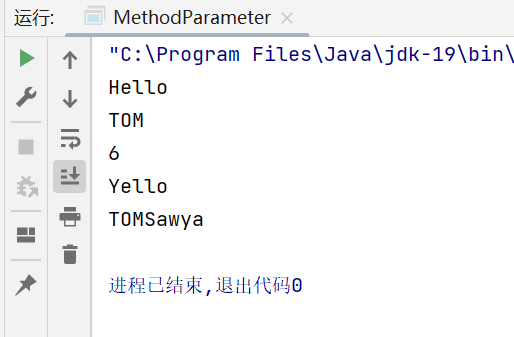


三：3.3.1（3）

1. 源码：

|  |
| --- |
| public class MethodParameter {  public static void main(String[] args) {  int a = 6;  char[] str = new char[]{'H', 'e', 'l', 'l', 'o'};  StringBuffer sb = new StringBuffer("TOM");  *changeAddr*(str, sb);  System.*out*.println(str);  System.*out*.println(sb.toString());  *changeValue*(a, str, sb);  System.*out*.println(a);  System.*out*.println(str);  System.*out*.println(sb.toString());  }   private static void changeAddr(char[] c, StringBuffer sb) {  c = new char[]{'Y', 'e', 'l', 'l', 'o'};  sb = new StringBuffer("SawYer");  }   private static void changeValue(int a, char[] c, StringBuffer sb) {  a = 8;  c[0] = 'Y';  sb.append("Sawya");  } } |

1. 运行结果：

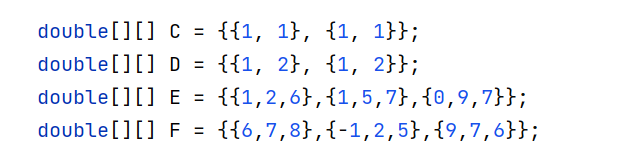


四：设计矩阵类Matrix，实现矩阵的加减法运算以及equals()方法和toString()方法。

1. 设计思路：设计一个矩阵类，及设计一个二维数组，包括其长度，以及宽度（长宽部分所有的变量个数），其中：
2. 构造方法：即输入这个矩阵的长宽，然后新建一个相应的二维数组
3. 加减法：先比较两矩阵能否相加减（即长宽是否相同），若能相加减再每一位对应相加减
4. equals（）方法：由题矩阵以二维数组的形式表示出来，只需调用相应二维数组的equals方法即可，值得注意的是在数组类里equals只对一位数组生效，因此这里调用 *deepEquals*
5. toString方法：同理即输出一个二维数组即可
6. 源代码：

|  |
| --- |
| public class Matrix {  private int row;*//行* private int col;*//列* private double[][] M;  *//构造方法* public Matrix(int row,int col) {  this.col=col;  this.row=row;  M = new double[row][col];  *//初始将矩阵内元素全部赋值为0* for(int i=0;i<row;i++){  for(int j=0;j<col;j++){  M[i][j] = 0;  }  }  }   public int getCol() {  return col;  }   public int getRow(){  return row;  }   public double[][] getM() {  return M;  }   *//为整个矩阵赋值* public void setM(double[][] m) {  M = m;  }   *//为某个矩阵赋值* public void setN(int a,int b,int N){  M[a][b]= N;  }   @Override  *//输出整个矩阵* public String toString() {  return "Matrix{" +  Arrays.*deepToString*(M)+  '}';  }   *//输出矩阵中的某个数* public double getN(int a,int b){  return M[a-1][b-1];  }   public double[][] matrixAdd(Matrix N){  if(N.getCol()!=this.getCol()||N.getRow()!=this.getRow()) {  System.*out*.println("error!");  return null;  }  double[][] K = new double[row][col];  for(int i=0;i<row;i++){  for(int j=0;j<col;j++){  K[i][j] = M[i][j]+N.getM()[i][j];  }  }  return K;  }  *//判断矩阵是否相等* public double[][] matrixMultiy(Matrix N){  if(N.getCol()!=this.getCol()||N.getRow()!=this.getRow()) {  System.*out*.println("error!");  return null;  }  double[][] K = new double[row][col];  for(int i=0;i<row;i++){  for(int j=0;j<col;j++){  K[i][j] = M[i][j]-N.getM()[i][j];  }  }  return K;  }  public boolean equals(Matrix matrix){  if(matrix.getCol()==this.col&&matrix.getRow()==this.row&&Arrays.*deepEquals*(matrix.getM(), this.M)) return true;  else return false;  } }  class MatrixTest{  public static void main(String[] args) {   double[][] C = {{1, 1}, {1, 1}};  double[][] D = {{1, 2}, {1, 2}};  Matrix A = new Matrix(2, 2);  Matrix B = new Matrix(2, 2);  Matrix K = new Matrix(2, 2);   A.setM(C);  B.setM(C);  K.setM(D);   System.*out*.println("A="+A.toString());  System.*out*.println("B="+B.toString());  System.*out*.println("K="+K.toString());  *//这里的矩阵用数组进行表示，直接调用数组的toString方法* System.*out*.print("A+B=");  if(A.matrixAdd(B)!=null) System.*out*.println(Arrays.*deepToString*(A.matrixAdd(B)));   System.*out*.print("A-B=");  if(A.matrixMultiy(B)!=null) System.*out*.println(Arrays.*deepToString*(A.matrixMultiy(B)));  System.*out*.println("A、B是否相等：" + A.equals(B));  System.*out*.println("A、K是否相等：" + A.equals(K));  } } |

1. 测试数据（两个二阶、两个三阶）



1. 运行结果：

