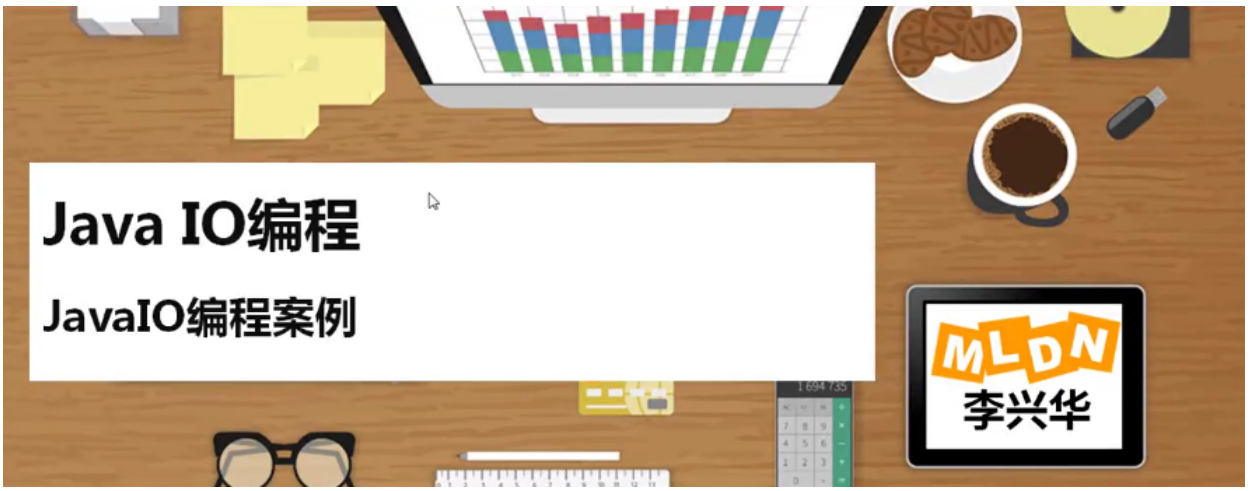

博客： https://www.cnblogs.com/HOsystem/p/14116443.html

2、具体内容

■数字大小比较

编写Java程序，输入3个整数，并求出3个整数的最大值和最小值。

如果要想进行数字的输入处理，那么应该保证输入错误的时候可以重新输入，那么为了可以达到重用的设计，应该准备一个单独的输入数据类。

1、定义一个输入工具类：

```
package cn.mldn.demo.util;
import java.util.Scanner;
public class InputUtil {
	private InputUtil () {}
	/**
	 * 实现键盘接收数字的操作
	 * @param prompt 提示信息
	 * @return 一个可以使用的数字
	 */
	public static int getInt(String prompt) {
		int num = 0 ;
		boolean flag = true ;
		while (flag) {
			Scanner input = new Scanner(System.in) ;
			System.out.print(prompt); // 打印提示信息
			if (input.hasNext("\\d+")) {
				num = Integer.parseInt(input.next("\\d+")) ;
				flag = false ;
			} else {
				System.out.println("输入的内容不是数字！");
			}
		}
		return num ;
	}
}
```

```
package cn.mldn.demo.util;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class InputUtil {
	private InputUtil () {}
	/**
	 * 实现键盘接收数字的操作
	 * @param prompt 提示信息
	 * @return 一个可以使用的数字
	 */
	public static int getInt(String prompt) {
		BufferedReader buf = new BufferedReader(new InputStreamReader(System.in)) ;
		int num = 0 ;
		boolean flag = true ;
		while (flag) {
			System.out.print(prompt); // 打印提示信息
			String str = null ;
			try {
				str = buf.readLine() ;
				if (str.matches("\\d+")) {
					num = Integer.parseInt(str) ;
					flag = false ;
				} else {
					System.out.println("输入的内容不是数字！");
				}
			} catch (IOException e) {
				System.out.println("输入的内容不是数字！");
			}
		}
		return num ;
	}
}
```

2、定义数据的输入处理：

```
package cn.mldn.demo.service;
public interface INumberService {
	/**
	 * 输入数据并且返回输入数据的最大值与最小值
	 * @param count 表示要输入数据的个数
	 * @return 包含有两个内容，第一个是最大值，第二个是最小值
	 */
	public int[] stat(int count) ;
}
```

3、定义接口的实现子类：

```
package cn.mldn.demo.service.impl;
import cn.mldn.demo.service.INumberService;
import cn.mldn.demo.util.InputUtil;
public class NumberServiceImpl implements INumberService {
	@Override
	public int[] stat(int count) {
```

```
            int result [] = new int [2] ; // 定义返回的结果
            int data [] = new int [count] ; // 开辟一个数组
            for (int x = 0 ; x < data.length ; x ++) {  // 数字的循环输入
                  data[x] = InputUtil.getInt("请输入第“" + (x + 1) + "”个数字：");
            }
            result[0] = data[0] ; // 最大值
            result[1] = data[0] ; // 最小值
            for (int x = 0 ; x < data.length ; x ++) {
                  if (data[x] > result[0]) {
                        result[0] = data[x] ;
                  }
                  if (data[x] < result[1]) {
                        result[1] = data[x] ;
                  }
            }
            return result ;
      }
}
```

4、定义工厂类获取接口对象

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.INumberService;
import cn.mldn.demo.service.impl.NumberServiceImpl;
public class Factory {
      private Factory() {}
      public static INumberService getInstance() {
            return new NumberServiceImpl() ;
      }
}
```

5、编写测试程序类

```
package cn.mldn.demo;
import cn.mldn.demo.factory.Factory;
import cn.mldn.demo.service.INumberService;
public class IOCaseDemo {
      public static void main(String[] args) {
            INumberService numberService = Factory.getInstance() ;
            int result [] = numberService.stat(5) ;
            System.out.println("最大值：" + result[0] + "、最小值：" + result[1]);
      }
}
```

■文件保存处理

从键盘输入文件的内容和要保存的文件名称，然后根据输入的名称创建文件，并将内容保存到文件中。

在本程序里面只要求开发者保存的是文件名称而没有设置文件路径，那么对于文件路径就应该在程序启动之前就准备好。

1、定义一个文件操作的服务接口：

```
package cn.mldn.demo.service;
import java.io.File;
public interface IFileService {
	public static final String SAVE_DIR = "D:" + File.separator + "mldndata" + File.separator
;
	/**
	 * 定义文件的保存处理方法
	 * @return 保存成功返回true，否则返回false
	 */
	public boolean save();
}
```

2、在InputUtil类里面追加有输入字符串的处理方法

```
package cn.mldn.demo.util;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class InputUtil {
	private static final BufferedReader INPUT = new BufferedReader(new
InputStreamReader(System.in)) ;
	private InputUtil () {}
	public static String getString(String prompt) {
		String str = null ;
		boolean flag = true ;
		while(flag) {
			System.out.print(prompt);
			try {
				str = INPUT.readLine() ;
				if (!"".equals(str)) {
					flag = false ;
				} else {
					System.out.println("输入的内容不允许为空！");
				}
			} catch (IOException e) {
				System.out.println("输入的内容不允许为空！");
			}
		}
		return str ;
	}
```

完整代码

```
package cn.mldn.demo.util;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class InputUtil {
	private static final BufferedReader INPUT = new BufferedReader(new
InputStreamReader(System.in)) ;
	private InputUtil () {}
```

```
	public static String getString(String prompt) {
		String str = null ;
		boolean flag = true ;
		while(flag) {
			System.out.print(prompt);
			try {
				str = INPUT.readLine() ;
				if (!"".equals(str)) {
					flag = false ;
				} else {
					System.out.println("输入的内容不允许为空！");
				}
			} catch (IOException e) {
				System.out.println("输入的内容不允许为空！");
			}
		}
		return str ;
	}
	/**
	 * 实现键盘接收数字的操作
	 * @param prompt 提示信息
	 * @return 一个可以使用的数字
	 */
	public static int getInt(String prompt) {
		int num = 0 ;
		boolean flag = true ;
		while (flag) {
			System.out.print(prompt); // 打印提示信息
			String str = null ;
			try {
				str = INPUT.readLine() ;
				if (str.matches("\\d+")) {
					num = Integer.parseInt(str) ;
					flag = false ;
				} else {
					System.out.println("输入的内容不是数字！");
				}
			} catch (IOException e) {
				System.out.println("输入的内容不是数字！");
			}
		}
		return num ;
	}
}



package cn.mldn.demo.service;
import java.io.File;
public interface IFileService {
	public static final String SAVE_DIR = "D:" + File.separator + "mldndata" + File.separator
;
```

```
	/**
	 * 定义文件的保存处理方法
	 * @return 保存成功返回true，否则返回false
	 */
	public boolean save();
}



package cn.mldn.demo.service.impl;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.PrintWriter;
import cn.mldn.demo.service.IFileService;
import cn.mldn.demo.util.InputUtil;
public class FileServiceImpl implements IFileService {
	private String name ;
	private String content ;
	public FileServiceImpl() {
		this.name = InputUtil.getString("请输入保存文件名称：") ;
		this.content = InputUtil.getString("请输入保存文件的内容：") ;
	}
	@Override
	public boolean save() {
		File file = new File(IFileService.SAVE_DIR + this.name) ;
		PrintWriter out = null ;
		try {
			out = new PrintWriter(new FileOutputStream(file));
			out.print(this.content);
		} catch (FileNotFoundException e) {
			return false ;
		} finally {
			if (out != null) {
				out.close();
			}
		}
		return true;
	}

}



package cn.mldn.demo.factory;
import cn.mldn.demo.service.IFileService;
import cn.mldn.demo.service.impl.FileServiceImpl;
public class Factory {
	private Factory() {}
	public static IFileService getInstance() {
		return new FileServiceImpl() ;
	}
```

```
}



package cn.mldn.demo;
import java.io.File;
import cn.mldn.demo.factory.Factory;
import cn.mldn.demo.service.IFileService;
public class IOCaseDemo {
	static {	// 项目启动的时候该路径应该首先创建
		File file = new File(IFileService.SAVE_DIR) ;	// 路径，但是这个文件目录有可能不存
在
		if (!file.exists()) {	// 文件目录不存在
			file.mkdirs() ; // 创建目录
		}
	}
	public static void main(String[] args) {
		IFileService fileService = Factory.getInstance() ;
		System.out.println(fileService.save());
	}
}
```

■字符串逆序显示

从键盘传入多个字符串到程序中，并将它们按逆序输出在屏幕上。

本程序之中应该考虑到如下的几种设计：

·既然字符串的内容可以随时修改，那么最好建立一个StringBuffer做保存；

·在进行数据处理的时候应该由用户自己来决定是否继续输入；

1、定义字符串的操作标准：

```
package cn.mldn.demo.service;
public interface IStringService {
	public void append(String str) ; // 追加数据
	public String[] reverse() ; // 反转处理
}
```

2、实现子类里面就使用StringBuffer来操作：

```
package cn.mldn.demo.service.impl;
import cn.mldn.demo.service.IStringService;
public class StringServiceImpl implements IStringService {
	private StringBuffer data = new StringBuffer() ;
	@Override
	public void append(String str) {
		this.data.append(str).append("|") ;
	}

	@Override
	public String[] reverse() {
```

```
            String result [] = this.data.toString().split("\\|") ;
            int center = result.length / 2 ;
            int head = 0 ;
            int tail = result.length - 1 ;
            for (int x = 0 ; x < center ; x ++) {
                  String temp = result[head] ;
                  result[head] = result[tail] ;
                  result[tail] = temp ;
            }
            return result ;
      }
}
```

3、定义工厂类：

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.IStringService;
import cn.mldn.demo.service.impl.StringServiceImpl;
public class Factory {
      private Factory() {}
      public static IStringService getInstance() {
            return new StringServiceImpl() ;
      }
}
```

4、定义一个Menu处理类，采用交互式的界面形式完成处理：

```
package cn.mldn.demo.menu;
import java.util.Arrays;
import cn.mldn.demo.factory.Factory;
import cn.mldn.demo.service.IStringService;
import cn.mldn.demo.util.InputUtil;
public class Menu {
      private IStringService stringService ;
      public Menu() {
            this.stringService = Factory.getInstance() ;
            this.choose();
      }
      public void choose() {
            this.show() ;
            String choose = InputUtil.getString("请进行选择：") ;
            switch(choose) {
            case "1" : {      // 接收输入数据
                  String str = InputUtil.getString("请输入字符串数据：") ;
                  this.stringService.append(str); // 进行数据的保存
                  choose() ;        // 重复出现
            }
            case "2" : {      // 逆序显示数据
                  String result [] = this.stringService.reverse() ;
                  System.out.println(Arrays.toString(result)); // 输出
                  choose() ;        // 重复出现
            }
            case "0" : {
                  System.out.println("下次再见，拜拜！");
```

```
                System.exit(1);
            }
            default : {
                System.out.println("您输入了非法的选项，无法进行处理，请确认后再次执行程
序！");
                choose() ;
            }
            }
        }
        public void show() {
            System.out.println("【1】追加字符串数据\n");
            System.out.println("【2】逆序显示所有的字符串数据\n");
            System.out.println("【0】结束程序执行。");
            System.out.println("\n\n\n");
        }
}
```

5、编写程序测试类

```
package cn.mldn.demo;
import cn.mldn.demo.menu.Menu;
public class IOCaseDemo {
        public static void main(String[] args) {
            new Menu() ;  // 启动程序界面
        }
}
```

完整代码

```
package cn.mldn.demo.service;
public interface IStringService {
        public void append(String str) ; // 追加数据
        public String[] reverse() ; // 反转处理
}



package cn.mldn.demo.factory;
import cn.mldn.demo.service.IStringService;
import cn.mldn.demo.service.impl.StringServiceImpl;
public class Factory {
        private Factory() {}
        public static IStringService getInstance() {
            return new StringServiceImpl() ;
        }
}



package cn.mldn.demo.service.impl;
import cn.mldn.demo.service.IStringService;
public class StringServiceImpl implements IStringService {
        private StringBuffer data = new StringBuffer() ;
```

```
	@Override
	public void append(String str) {
		this.data.append(str).append("|") ;
	}

	@Override
	public String[] reverse() {
		String result [] = this.data.toString().split("\\|") ;
		int center = result.length / 2 ;
		int head = 0 ;
		int tail = result.length - 1 ;
		for (int x = 0 ; x < center ; x ++) {
			String temp = result[head] ;
			result[head] = result[tail] ;
			result[tail] = temp ;
		}
		return result ;
	}
}



package cn.mldn.demo.menu;
import java.util.Arrays;
import cn.mldn.demo.factory.Factory;
import cn.mldn.demo.service.IStringService;
import cn.mldn.demo.util.InputUtil;
public class Menu {
	private IStringService stringService ;
	public Menu() {
		this.stringService = Factory.getInstance() ;
		this.choose();
	}
	public void choose() {
		this.show() ;
		String choose = InputUtil.getString("请进行选择：") ;
		switch(choose) {
		case "1" : {	// 接收输入数据
			String str = InputUtil.getString("请输入字符串数据：") ;
			this.stringService.append(str); // 进行数据的保存
			choose() ;	// 重复出现
		}
		case "2" : {	// 逆序显示数据
			String result [] = this.stringService.reverse() ;
			System.out.println(Arrays.toString(result)); // 输出
			choose() ;	// 重复出现
		}
		case "0" : {
			System.out.println("下次再见，拜拜！");
			System.exit(1);
		}
		default : {
```

```
                System.out.println("您输入了非法的选项，无法进行处理，请确认后再次执行程
序！");
                choose() ;
            }
            }
        }
        public void show() {
            System.out.println("【1】追加字符串数据\n");
            System.out.println("【2】逆序显示所有的字符串数据\n");
            System.out.println("【0】结束程序执行。");
            System.out.println("\n\n\n");
        }
}



package cn.mldn.demo;
import cn.mldn.demo.menu.Menu;
public class IOCaseDemo {
        public static void main(String[] args) {
            new Menu() ;  // 启动程序界面
        }
}
```

■ 数据排序处理

从键盘输入以下的数据：“TOM：89 | JERRY：90 | TONY:95”，数据格式为“姓名：成绩 | 姓名：成绩 | 姓名：成绩”，对输入的内容按成绩进行排序，并将排序结果按照成绩由高到低排序。

对于排序的处理肯定使用Comparable接口完成，同时利用Arrays类来处理，这里面唯一不同的地方就在于此时的数据显示需要通过键盘输入。

1、建立Student的程序类，并且进行排序规则的配置：

```
package cn.mldn.demo.vo;
public class Student implements Comparable<Student> {
        private String name;
        private double score;
        public Student(String name,double score) {
            this.name = name ;
            this.score = score ;
        }
        public String toString() {
            return "姓名：" + this.name + "、成绩：" + this.score ;
        }
        @Override
        public int compareTo(Student obj) {
            if (this.score > obj.score) {
```

```
            return -1 ;
        } else if (this.score < obj.score) {
            return 1 ;
        } else {
            return 0 ;
        }
    }
}
```

2、建立数据的输入处理操作，因为牵扯到拆分问题：

```
package cn.mldn.demo.service;
import cn.mldn.demo.vo.Student;
public interface IStudentService {
    public Student[] getData() ; // 获取排序数据
}
```

3、建立IStudentService子类

```
package cn.mldn.demo.service.impl;
import java.util.Arrays;
import cn.mldn.demo.service.IStudentService;
import cn.mldn.demo.vo.Student;
public class StudentServiceImpl implements IStudentService{
    private String content;
    private Student [] students;
    public StudentServiceImpl(String content) {
        this.content = content ;
        this.handle() ; //進行數據處理
    }
    private void handle() {//進行字符串數據的處理操作
        String result[] = this.content.split("\\|") ; //拆分數據
        this.students = new Student[result.length] ;
        for(int x = 0 ; x < result.length ; x++) {
            String temp [] = result[x].split(":");
            this.students[x] = new Student(temp[0],Double.parseDouble(temp[1]));
        }

    }
    @Override
    public Student[] getData() {
        Arrays.sort(this.students);
        return this.students;
    }
}
```

4、定义Factory工厂类

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.IStudentService;
import cn.mldn.demo.service.impl.StudentServiceImpl;
import cn.mldn.demo.util.InputUtil;
public class Factory {
    private Factory() {}
    public static IStudentService getInstance() {

```

```
            return new StudentServiceImpl(InputUtil.getString("请输入数据信息："));
        }
}
```

5、编写程序测试类

```
package cn.mldn.demo;
import java.util.Arrays;
import cn.mldn.demo.factory.Factory;
public class IOCaseDemo {
        public static void main(String[] args) {
            System.out.println(Arrays.toString(Factory.getInstance().getData()));
        }
}
```

```
Tom:89.1|Jerry:90.01|qiangqiang:59.9
```

完整代码：

```
package cn.mldn.demo.vo;
public class Student implements Comparable<Student> {
        private String name;
        private double score;
        public Student(String name,double score) {
            this.name = name ;
            this.score = score ;
        }
        public String toString() {
            return "姓名：" + this.name + "、成绩：" + this.score ;
        }
        @Override
        public int compareTo(Student obj) {
            if (this.score > obj.score) {
                return -1 ;
            } else if (this.score < obj.score) {
                return 1 ;
            } else {
                return 0 ;
            }
        }
}



package cn.mldn.demo.service.impl;
import java.util.Arrays;
import cn.mldn.demo.service.IStudentService;
import cn.mldn.demo.vo.Student;
public class StudentServiceImpl implements IStudentService{
        private String content;
        private Student [] students;
        public StudentServiceImpl(String content) {
            this.content = content ;
```

```
            this.handle() ; //進行數據處理
        }
        private void handle() {//進行字符串數據的處理操作
            String result[] = this.content.split("\\|") ; //拆分數據
            this.students = new Student[result.length] ;
            for(int x = 0 ; x < result.length ; x++) {
                String temp [] = result[x].split(":");
                this.students[x] = new Student(temp[0],Double.parseDouble(temp[1]));
            }

        }
        @Override
        public Student[] getData() {
            Arrays.sort(this.students);
            return this.students;
        }
}


package cn.mldn.demo.service;
import cn.mldn.demo.vo.Student;
public interface IStudentService {
        public Student[] getData() ; // 获取排序数据
}



package cn.mldn.demo.factory;
import cn.mldn.demo.service.IStudentService;
import cn.mldn.demo.service.impl.StudentServiceImpl;
import cn.mldn.demo.util.InputUtil;
public class Factory {
        private Factory() {}
        public static IStudentService getInstance() {
            return new StudentServiceImpl(InputUtil.getString("请输入数据信息："));
        }
}



package cn.mldn.demo;
import java.util.Arrays;
import cn.mldn.demo.factory.Factory;
public class IOCaseDemo {
        public static void main(String[] args) {
            System.out.println(Arrays.toString(Factory.getInstance().getData()));
        }
}
Tom:89.1|Jerry:90.01|qiangqiang:59.9
```

■数据排序处理深入

将第4题中的内容进行扩展，可以将全部输入的信息保存在文件中，还可以添加信息，并可以显示全部的数据。

如果此时要进行内容的保存，那么首先一定要确认所有输入数据的保存位置，所有的数据之间如果要想沿用之前的设计的结构，则数据文件里面的保存应该做到格式统一，即：“姓名:成绩|”的形式进行存储，而在数据添加的时候可以添加两类数据：“单独的内容”，“一组内容”。还有一个前提：暂时不去考虑数据过大的问题；

1、设置一个文件的处理类，该类之中只是提供有数据的增加以及读取：

```
package cn.mldn.demo.util;
import java.io.File;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;
public class FileUtil {
	public static String load(File file) {
		Scanner scan = null;
		try {
			scan = new Scanner(file) ; // 文件加载
			if (scan.hasNext()) {
				String str = scan.next(); // 获取数据
				return str ;
			}
			return null ;
		} catch (Exception e) {
			return null ;
		} finally {
			if (scan != null) {
				scan.close();
			}
		}
	}
	public static boolean append(File file,String content) {
		PrintStream out = null ;
		try {
			out = new PrintStream(new FileOutputStream(file,true)) ;
			out.print(content); // 内容追加
			return true ;
		} catch (Exception e) {
			return false ;
		} finally {
			if (out != null) {
				out.close();
			}
		}
	}
```

```
}
```

2、扩充IStudentService操作方法：

```
package cn.mldn.demo.service;
import cn.mldn.demo.vo.Student;
public interface IStudentService {
	public void append(String str); //追加數據並且保存到文件
	public Student[] getData() ; // 获取排序数据
}
```

3、修改StudentServiceImpl中的功能：

```

```

4、此时工厂类不再需要输入数据：

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.IStudentService;
import cn.mldn.demo.service.impl.StudentServiceImpl;
public class Factory {
	private Factory() {}
	public static IStudentService getInstance() {
		return new StudentServiceImpl() ;
	}
}
```

5、定义一个菜单处理：

```
package cn.mldn.demo.menu;
import java.util.Arrays;
import cn.mldn.demo.factory.Factory;
import cn.mldn.demo.service.IStudentService;
import cn.mldn.demo.util.InputUtil;
public class Menu {
	public Menu() {
		this.choose();
	}
	public void choose() {
		this.show() ;
		String choose = InputUtil.getString("请进行选择：") ;
		switch(choose) {
		case "1" : {	// 接收输入数据
			String str = InputUtil.getString("请输入要追加的数据：") ;
			IStudentService studentService = Factory.getInstance() ;
			studentService.append(str); // 追加数据
			choose() ;	// 重复出现
		}
		case "2" : {	// 显示数据
			IStudentService studentService = Factory.getInstance() ;
			System.out.println(Arrays.toString(studentService.getData()));
			choose() ;	// 重复出现
		}
		case "0" : {
			System.out.println("下次再见，拜拜！");
			System.exit(1);
```

```
            }
            default : {
                System.out.println("您输入了非法的选项，无法进行处理，请确认后再次执行程
序！");
                choose() ;
            }
            }
        }
        public void show() {
            System.out.println("【1】追加字符串数据\n");
            System.out.println("【2】显示所有的学生数据\n");
            System.out.println("【0】结束程序执行。");
            System.out.println("\n\n\n");
        }
}
```

6、编写程序测试类：

```
package cn.mldn.demo;
import cn.mldn.demo.menu.Menu;
public class IOCaseDemo {
    public static void main(String[] args) {
        new Menu();
    }
}
```

```
Tom:89.1|Jerry:90.01|qiangqiang:59.9
```

■奇偶数统计

编写程序，当程序运行后，根据屏幕提示输入一个数字字符串，输入后统计有多少个偶数数字和奇数数字。

本质的流程就是进行每一个字符串的拆分，而后进行数字的转换处理。

1、定义INumberService接口，进行数字的处理服务：

```
package cn.mldn.demo.service;
public interface INumberService {
    public int[] stat() ;
}
```

2、定义NumberServiceImpl接口子类：

```
package cn.mldn.demo.service.impl;
import cn.mldn.demo.service.INumberService;
import cn.mldn.demo.util.InputUtil;
public class NumberServiceImpl implements INumberService {
    @Override
    public int[] stat() {
        int stat[] = new int [] {0,0} ;    // 第一个为偶数，第二个为奇数
        String str = InputUtil.getString("请输入数字信息：") ;
```

```
            String result [] = str.split("") ; // 按照每个字符拆分
            for (int x = 0 ; x < result.length ; x ++) {
                  if (Integer.parseInt(result[x]) % 2 == 0) {
                        stat [0] ++ ;
                  } else {
                        stat [1] ++ ;
                  }
            }
            return stat ;
      }
}
```

3、定义工厂类：

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.INumberService;
import cn.mldn.demo.service.impl.NumberServiceImpl;
public class Factory {
      private Factory() {}
      public static INumberService getInstance() {
            return new NumberServiceImpl() ;
      }
}
```

4、定义主类调用程序

```
package cn.mldn.demo;
import java.util.Arrays;
import cn.mldn.demo.factory.Factory;
public class IOCaseDemo {
      public static void main(String[] args) {
            System.out.println(Arrays.toString(Factory.getInstance().stat()));
      }
}
```

■用户登录

完成系统登录程序，从命令行输入用户名和密码，如果没有输入用户名和密码，则提示输入用户名和密码；如果输入了用户名但是没有输入密码，则提示用户输入密码，然后判断用户名是否是mldn，密码是否是hello，如果正确，则提示登录成功；如果错误，显示登录失败的信息，用户再次输入用户名和密码，连续3次输入错误后系统退出。

对于此时的程序发现可以将用户和密码同时输入，也可以先输入用户名，而后输入密码，如果超过了3次就表示登录结束，对于用户名和密码的使用可以采用“用户名/密码”的形式完成，如果发现没有“/”表示没有输入密码。

1、定义用户的操作接口：

```
package cn.mldn.demo.service;
public interface IUserService {
      public boolean isExit();
```

```
    public boolean login(String name, String password);
}
```

2、定义操作接口的子类：

```
package cn.mldn.demo.service.impl;
import cn.mldn.demo.service.IUserService;
public class UserServiceImpl implements IUserService {
    private int count = 0 ; // 作为登录统计
    @Override
    public boolean isExit() {
        return this.count >= 3; // 执行登录退出的条件
    }

    @Override
    public boolean login(String name, String password) {
        this.count ++ ;
        return "mldn".equals(name) && "hello".equals(password);
    }
}
```

3、对于登录失败的检测处理操作，应该单独定义一个用户的代理操作类：

```
package cn.mldn.demo.service.proxy;
import cn.mldn.demo.service.IUserService;
import cn.mldn.demo.util.InputUtil;
public class UserServiceProxy implements IUserService {
    private IUserService userService ;
    public UserServiceProxy(IUserService userService) {
        this.userService = userService ;
    }
    @Override
    public boolean login(String name, String password) {
        while(!this.isExit()) { // 不进行退出
            String inputData = InputUtil.getString("请输入登录信息：") ;
            if (inputData.contains("/")) {  // 输入了用户名和密码
                String temp [] = inputData.split("/") ;    // 数据拆分
                if (this.userService.login(temp[0], temp[1])) { // 登录认证
                    return true ;    // 循环结束了
                } else {
                    System.out.println("登录失败，错误的用户名或密码！");
                }
            } else {    // 现在只有用户名
                String pwd = InputUtil.getString("请输入密码：") ;
                if (this.userService.login(inputData, pwd)) {  // 登录认证
                    return true ;    // 循环结束了
                } else {
                    System.out.println("登录失败，错误的用户名或密码！");
                }
            }
        }
        return false;
    }
    @Override
```

```
    public boolean isExit() {
        return this.userService.isExit();
    }
}
```

4、修改工厂类定义：

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.IUserService;
import cn.mldn.demo.service.impl.UserServiceImpl ;
import cn.mldn.demo.service.proxy.UserServiceProxy;
public class Factory {
    private Factory() {}
    public static IUserService getInstance() {
        return new UserServiceImpl(new UserServiceImpl()) ;
    }
}
```

5、定义程序测试类：

```
package cn.mldn.demo;
import cn.mldn.demo.factory.Factory;
public class IOCaseDemo {
    public static void main(String[] args) {
        System.out.println(Factory.getInstance().login(null, null));
    }
}
```

真实业务只实现核心功能，辅助逻辑处理交给代理控制。

■投票选举

编写一个投票程序，具体如下。

```
（1）功能描述
有一个班采用民主投票方法推选班长，班长候选人共4位，每个人姓名及代号分别为“张三 1；李四 2；王五 3；赵六 4”。程序操作员将每张选票上所填的代号（1、2、3或4）循环输入电脑，输入数字0结束输入，然后将所有候选人的得票情况显示出来，并显示最终当选者的信息。
（2）具体要求
① 要求用面向对象方法，编写学生类Student，将候选人姓名、代号和票数保存到类Student中，并实现相应的getXXX 和 setXXX方法。
② 输入数据前，显示出各位候选人的代号及姓名（提示，建立一个候选人类型数组）。
③ 循环执行接收键盘输入的班长候选人代号，直到输入的数字为0，结束选票的输入工作。
④ 在接收每次输入的选票后要求验证该选票是否有效，即如果输入的数不是0、1、2、3、4这5个数字之一，或者输入的是一串字母，应显示出错误提示信息“此选票无效，请输入正确的候选人代号！”，并继续等待输入。
⑤ 输入结束后显示所有候选人的得票情况，如参考样例所示。
⑥ 输出最终当选者的相关信息，如参考样例所示。
（3）参考样例
1：张三【0票】
2：李四【0票】
3：王五【0票】
4：赵六【0票】
请输入班长候选人代号（数字0结束）：1
```

```
请输入班长候选人代号（数字0结束）：1
请输入班长候选人代号（数字0结束）：1
请输入班长候选人代号（数字0结束）：2
请输入班长候选人代号（数字0结束）：3
请输入班长候选人代号（数字0结束）：4
请输入班长候选人代号（数字0结束）：5
此选票无效，请输入正确的候选人代号!
请输入班长候选人代号（数字0结束）：hello
此选票无效，请输入正确的候选人代号!
请输入班长候选人代号（数字0结束）：0
1：张三【4票】
2：李四【1票】
3：王五【1票】
4：赵六【1票】
投票最终结果：张三同学，最后以4票当选班长!
```

1、建立学生类，这个类里面需要保存有编号、姓名、票数：

```
package cn.mldn.demo.vo;
public class Student implements Comparable<Student> {
	private long sid ;
	private String name ;
	private int vote ;
	public Student(long sid,String name,int vote) {
		this.sid = sid ;
		this.name = name ;
		this.vote = vote ;
	}
	@Override
	public String toString() {
		return "【"+this.sid+"】姓名：" + this.name + "、票数：" + this.vote ;
	}
	@Override
	public int compareTo(Student stu) {
		return stu.vote - this.vote ;
	}
}
```

2、定义投票处理的业务接口

```
package cn.mldn.demo.service;
import cn.mldn.demo.vo.Student;
public interface IVoteService {
	public boolean inc(long sid) ; // 根据编号进行增长
	public Student[] result() ; // 获取投票的结果
	public Student[] getData() ; // 获取全部数据
}
```

3、定义VoteServiceImpl子类：

```
package cn.mldn.demo.service;
import java.util.Arrays;
import cn.mldn.demo.vo.Student;
public class VoteServiceImpl implements IVoteService {
	private Student [] students = new Student[] {
```

```
                new Student(1,"张三",0) , new Student(2,"李四",0) ,
                new Student(3,"王五",0) , new Student(4,"赵六",0)} ;
    @Override
    public boolean inc(long sid) {
        for (int x = 0 ; x < students.length ; x ++) {
            if (this.students[x].getSid() == sid) {       // 获取了指定的编号
                this.students[x].setVote(this.students[x].getVote() + 1); // 票数增长
                return true ;
            }
        }
        return false ;
    }
    @Override
    public Student[] getData() {
        return this.students ;
    }
    @Override
    public Student[] result() {
        Arrays.sort(this.students);
        return this.students ;
    }
}
```

4、定义工厂类

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.IVoteService;
import cn.mldn.demo.service.VoteServiceImpl;
public class Factory {
    private Factory() {}
    public static IVoteService getInstance() {
        return new VoteServiceImpl() ;
    }
}
```

5、定义一个菜单的信息显示类

```

```

6、定义程序测试类

```
package cn.mldn.demo.factory;
import cn.mldn.demo.service.IVoteService;
import cn.mldn.demo.service.VoteServiceImpl;
public class Factory {
    private Factory() {}
    public static IVoteService getInstance() {
        return new VoteServiceImpl() ;
    }
}
```