面向对象与多线程综合实验

结合华为云平台的基于JAVASE数据挖掘系统设计与实现

主要内容

课程简介
- 考核办法
- 教学目标
- 教学安排

系统分析
- 系统简介
- 需求分析
- 功能要求

本次实验内容
- 实验目标和要求
- 实验内容（包括要完成的系统功能、模块设计和开发难点）
- 知识点介绍

一、课程简介

考核办法

- 百分制
- 平时成绩40分
- 验收35分
- 实验报告25分
  - 规范性：字体大小一致，排版美观，段落规整，无错别字，图和表有题注，双面打印
  - 重点记录和说明实验过程中遇到问题和解决办法
  - 实现的功能模块给出必要的截图和说明

*切记粘贴大段代码；如有雷同或者相似的心得体会，直接列为抄袭！

教学目标

- 深入理解面向对象语言的基本概念和原理，掌握JavaSE中关于输入输出、JDBC、GUI、网络编程、多线程的相关技术和应用
- 熟悉软件开发过程，了解企业级业界先进的开发平台、工具和流程，培养独立分析和解决问题的软件工程实践能力
- 提高在线学习和自主化、碎片化学习的能力

教学安排

- 经过7次迭代开发一个数据挖掘系统；
- 第8次课系统演示，并交代码和实验报告，其中实验报告纸质版作为最终考试试卷上交教务处。

教学平台

- 面向对象与多线程综合实验开发工具下载链接：

登陆实验中心网址：59.69.101.195　　文件下载》》常用下载

教学平台

- **课程教学平台：**

登陆实验中心网址：59.69.101.195 虚拟实验室》》智学云教育

用户名为学号

教学平台

- **课程教学平台：**

登陆实验中心网址：59.69.101.195 虚拟实验室》》智学云教育

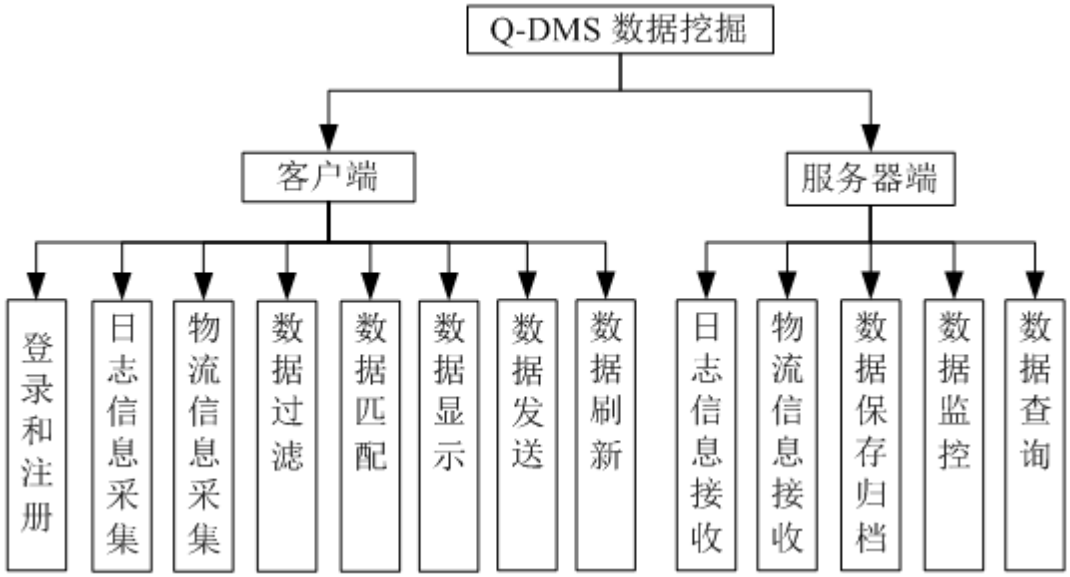
用户名为学号，进入到我的课程进行学习

二、系统分析

系统简介

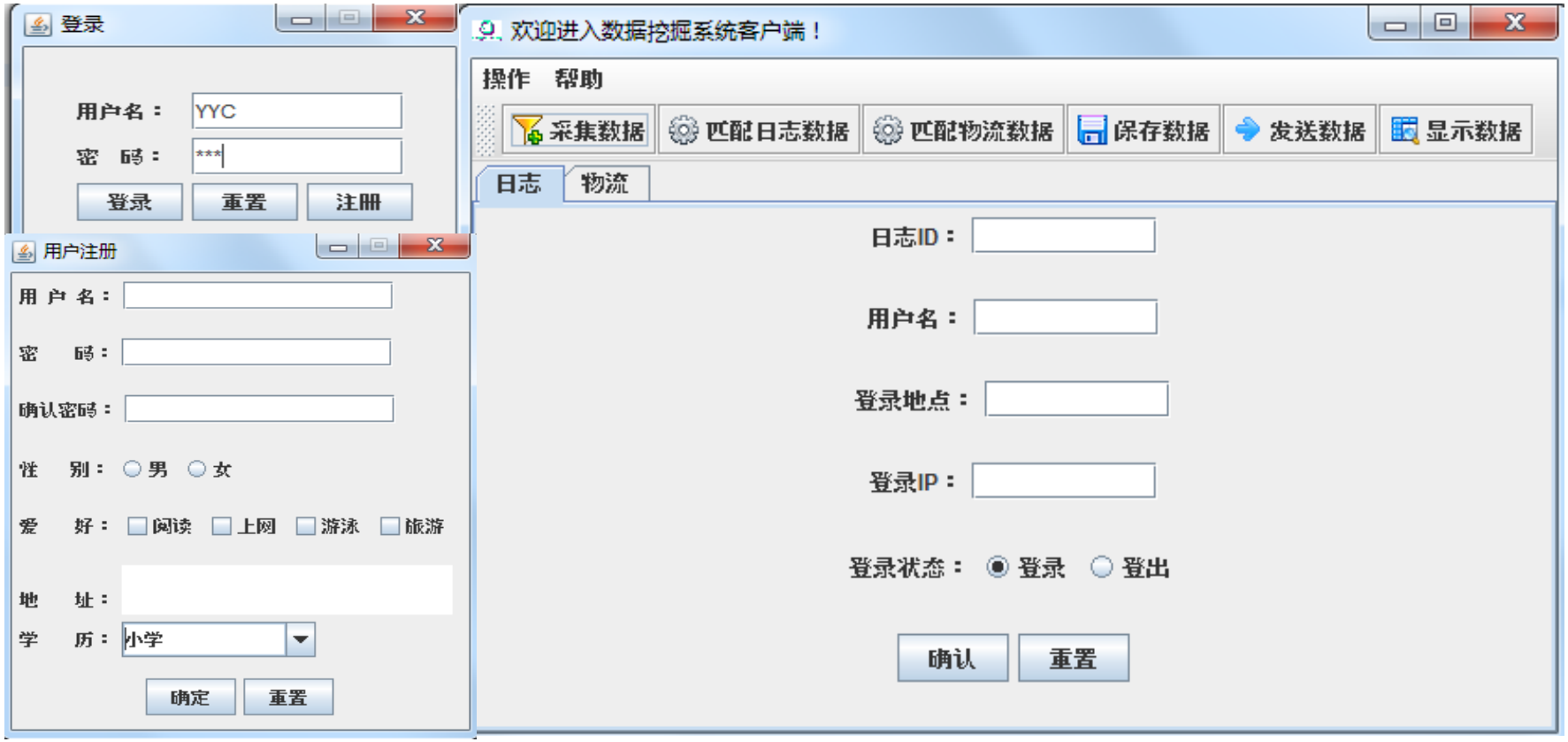
- 基于Java SE 的数据挖掘系统

基于客户端服务器端（Client-Server，C-S）模式，实现日志与物流数据信息的采集、匹配、保存、显示等功能，为数据分析挖掘提供基础支撑。



系统简介

- 界面显示


登录
用户名：
YYC
密　码：
***
登录
重置
注册
用户注册
用 户 名：
密　码：
确认密码：
性　别：
男
女
爱　好：
阅读
上网
游泳
旅游
地　址：
学　历：
小学
确定
重置
欢迎进入数据挖掘系统客户端！
操作
帮助
采集数据
匹配日志数据
匹配物流数据
保存数据
发送数据
显示数据
日志
物流
日志ID：
用户名：
登录地点：
登录IP：
登录状态：
登录
登出
确认
重置

需求分析

- 系统包括客户端应用程序、服务器端应用程序---C/S模式
- 用户和数据信息的保存---JDBC数据库保存和查询
- 用户能够进行注册和登录，授权后使用系统---GUI登录和注册界面设计与功能实现
- 能够实现日志和物流信息的数据采集（录入），登录登出对匹配、信息保存和数据显示等功能---GUI主界面设计与功能实现
- 系统能够进行数据自动刷新功能，与数据库保持同步---线程
- 客户端与服务器端交互，客户端能够将数据发送到服务器端，服务器端接收客户端发送的日志和物流信息，进行保存和处理---Socket通信
- 系统优化---JAVA高级应用与新特性

任务分配

课时分配	实验任务	任务列表	主要知识点分解
第1-2次课	基于控制台的系统数据**采集、匹配、显示**和**记录**功能实现	✓ **注册华为软开云平台账户** ✓ 搭建数据挖掘系统**框架** ✓ 实现日志和物流数据信息的**采集**、**匹配**和**显示**功能 ✓ 实现匹配的物流数据信息的文件保存和读取**记录**功能	继承与多态/异常处理/集合/文件存储及IO流/华为软开云
第3次课	基于**JDBC**的控制台系统基本功能实现	✓ 创建项目所需的数据库表，并搭建数据访问基础环境 ✓ 实现并测试匹配的日志、物流信息的数据库保存和查询功能	JAVAJDBC/MySQL
第4次课	基于SwingGUI的系统**注册**和**登录界面**设计实现	✓ 创建用户数据库表、用户实体类和用户业务逻辑类 ✓ 创建用户注册窗口，并将用户注册信息保存到数据库 ✓ 创建用户登录窗口，登录成功则进入系统主界面	SwingGUI/事件驱动/WinBuilder
第5次课	基于SwingGUI系统**主界面**设计实现和系统**优化**	✓ 实现主界面中的菜单和工具栏 ✓ 实现主界面中的日志和物流**数据采集**、**匹配**、**保存**和**显示**功能 ✓ GUI系统优化	高级UI组件
第6次课	系统信息**自动刷新**功能实现	✓ 使用线程实现每隔**1分钟(或自定义)**日志和物流显示数据表格**自动刷新**功能，以便与数据库保持同步	线程
第7次课	系统客户端/服务器端数据**发送(交互)**功能实现	✓ 客户端应用程序：修改主界面**发送数据**页面响应，即使用Socket实现数据由客户端发送到服务器 ✓ 服务器端应用程序：使用Server Socket实现接收客户端发送的日志和物流数据信息，并将信息保存到数据库	Socket网络编程
第8次课	系统验收	✓ 数据挖掘系统的基本功能实现与演示 ✓ **在华为软开云平台反推系统需求分析和设计**	PPT演示、系统运行
拓展	Java 高级应用以及Java8新特性	◆ 使用注解重新迭代升级系统代码 ◆ 使用格式化将输出的日期进行格式化输出 ◆ 使用Lambda表达式迭代升级主窗口中“帮助”菜单的事件处理 ◆ 使用Lambda表达式实现查找指定的匹配信息并显示	增加注解和格式化以及Lambda优化和查询

功能要求

系统基于C/S模式，包括客户端和服务器端应用程序

- 1. 用户登录和注册功能:用户验证口令通过后登录系统，新用户进行注册，并将注册信息保存数据库
- 2. 日志、物流数据的采集功能：对日志和物流数据进行采集，并保存到Mysql数据库；
- 3. 日志、物流数据的筛选匹配功能，以日志信息为例：
  - 根据日志的登录、登出状态，对日志进行分类，分别存放到**登录**日志集合（ArrayList<LogRec> logIns）和**登出**日志集合（ArrayList<LogRec> logOuts）中。
  - 在登录日志和登出日志中，根据**用户名**和IP地址进行匹配：如果存在相同的用户名和IP地址，则日志信息匹配成功，将匹配的日志数据封装到MatchedLogRec对象，并保存到匹配日志集合（ArrayList<MatchedLogRec> matchLogs）中。
- 4. 日志、物流数据的数据保存功能：在系统主界面中点击“保存数据”按钮时，将匹配的日志数据和物流数据保存到本地文件和数据库中。
- 5. 客户端服务器端交互功能：
  - ✓ 客户端的数据发送功能：在客户端通过Socket技术向服务器端发送匹配的日志数据和物流数据；当数据发送成功后，清空客户端暂时存放数据的集合，然后弹出信息提醒；
  - ✓ 服务器数据查询功能：当点击客户端显示数据功能时，服务器端从数据库中查找符合条件的数据，并发送到客户端；在客户端以表格的形式将数据显示。

三、本次实验内容

实验目标

1. 理解与掌握知识点：
   - 类与对象：类定义、对象创建、方法调用、重载与覆盖、继承与多态、抽象类、接口的使用
   - 异常处理：异常种类，异常机制
2. 熟悉工具与平台：华为软开云平台，及账户注册

网址：https://www.huaweicloud.com

3. 学会开发控制台程序：
   ① 掌握如何进行程序的调试（单步执行，监控变量等）
   ② 理解给定的程序系统框架
   ③ 实现系统中的数据采集、数据匹配和显示功能

实验内容与交付

1. 设计控制台主界面菜单

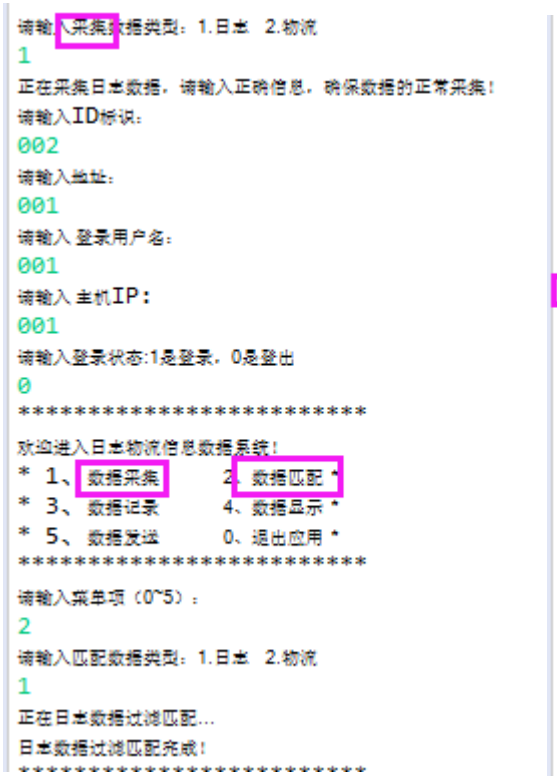
2. 熟悉系统框架，借鉴物流数据的功能实现，完成日志数据的采集、匹配和显示功能

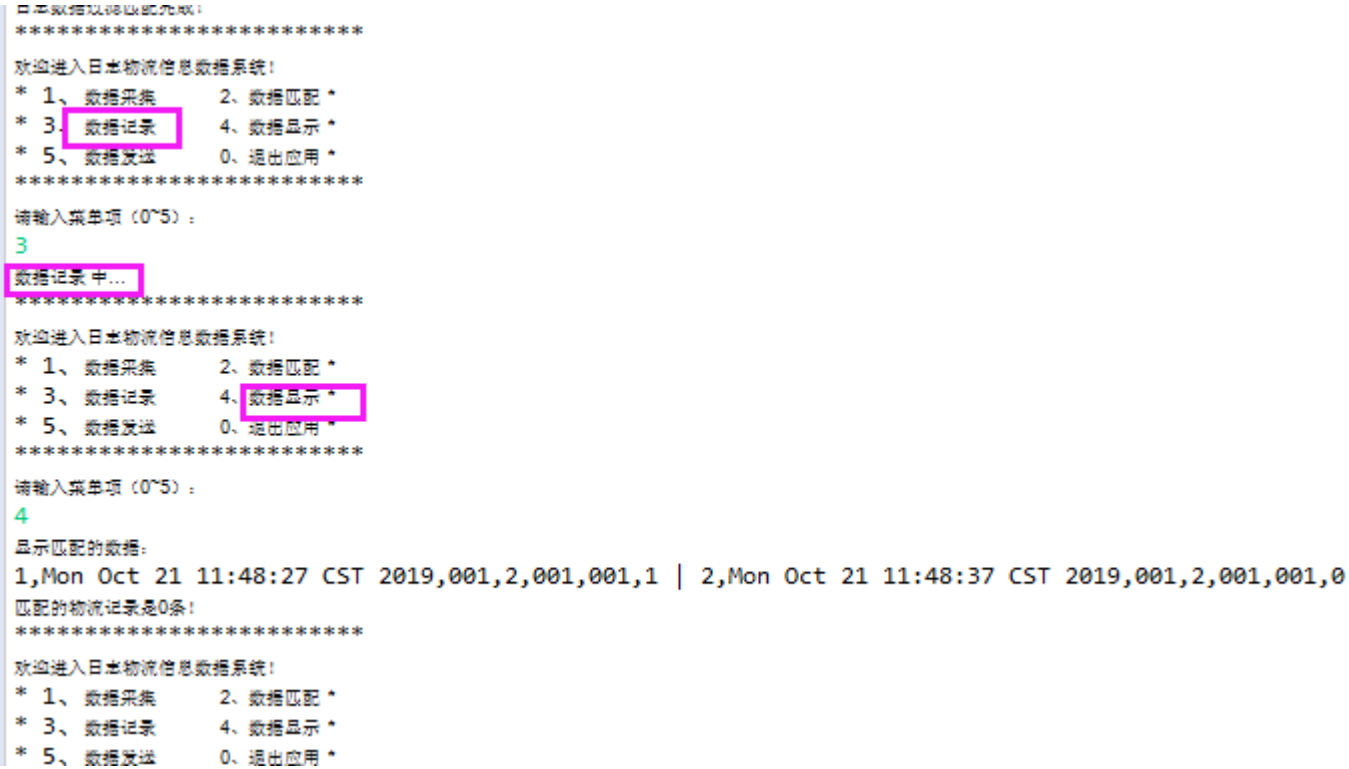
★要求：循环输出菜单，并能够有较好的异常处理跟踪机制，友好的错误提示信息

★交付：可运行源代码

实验内容-控制台主界面菜单

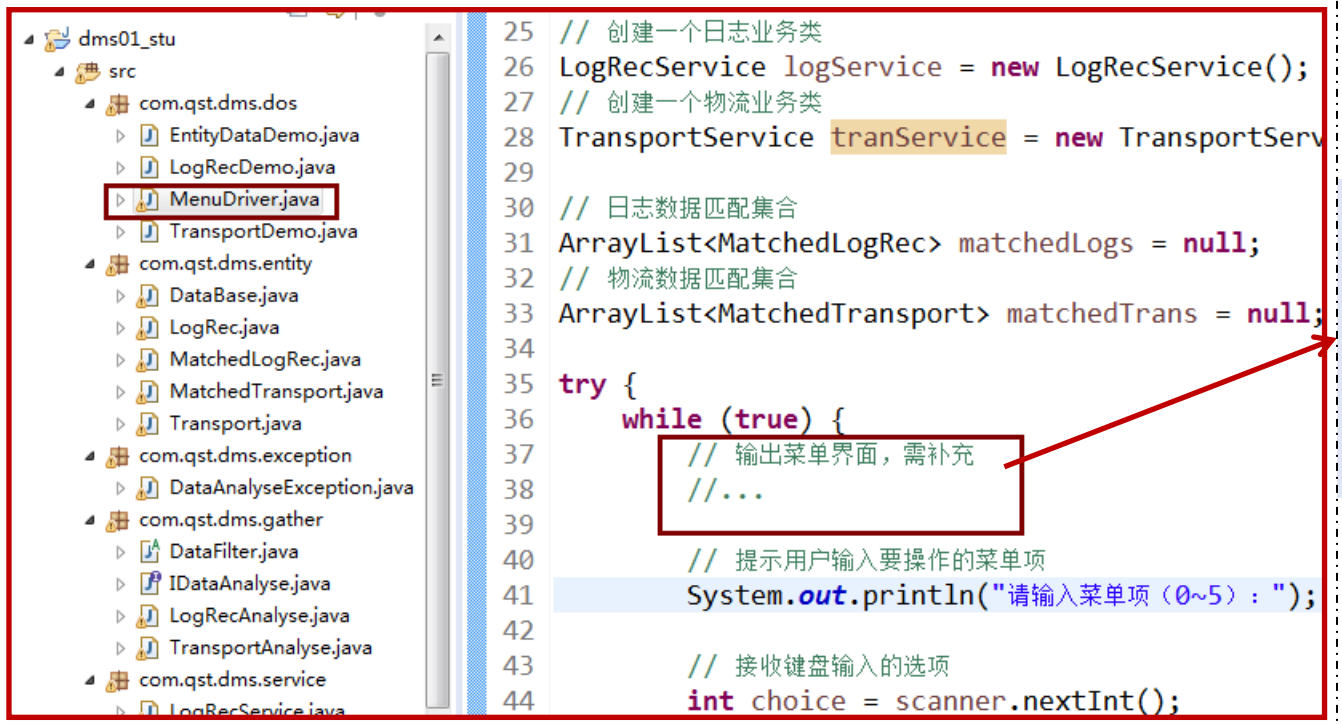
日志和物流数据完成的业务功能菜单，日志数据的采集、匹配和显示





实验内容-控制台主界面菜单

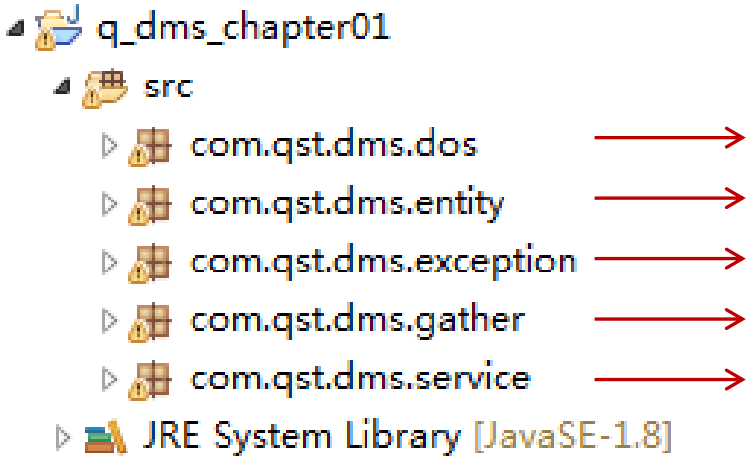
1. 按右图形式补全输出菜单界面




```
**************************
欢迎进入日志物流信息数据系统!
* 1、数据采集          2、数据匹配 *
* 3、数据记录          4、数据显示 *
* 5、数据发送          0、退出应用 *
**************************
请输入菜单项（0~5）：
```

实验内容-系统框架介绍

2. 熟悉系统框架

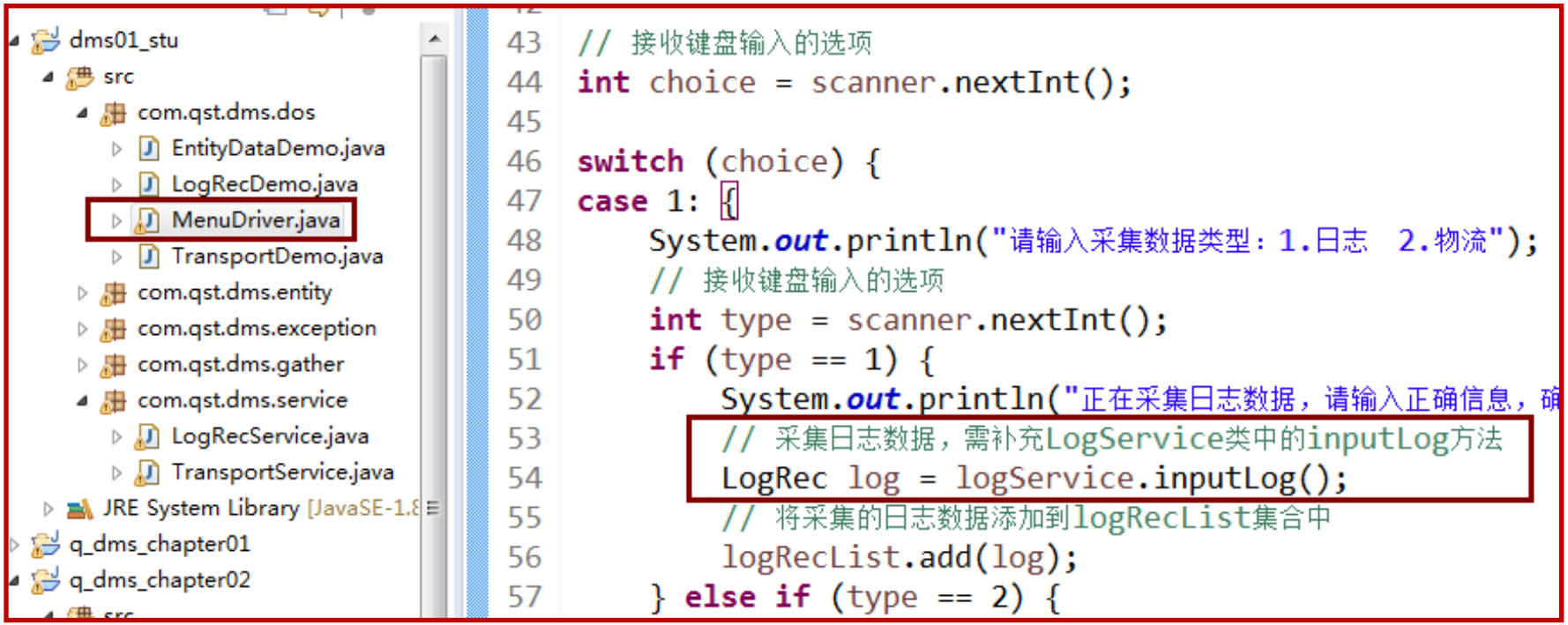


- com.qst.dms.dos → 控制台主程序
- com.qst.dms.entity → 实体类，日志和物流数据以及匹配后的数据类
- com.qst.dms.exception → 自定义的异常处理类
- com.qst.dms.gather → 集合类，匹配筛选的数据集合
- com.qst.dms.service → 业务类，封装关于日志和物流数据的操作如采集、输出、保存等

实验内容-日志数据采集

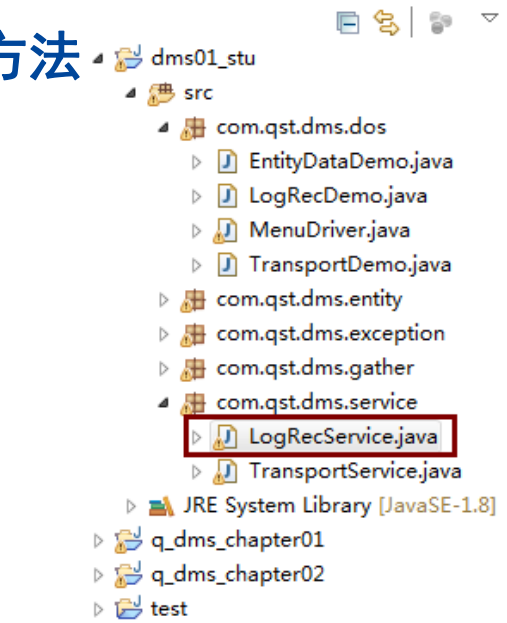
3. 日志数据采集步骤

- ✓ 将日志数据采集操作封装到inputLog()方法中
- ✓ 将采集后的数据添加到日志集合



实验内容-日志数据采集

inputLog()方法




```
请输入采集数据类型：1.日志  2.物流
1
正在采集日志数据，请输入正确信息，确保数据的正常采集！
请输入ID标识：
001
请输入地址：
wuhan
请输入 登录用户名：
yyc
请输入 主机IP:
100
请输入登录状态:1是登录，0是登出
1
```

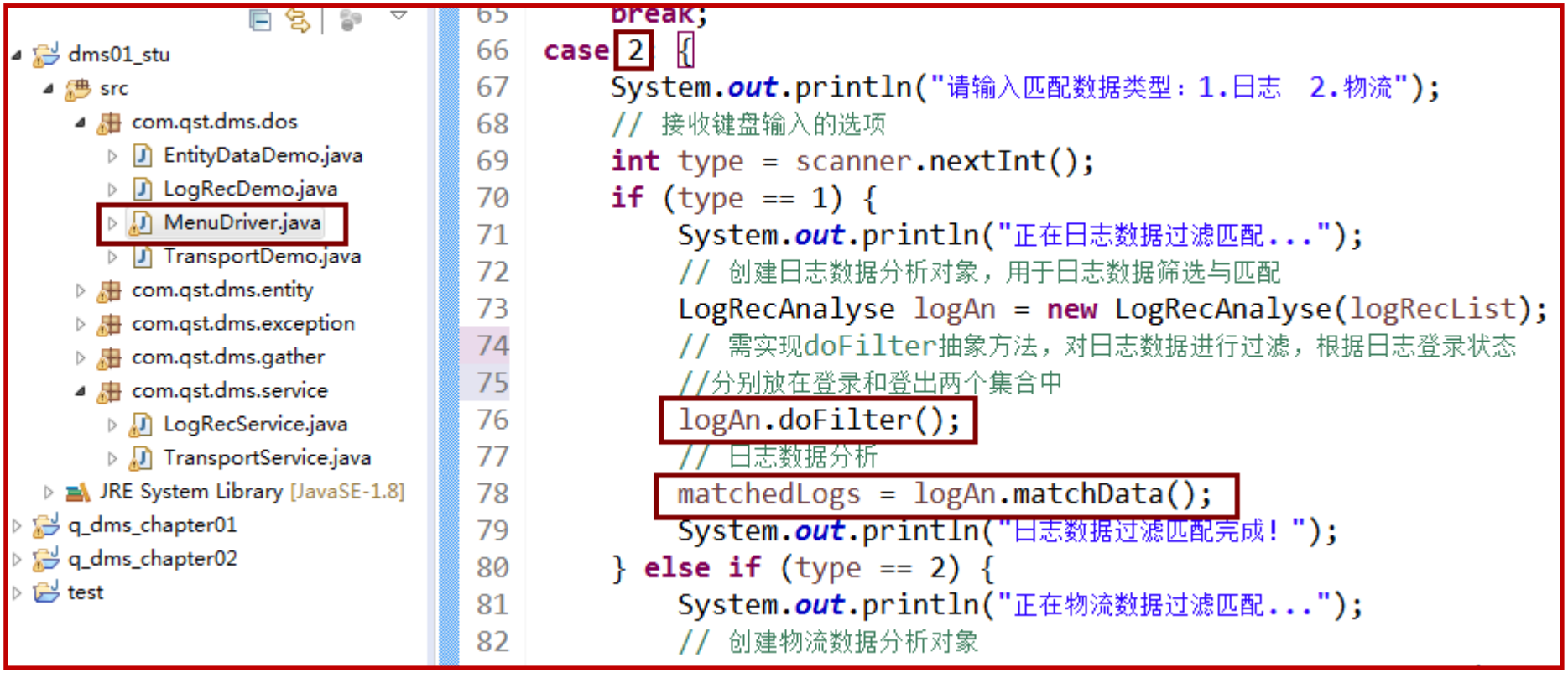
```
15  // 日志数据采集
16  public LogRec inputLog() {
17      LogRec log = null;
18      // 建立一个从键盘接收数据的扫描器
19      Scanner scanner = new Scanner(System.in);
20      try {
21          // 提示用户输入ID标识
22          System.out.println("请输入ID标识：");
23          // 接收键盘输入的整数
24
25          // 获取当前系统时间
26
27          // 提示用户输入地址
28
29          // 接收键盘输入的字符串信息
30
31          // 数据状态是“采集”
32          int type = DataBase.GATHER;
33          // 提示用户输入登录用户名
34
35          // 接收键盘输入的字符串信息
36
37          // 提示用户输入主机IP
38
39          // 接收键盘输入的字符串信息
40
41          // 提示用户输入登录状态、登出状态
42          System.out.println("请输入登录状态:1是登录，0是登出");
43          int logType = scanner.nextInt();
```

完成注释的语句

实验内容-日志数据采集

4. 日志数据匹配Match

- ✓ 创建日志分析对象LogRecAnalyse，用于数据过滤与匹配
- ✓ 对采集后的日志集合按照登录状态进行过滤，过滤出登录集合和登出2个集合
- ✓ 按用户名和IP规则匹配登录登出集合，得到登录登出对，视为数据匹配完成



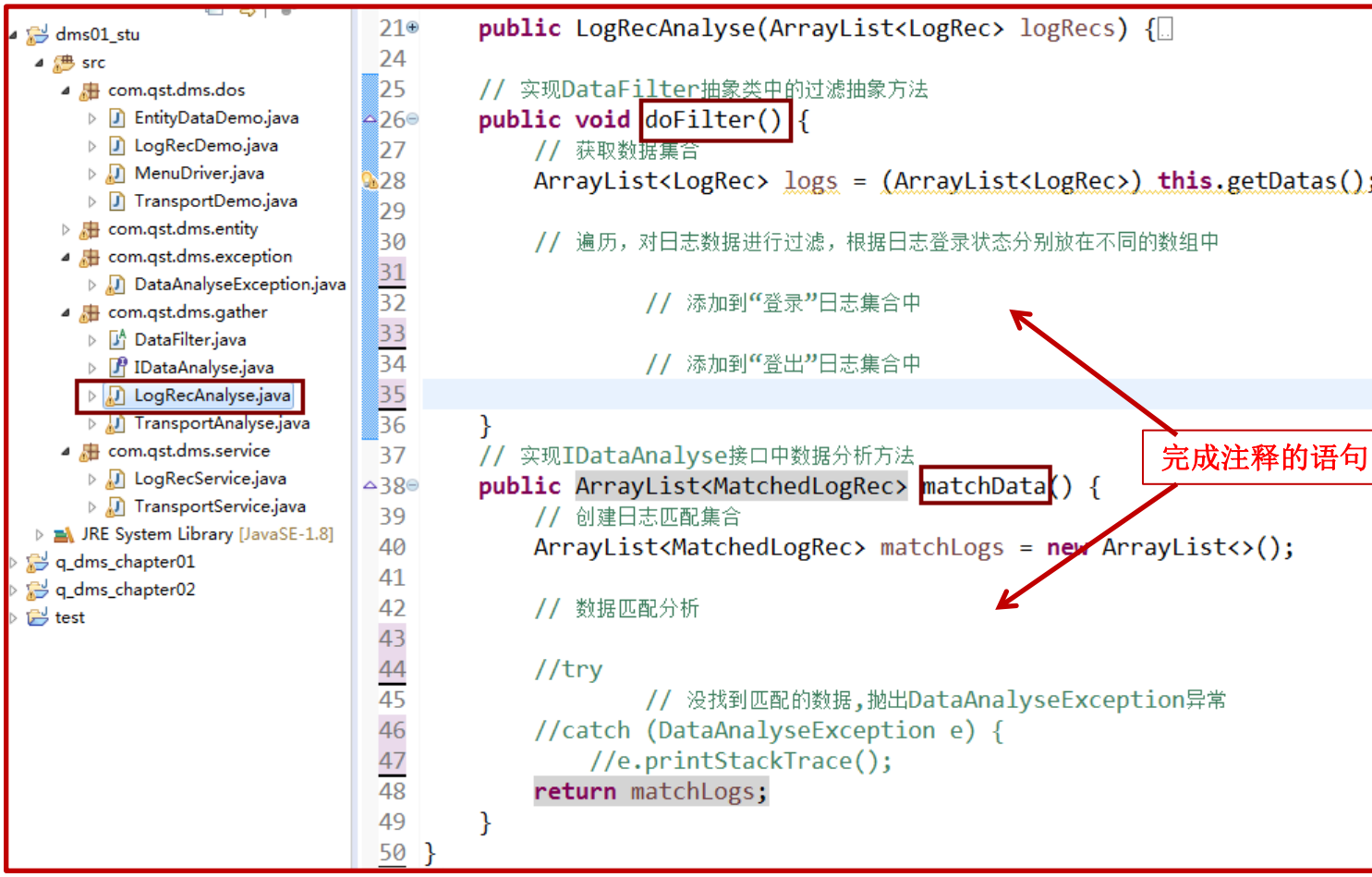
实验内容-日志数据采集

doFilter()方法

按登录状态过滤出登录和登出两个集合

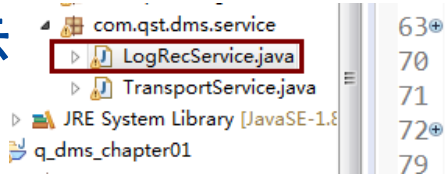
matchData()方法

在登录登出集合中按用户名和IP一致的条件进行匹配，符合条件的一对登录登出数据即为匹配，并添加到匹配集合matchLogs



实验内容-控制台主界面菜单

5. 日志数据显示




```
63    public void showMatchLog(MatchedLogRec... matchLogs) {
70
71    // 匹配日志信息输出,参数是集合
72    public void showMatchLog(ArrayList<MatchedLogRec> matchLogs) {
79
```

```
case 4: {
    System.out.println("显示匹配的数据：");
    if (matchedLogs == null || matchedLogs.size() == 0) {
        System.out.println("匹配的日志记录是0条！");
    } else {
        //输出匹配的日志信息
        logService.showMatchLog(matchedLogs);
    }
    if (matchedTrans == null || matchedTrans.size() == 0) {
        System.out.println("匹配的物流记录是0条！");
    } else {
        // 输出匹配的物流信息
        tranService.showMatchTransport(matchedTrans);
    }
}
```

完善showMatchLog()方法

```
**************************
欢迎进入日志物流信息数据系统！
*  1、数据采集          2、数据匹配  *
*  3、数据记录          4、数据显示  *
*  5、数据发送          0、退出应用  *
**************************
请输入菜单项（0~5）：
4
显示匹配的数据：
1,Wed Nov 04 12:57:31 CST 2020,wuhan,2,yyc,100,1 | 2,Wed Nov 04 12:57:40 CST 2020,wuhan,2,yyc,100,0
匹配的物流记录是0条！
```

实验内容--一些问题

- 系统框架某个包某个类的作用
- 物流数据的匹配是根据什么条件？
- Eclipse下如何进行断点调试？

数据实体entity类

基础数据父类DataBase

ID标识	时间戳	地点	数据状态
Id	Time	address	type

采集
匹配
记录
发送
接收
归档
保存

LogRec 日志登录记录子类

用户名	主机IP	登录状态
user	ip	logType 登入1登出0

Transport物流登录记录子类

经手人	收货人	物流状态
handler	receiver	transportType 发货中1 送货中2 已签收3

数据实体entity类

MatchedLogRec匹配日志登入登出对记录

根据IP地址和用户名进行匹配

```java
//匹配日志记录."登录登出对"类型
public class MatchedLogRec implements Serializable{
    private LogRec login;
    private LogRec logout;

    // user用户登录名
    public String getUser() {...}

    // 登入时刻
    public Date getLogInTime() {...}

    // 登出时刻
    public Date getLogoutTime() {...}

    // 登入记录
    public LogRec getLogin() {
        return login;
    }

    // 登出记录
    public LogRec getLogout() {
        return logout;
    }

    public MatchedLogRec() {...}

    public MatchedLogRec(LogRec login, LogRec logout) {
        if (login.getLogType() != LogRec.LOG_IN) {
            throw new RuntimeException("不是登录记录!");
        }
        if (logout.getLogType() != LogRec.LOG_OUT) {
            throw new RuntimeException("不是登出记录");
        }
        if (!login.getUser().equals(logout.getUser())) {
            throw new RuntimeException("登录登出必须是同一个用户!");
        }
        if (!login.getIp().equals(logout.getIp())) {
            throw new RuntimeException("登录登出必须是同一个IP地址!");
        }
```

数据实体entity类

MatchedTransport匹配物流登入登出对记录

根据什么进行匹配？？

```java
package com.cst.dms.entity;

import java.io.Serializable;

public class MatchedTransport implements Serializable {

    private Transport send;
    private Transport trans;
    private Transport receive;

    public Transport getSend() {

    public void setSend(Transport send) {

    public Transport getTrans() {

    public void setTrans(Transport trans) {

    public Transport getReceive() {

    public void setReceive(Transport receive) {

    public MatchedTransport() {

    public MatchedTransport(Transport send, Transport trans, Transport receive) {
        if (send.getTransportType() != Transport.SENDDING) {
            throw new RuntimeException("不是发货记录!");
        }
        if (trans.getTransportType() != Transport.TRANSPORTING) {
            throw new RuntimeException("不是运货记录!");
        }
        if (receive.getTransportType() != Transport.RECIEVED) {
            throw new RuntimeException("不是签收记录!");
        }
        this.send = send;
        this.trans = trans;
        this.receive = receive;
    }

    public String toString() {
        // TODO Auto-generated method stub
```

集合类

- 采用泛型数据集合存储匹配后的日志和物流数据

```java
import java.util.ArrayList;

//数据过滤抽象类
public abstract class DataFilter {
    // 数据集合,使用泛型集合
    private ArrayList<? extends DataBase> datas;

    public ArrayList<? extends DataBase> getDatas() {
        return datas;
    }

    public void setDatas(ArrayList<? extends DataBase> datas) {
        this.datas = datas;
    }
```

```java
package com.cst.dms.gather;

import java.util.ArrayList;

//数据分析接口
public interface IDataAnalyse {
// 进行数据匹配,返回泛型ArrayList集合
        ArrayList<?> matchData();

}
```

集合类

- 采用泛型数据集合存储匹配后的物流数据

```
//物流分析类，继承DataFilter抽象类，实现数据分析接口
public class TransportAnalyse extends DataFilter implements IDataAnalys
    // 发货集合
    private ArrayList<Transport> transSends = new ArrayList<>();
    // 送货集合
    private ArrayList<Transport> transIngs = new ArrayList<>();
    // 已签收集合
    private ArrayList<Transport> transRecs = new ArrayList<>();

    // 构造方法
    public TransportAnalyse() {
    }

    public TransportAnalyse(ArrayList<Transport> trans) {
        super(trans);
    }

    // 实现DataFilter抽象类中的过滤抽象方法
    public void doFilter() {
        // 获取数据集合
        ArrayList<Transport> trans = (ArrayList<Transport>) this.getDat

        // 遍历，对物流数据进行过滤，根据物流状态分别放在不同的集合中
        for (Transport tran : trans) {
            if (tran.getTransportType() == Transport.SENDDING) {
                transSends.add(tran);
            } else if (tran.getTransportType() == Transport.TRANSPORTIN
                transIngs.add(tran);
            } else if (tran.getTransportType() == Transport.RECIEVED) {
                transRecs.add(tran);
            }
        }

    }
```

```
    // 实现IDataAnalyse接口中数据分析方法
    public ArrayList<MatchedTransport> matchData() {
        // 创建物流匹配集合
        ArrayList<MatchedTransport> matchTrans = new ArrayList<>();
        // 数据匹配分析
        for (Transport send : transSends) {
            for (Transport tran : transIngs) {
                for (Transport rec : transRecs) {
                    if ((send.getReciver().equals(tran.getReciver()))
                            && (send.getReciver().equals(rec.getReciver()))) {
                        // 修改物流状态类型为“匹配”
                        send.setType(DataBase.MATHCH);
                        tran.setType(DataBase.MATHCH);
                        rec.setType(DataBase.MATHCH);
                        // 添加到匹配集合中
                        matchTrans.add(new MatchedTransport(send, tran, rec));
                    }
                }
            }
        }
        try {
            if (matchTrans.size() == 0) {
                // 没找到匹配的数据，抛出DataAnalyseException异常
                throw new DataAnalyseException("没有匹配的物流数据！");
            }
        } catch (DataAnalyseException e) {
            e.printStackTrace();
        }
        return matchTrans;
    }
}
```

三、知识点介绍

类与对象

1. 对象Object

现实世界：客观存在实体

计算机：映射存储区域

2. 类Class

描述:静态属性

动态方法

类是对象的抽象/模板

对象是类的实例/具体

封装(Encapsulation)

- 即将类的状态信息隐藏在类内部，不允许外部程序直接访问，必须通过该类提供的方法来实现对隐藏信息的操作和访问。

- 修改属性的可见性来限制对属性的访问；为每个属性创建一对赋值(setter)方法和取值(getter)方法，用于对这些属性的存取。

继承(Inheritance)

- 是使用已存在的类的定义作为基础建立新类的技术，新类可以吸收现有类的成员，并可以增加新的功能或修改原有的功能。
  - is-a的关系
  - （is a 代表的是类之间的继承关系，比如PC机是计算机，工作站也是计算机。PC机和工作站是两种不同类型的计算机，但都继承了计算机的共同特性。因此在用 Java语言实现时，应该将PC机和工作站定义成两种类，均继承计算机类。）
  - 注意：区分父类的那些属性和方法，子类可以继承，那些子类不可以继承

多态(Polymorphism)

- 是指计算机程序运行时，系统可依据对象所属类，引发对应类的方法，从而有不同的行为
- 设计时多态和运行时多态

（多态的概念 ：父类引用指向子类对象 ，而实际调用的方法为子类的方法；编译时多态即为重载 ；运行时多态有继承、重写、父类引用指向子类对象。）

- 实现多态
  - 继承的存在
  - 子类重写父类的方法
  - 父类引用变量指向子类对象

面向对象语言的程序设计方法

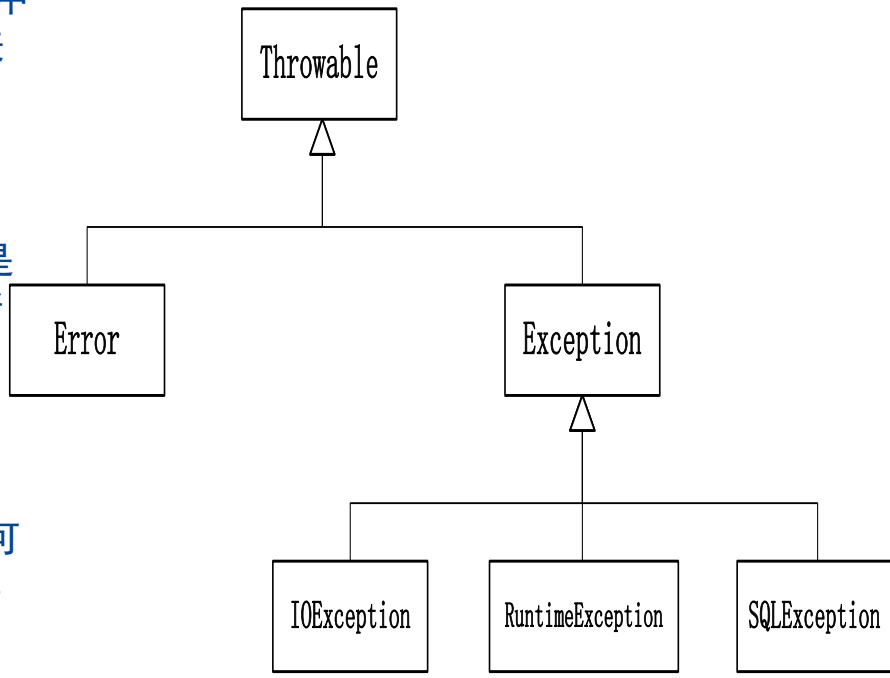
- 有哪些对象类
- 每个类有哪些属性和方法
- 类与类之间的关系是什么
- 不同对象之间如何进行通信

异常

- **异常：指不期而至的各种状况**
  - 异常的发生干扰了正常的指令流程
  - Java通过API中Throwable类的众多子类描述各种不同的异常
  - Java异常都是对象，是Throwable子类的实例，描述了出现在一段编码中的错误条件，当条件发生时，错误将引发异常

异常

Error错误：是程序无法处理的错误，表示运行应用程序中较严重问题。大多数错误与代码编写者执行的操作无关，而表示代码运行时 JVM(Java 虚拟机)出现的问题。



Exception异常： 是程序本身可以处理的异常。该类是JAVA异常类体系父类，根据是否是程序自身导致的异常，将所有的异常类分为：

- RuntimeException及其子类：

  属于程序运行时异常，即由程序自身的问题导致的异常。属于不可查异常，编译器不要求强制处置的异常,程序中可以选择捕获处理，也可以不处理。即不处理这样的异常也不会出现语法错误。

- 其他Exception子类

  属于由程序外部问题引起的异常，属于可查异常。除了RuntimeException及其子类以外，其他的Exception类及其子类都属于可查异常。这种异常的特点是Java编译器会检查它，在语法上强制程序员必须处理，否则出现语法错误，编译不通过。

异常处理机制

- 当程序运行过程中发生异常情况时，JAVA运行环境会自动生成相应异常类的对象保存相应异常信息。异常对象来通知程序，程序再根据异常对象的类型进行相应的处理。

- 为了使程序出现异常时也能正常运行，需要对异常进行相关的处理操作，这种操作称之为异常处理。

- JAVA异常处理机制：

1. 使用Try…catch捕获异常
2. 使用throws声明抛出异常

- JAVA异常处理机制优点：思考

异常处理机制

A. 抛出throw异常

当一个方法出现错误引发异常时，方法创建异常对象并交付运行时系统，异常对象中包含了异常类型和异常出现时的程序状态等异常信息。运行时系统负责寻找处置异常的代码并执行。

在方法声明处抛出异常

```
methodname throws
Exception1,Exception2,..,ExceptionN {…… }
```

在方法体中抛出异常

```
throw new exceptionname;
```

注意：程序会在throw语句后立即终止。

B. 捕捉catch异常

当一个方法出现在方法抛出异常之后，运行时系统将寻找合适的异常处理器（exception handler）。运行时系统从发生异常的方法开始，依次回查调用栈中的方法，直至找到含有合适异常处理器的方法并执行。当运行时系统遍历调用栈而未找到合适的异常处理器，则运行时系统终止。同时，意味着Java程序的终止。

异常处理机制

B. 捕捉catch异常

```
try {
   // 可能会发生异常的程序代码
} catch (Type1 id1) {
   // 捕获并处理try抛出的异常类型Type1
} catch (Type2 id2) {
   // 捕获并处理try抛出的异常类型Type2
} finally {
   // 无论是否发生异常，都将执行的语句块
}
```

无论是否捕获或处理异常，finally块里的语句都会被执行。当在try块或catch块中遇到return语句时，finally语句块将在方法返回之前被执行。

泛型与集合

JDK 5.0后引入泛型，即将数据类型参数化，而无需强制类型转换

“<>”

◆ 【实例化泛型类语法】

类名<类型参数列表>对象=new 类名<类型参数列表>([构造方法参数列表])

例如：Test<String> mytest=new Test<String>();

◆ 【?号通配符表示可以匹配任意类型，任意的Java类都可以匹配】

<? super Type>

<? extends Type>

例如：private ArrayList<? extends DataBase> datas

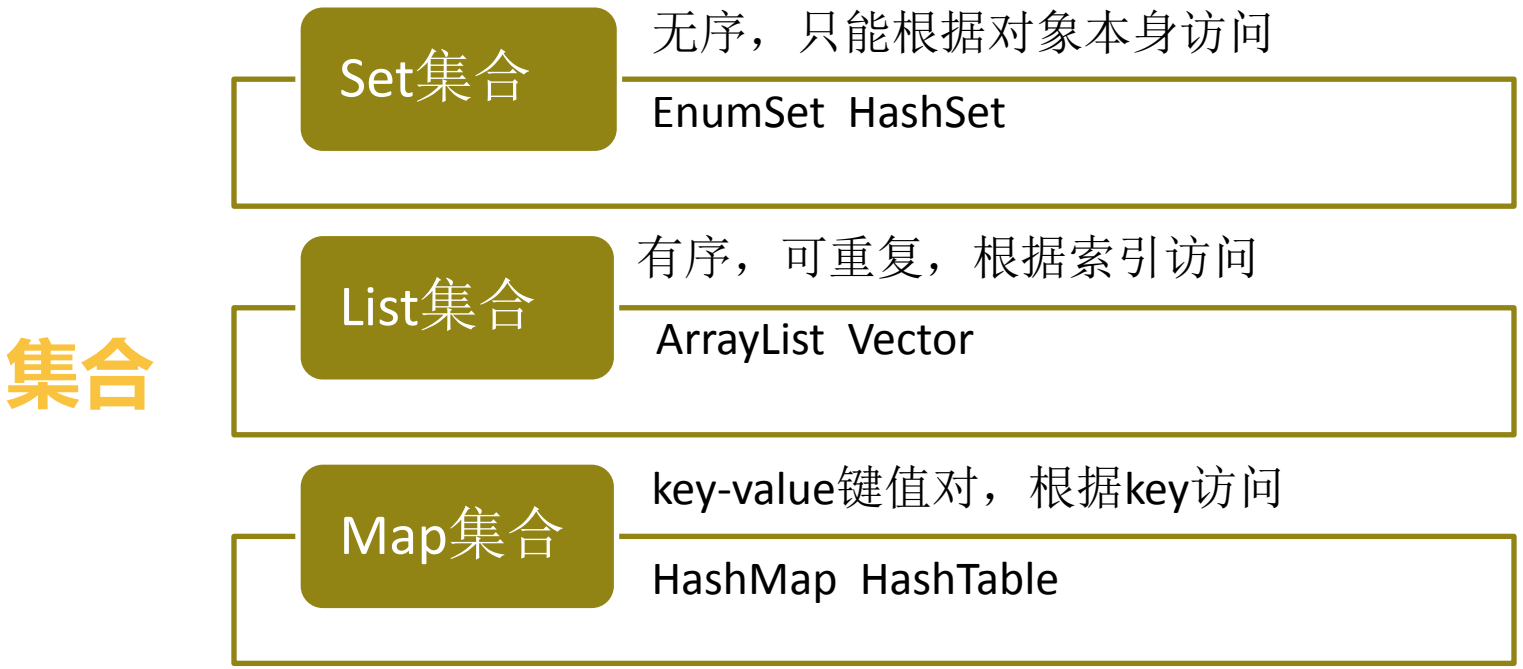
List集合装载的元素只能是DataBase的子类或自身

✱注：泛型的类型参数只能是类类型，不能是简单类型

泛型与集合

集合

- 集合类—数据结构
- 集合类—容器，存储不同种类和数量的对象，并按照规范实现一些常用的操作和算法，根据需要直接使用这些集合类并调用相应的方法。



JDK5.0后JAVA集合支持泛型。
JAVA所有集合类在java.util包下，主要由两个接口Collection和Map派生而成。

泛型与集合

ArrayList和Vector是List接口两个实现类,支持List接口所有功能方法

ArrayList：数组列表，非线程安全

Vector：向量，线程安全

二者是基于数组实现的列表集合，数组是动态的，可变的，允许再分配的Object[]数组。

```
ArrayList list=new ArrayList();//使用泛型ArrayList集合
list.add(“武汉”)//向集合中添加元素
List.add(1);//
```

```
ArrayList<String>list=new ArrayList<String>();//使用泛型ArrayList集合
list.add(“武汉”)//向集合中添加元素
List.add(1);//在编译阶段，编译器就会报错
```