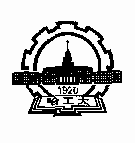
**《软件体系结构与设计模式》大作业报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 职务 | 学号 | 组内成绩 | 小组成绩 | 个人成绩 |
| 王少博 | 组长 | 181110315 |  |  |  |
| 王钟杰 | 组员 | 181110316 |  |  |
| 初征 | 组员 | 181110303 |  |  |
| 丁文琪 | 组员 | 181110304 |  |  |
| 付宽 | 组员 | 181110306 |  |  |
| 董成相 | 组员 | 181110305 |  |  |
| 凌龙 | 组员 | 181110311 |  |  |

**哈尔滨工业大学（威海）计算机科学与技术学院**

**2020年秋季学期**

**前 言**

一、本课程的主要任务

本课程主要讲述典型的软件设计模式与体系结构的特点及其典型应用。本课程还讨论了如何在软件体系结构的设计中应用具体的设计模式。通过对于软件设计模式与软件体系结构的学习，可以给学生提供大型软件设计的完整的新观点与新技术。通过本课程的学习，学生能够应用基本的软件设计模式与体系结构设计与编写Java程序，从而加深对面向对象设计原则的理解，为成为一名高水平的软件工程师打下基础。

二、本课程的要求

本课程涵盖10个设计模式与11个软件体系结构。课程的重点：讲清楚各个软件设计模式的结构与典型的交互；讲清楚各个软件体系结构的架构与各个组成组件之间的交互情况。课程的难点：设计模式的典型交互与实现，软件体系结构的各个层的典型交互与实现。教学内容与教学要求如下所述：工厂方法模式，抽象工厂模式，适配器模式，桥接模式，策略模式与状态模式，访问者模式，中介者模式，观察者模式；MVC架构与层次架构，主程序-子程序架构与面向对象架构，数据流系统，数据流体系结构-案例分析与程序设计，事件驱动体系结构-理论，客户端/服务器体系结构，P2P风格的体系结构与网格计算体系结构，SOA架构与云计算架构。

三、作业的验收内容（以小组为单位）

1. Java程序源代码（电子稿形式）；**要求使用Eclipes, Idea或者其它的Java集成开发环境(IDE)开发；你的程序要合理地分成若干个package以体现软件设计模式或软件体系结构。然后提交的时候，对于每个程序都要有一份可执行的xxx.jar文件与一个你的程序的源代码文件包（需要压缩）。**
2. 作业报告（电子稿形式）；
3. 将以上1）和2）中所包含的3项内容放到一个文件夹中，然后打成一个压缩包，[发送到邮箱mikesun725@aliyun.com](mailto:发送到邮箱mikesun725@aliyun.com)中，包名为张xx-王xx-李xx-赵xx-黄xx-秦xx；在包名中不要包含学生号。
4. 每个小组提交一份纸质作业报告（即2）所述报告的打印稿）。

四、本作业共13个题目，要求全部完成，共占课程总成绩的20%，每6-7人一组。

**Homework 2: 抽象工厂模式(Abstract Factory Pattern)**

**作业描述：The following class diagram represents a design in abstract factory pattern to query about the features of different types of buildings. See the source code for the implementation of the following class diagram.**

1. **Run the AbstractFactoryGUI first to be familiar with the program**

**BuildingFactory**

**+getFactory(type: String): BuildingFactory**

***+getHouse(): House***

***+getCondo(): Condo***

**<<Interface>>**

**House**

***+getHouseInfo()***

**<<create>>**

**<<create>>**

**SuperFactory**

**+getHouse(): House**

**+getCondo(): Condo**

**<<create>>**

**<<interface>>**

**Condo**

***+getCondoInfo()***

**MediumFactory**

**+getHouse(): House**

**+getCondo(): Condo**

**ClientGUI**

**main()**

**<<create>>**

**SupCondo**

**+getCondoInfo()**

**MedCondo**

**+getCondoInfo()**

**SupHouse**

**+getHouseInfo()**

**MedHouse**

**+getHouseInfo()**

**Fig 1 Design of building information system in the abstract factory pattern**

1. **In the avove design, add a class hierarchy named SemiDetacher with 2 subclasses named SupSemiDe and MedSemiDe and similar interface and methods as class hierarchy House and class hierarchy Condo do.**
2. **Then you need to modify the corresponding part in the class BuildingFactory and its subclass to allow the objects of subclasses of SemiDetacher be created**
3. **Implement the modified class diagram, adding possible code as needed.**
4. **Run object of the class AbstractFactoryGUI to test the program with new function.**

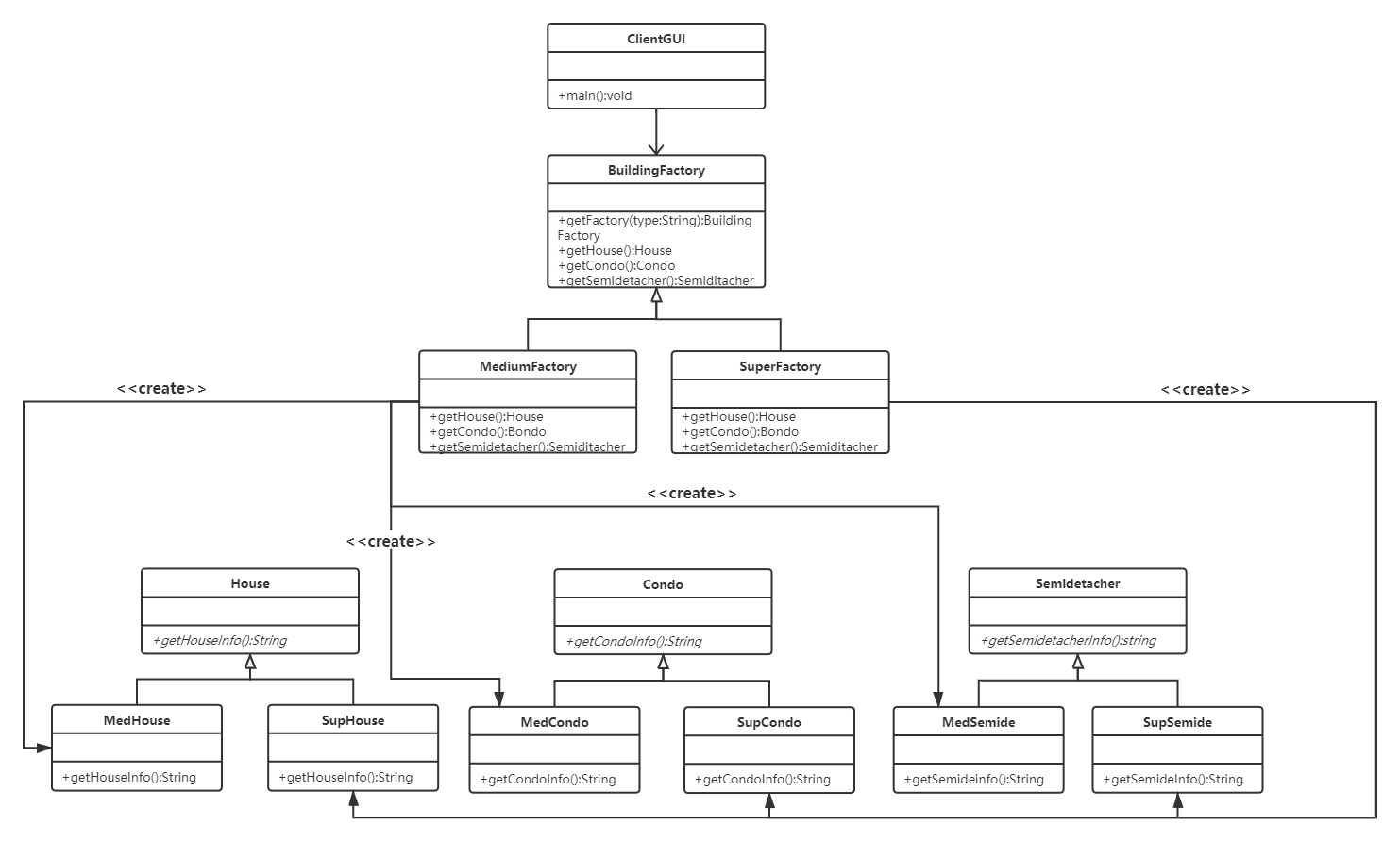
## 作业报告部分

**1. Describe your finished homework.**

* 1. **Tell me what classes have you added to the existing class hierarchy?**
  2. **Explain the relationship between the existing classes and the newly added classes.**

1. **添加了SupSemiDe,MedSemiDe,SemiDetacher三个类。**
2. **SemiDetacher是抽象类，SupSemiDe,MedSemiDe是其子类。SemiDetacher和House和Condo所处层次相同，都是房屋的抽象类；名字带Sup和Med的都是不同房屋的具体子类。通过抽象类BuildingFactory创建MediumFactory和SuperFactory两个工厂子类。这两个工厂子类分别负责Sup和Med两种不同规格房屋的创造。**

**2. Draw your new class diagram here**



**3. When a user chooses semi-detatcher and click on search, what methods will be called? List all of them in correct order.**

1. **首先调用GUI中ButtonListener类里的actionPerformed方法。（触发按钮）**
2. **cmbHouseClass.getSelectedItem()和cmbHouseType.getSelectedItem()，获用用户选中的房屋种类和房屋大小。**
3. **BuildingFactory.getFactory(clas) 获取相应房屋大小的房屋工厂类。（Med工厂或者是Sup工厂）**
4. **type.equals(ClientGUI. SEMIDETACHER)用户选择符合Semidetacher。**
5. **bf.getSemiDetacher() 通过创造的房屋工厂获得一个相应的房屋对象。**
6. **SemiDe.getSemiDetacherInfo() 通过刚刚获得的房屋对象获得房屋信息。**
7. **putHouseInfoToScreen(filename) 通过上一步获得的房屋信息（文件名称）将信息输出到GUI界面之中。**

**4. After you add the required class SemiDetacher hierarchy,**

* 1. **What classes in the BuildingFactory class hierarchy have been affected?**
  2. **What methods have you added to the class hierarchy?**
  3. **添加三个层次类后，还额外改变了BuildingFactory，MediumFactory和SupFactory这三个类。**
  4. **BuildingFactory中加入了抽象方法public abstract SemiDetacher getSemiDetacher();**
  5. **MediumFactory中加入了创造MedSemiDe的方法public SemiDetacher getSemiDetacher()**
  6. **SuperFactory中加入了创造SupSemiDe的方法public SemiDetacher getSemiDetacher()**

**5. Discuss the similarity and differences between factory method pattern and the abstract factory pattern in no more that 120 words.**

* 1. **工厂方法和抽象工厂方法的工厂都是具有层次关系的，具体工厂都是继承于(抽象)工厂父类。**
  2. **工厂方法的每一个工厂只负责生产一个对象。而抽象工厂方法中的每一个工厂负责生产一组对象。**
  3. **工厂方法一定遵循开闭原则，而抽象工厂方法不一定遵循开闭原则，是否遵守开闭原则由所扩展的内容决定。**
  4. **二者都实现了创建对象的责任分割，简化了复杂的条件语句。**

**6. Discuss about the method getFactory(String type) in the class**

**BuildingFactory.**

1. **What is the functionality of this class?**
2. **If we cancel this method from class BuildingFactory, then what do we need to do in the class AbstractFactoryGUI?**
3. **Point out advantages of the method getFactory(String type).**
4. **getFactory( ) 通过用户参数的输入生成Super或者Medium两种规格不同的工厂.**
5. **在GUI类的actionPerformed方法中添加对用户输入进行筛选的条件语句, 通过用户的输入直接创建不同规格的工厂.**
6. **BuildingFactory可以看作是创造具体工厂的抽象工厂, 通过这个类对工厂再次进行抽象, 实现了责任的分割,以及简化了繁琐的条件语句.**

**Homework 4: 桥接模式(Bridge Pattern)**

**作业描述（特工信息保密系统）：**

**The following design represents an agent information system. Agents’information should be encrypted and then saved into a text file or a database. There are two ways to encrypt agent’s name and code number, represented by classes EncryptedInfo1 and EncryptedInfo2.**

**EncryptedInfo1的加密算法：折叠算法**

1. **将26个英文字母按照顺序排列，加密的时候，以折叠的方式进行：**

**a🡨🡪z, b🡨🡪y, ...,m🡨🡪n**

1. **大写字母也是如此。大写字母加密为大写字母。**
2. **数字加密也以此方式进行0🡨🡪9,1🡨🡪8,2🡨🡪7,3🡨🡪6, 4🡨🡪5**

a b c d e f g h i j k l m n o p q r s t u v w x y z

0 1 2 3 4 5 6 7 8 9

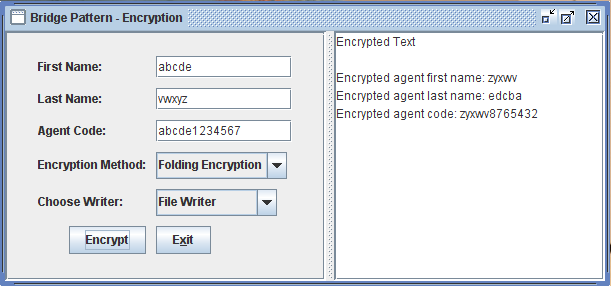
**EncryptedInfo2的加密算法：分组互换算法**

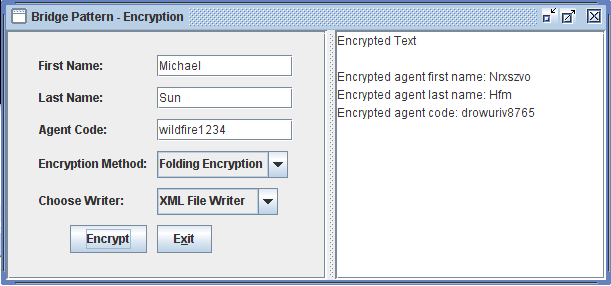
1. **将26个英文字母按照顺序排列，按照如下方式两两分成一组，本组内部加密的时候互换： a🡨🡪b, c🡨🡪d，…, w🡨🡪x, y🡨🡪z。**
2. **大写字母的加密方式也是如此，大写字母加密为大写字母。**
3. **数字加密：0🡨🡪1, 2🡨🡪3,4🡨🡪5, 6🡨🡪7, 8🡨🡪9**

a b c d e f g h i j k l m n o p q r s t u v w x y z

0 1 2 3 4 5 6 7 8 9

**程序运行界面如下**

****

****

**设计类图如下**

**<<interface>>**

**AgentInfo**

**+log(lNm: String, fNm: String, cd: String)**

**EncryptedInfo1**

**+encryptName(name: String)**

**+encryptCode(code: String)**

**+log(lNm: String, fNm: String,**

**cd: String)**

**-writer: MessageWriter**

**EncryptedInfo2**

**+encryptName(name: String)**

**+encryptCode(code: String)**

**+log(lNm: String, fNm: String,**

**cd: String)**

**-writer: MessageWriter**

**<<interface>>**

**MessageWriter**

**+logMsg(lNm: String, fNm: String, cd: String)**

**TxtFileWriter**

+logMsg(lNm: String,

fNm: String,

code: String)

**ClientGUI**

Encrypted.txt

Database

**XmlFileWriter**

+logMsg(lNm: String,

fNm: String,

code: String)

**DBWriter**

+logMsg(lNm: String,

fNm: String,

code: String)

AgentInfo.xml

**Fig. 1 Design class diagram for agent information system**

**Your task:**

**You are required to write a class EncryptedInfo3 to encrypt agent’s last name, first name and code number.**

1. **The class should be in the same format as EncryptedInfo1 and EncryptedInfo2, and provide a different encryption method**
2. **EncryptedInfo3 should encrypt the last name, first name and code number as described below as CaesarCypher: To encrypt a word, we replace the letter a with b, b with c, and so on, up to z, which is replaced by a. This is called the rotate-1 Caesar cipher. When a digital number between 0 and 9 is encountered, the rule to rotate is 0🡪1, 1🡪2,…,8🡪9 and 9🡪0. For example:**

**Plain text: Mike**

**Encrypted text: Njlf**

**Plain text: Sun**

**Encrypted text: Tvo**

**Plain text: Shanben56**

**Encrypted text: Tibocfo67**

**c) Modify the class ClientGUI to allow choice EncryptedInfo3 to be shown in the GUI.**

* **Note: 特工密码长度为12位，由英文字母与0，1，…，9阿拉伯数字组成。**
* **The length for a code is exactly 12, formed by mixed characters and digital numbers.**

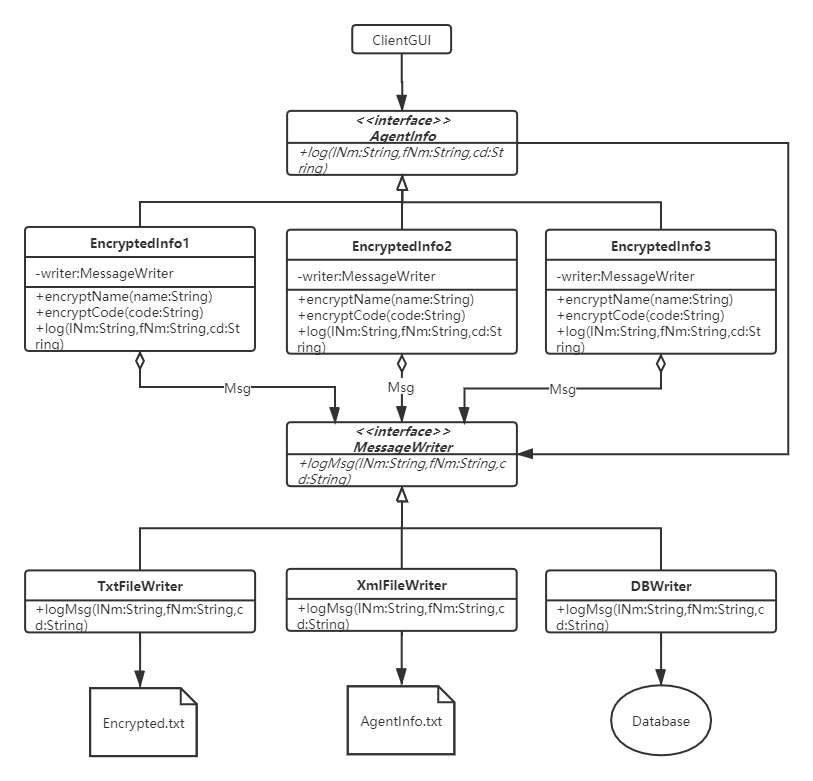
## 作业报告部分

**1. Describe your finished homework, including**

* 1. **What classes have been added to the existing class hierarchy?**
  2. **Explain the relationship between the existing classes and the newly added classes.**

1. **我添加了EncryptedInfo3类。**
2. **这个类继承自AgentInfo类，与其余的两个Encryption处于同一个层次。同时，这个类继承了MessageWriter。**

**2. Draw your new class diagram here**



**3. Answer questions as below: after you add class EncryptedInfo3,**

1. **Do you need to recompile class EncryptedInfo1 and EncryptedInfo2?**
2. **Do you need to recompile class AgentInfo?**
3. **Do you need to recompile class FileWriter or DBWriter?**
4. **不需要重新编译EncryptedInfo1和EncrypetdInfo2。**
5. **不需要重新编译AgentInfo**
6. **不需要重新编译FileWriter和DBWriter。**
7. **这就是桥接模式的优点。**

**4. In class EncryptedInfo1, there is a constructor**

**public EncryptedInfo1(MessageWriter l){**

**writer = l;**

**}**

**Why do we use type MessageWriter as the parameter type and why don’t we use type FileWriter or DBWriter as the parameter type?**

1. **首先，根据我们一贯的设计原则，要尽量依赖于抽象而不是依赖于具体。**
2. **由于依赖的是抽象类，解除了和具体类的耦合。新增具体类和修改具体类时，不会受到牵连。**
3. **可以实现动态配置，也就是在运行的时候才决定使用哪个子类，做到了动态。**
4. **改变抽象类的时候不需要重新编译客户类。**
5. **对用户隐藏了实现的细节。**

**5. In class ClientGUI , we have the source code**

**MessageWriter writer;**

**…**

**if( logWay.compareTo(ClientGUI.DBWRITER)==0 )**

**writer = new DBWriter();**

**if( logWay.compareTo(ClientGUI.FILEWRITER)==0 )**

**writer = new FileWriter();**

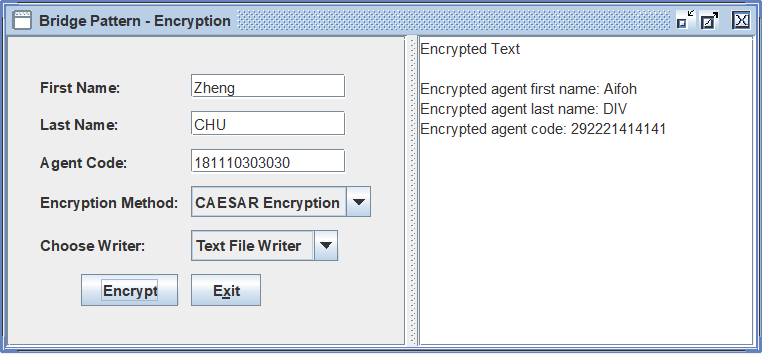
**What is the advantage of writing code above?**

1. **不采用具体类而是采用抽象类，降低了耦合度。**
2. **修改具体类的时候不需要重新编译。**
3. **添加新具体类的时候只需要修改少量客户代码的条件语句。**

**6. If we add a class named ConsolWriter parallel to DBWriter, do we need to recompile any classes?**

**不需要重新编译。因为客户类依赖的是抽象而不是具体。**

**7. Test your finished program. List your test input and output from running your program.**



**Homework 6: 策略模式(Strategy Pattern)**

**作业描述(This Homework needs only a little bit coding)：The following class diagram represents a design in strategy pattern to sort integer arrays using different algorithms. This homework requires you**

1. **Run the program ClientGUI first to see how an integer array is sorted.**
2. **Add a strategy sub class to implement SortAlgorithm to do Bidirectional bubbleSort. Thus you can name this class BidirBubbleSort. Note that the BidirBubbleSort.java file is already in the same package and what you need to do is just add it to the strategy class hierarchy.**
3. **In ClientGUI, modify the JcomboBox part to add an option to allow bidirectional bubble Sort.**

**<<interface>>**

**SortAlgorithm**

**+sort(a: int [ ], c: Context)**

**BubbleSort**

**+sort(a:int[],c:Context)**

**ClientGUI**

**HeapSort**

**+sort(a:int[],c:Context)**

**Context**

+Context(s: SortAlgorithm)

+sortIntArray(): int[]

+startExecution(): void

+endExecution(): void

+getExeTime(): long

context

sort

**InsertionSort**

**+sort(a:int[],c:Context)**

**QuickSort**

**+sort(a:int[],c:Context)**

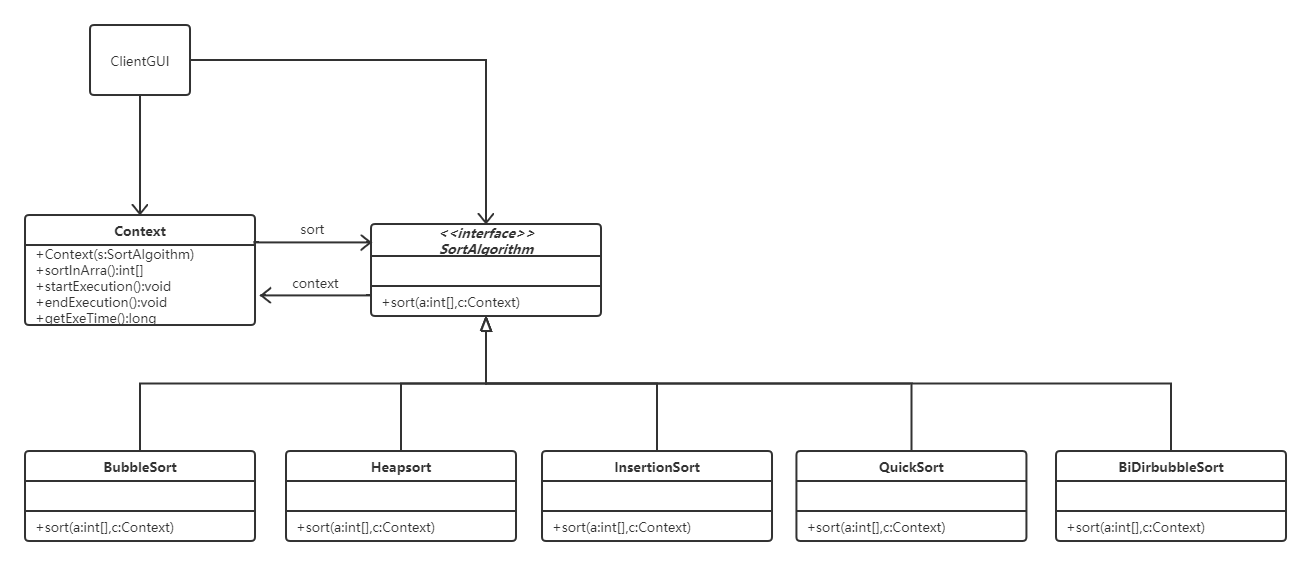
**Fig 1. The Design of the Sorting Integer Array Problem in example 7.2 of the book**

## 作业报告部分

**1. Describe your finished homework, including**

* 1. **What classes have been added to the existing class hierarchy?**
  2. **Specify the relationship between the existing classes and the newly added classes.**
  3. **What code has been added to what class?**
  4. **BidirBubbleSort类。**
  5. **BidirBubbleSort类继承自SortAlgorithm接口，Context类聚合一个具体的算法对象，再在这个算法对象中调用Context，进行排序。**
  6. **ClientGUI加入了一些界面的代码，以及选择双向排序的条件语句。在BidirBubbleSort类中加入了计时器内容。**

**2. Draw your new class diagram here**



**3. Describe the typical interactions of the design in Fig 1 above in details**

* 1. **用户首先创建Context类，然后聚合一个特定的算法类。**
  2. **调用context类的排序方法。**
  3. **Context的排序方法中调用algorithm类的排序方法。**
  4. **algorithm类的排序方法中调用context类的计时器方法。**
  5. **最后在GUI中显示计时和排序信息。**

**4. Discussions of the extensibility issues.**

**1) After you add your new class BidirBubbleSort in the strategy class hierarchy,**

* 1. **Do you have to change the Context class?**

No

* 1. **Do you have to change the Client class ClientGUI?**

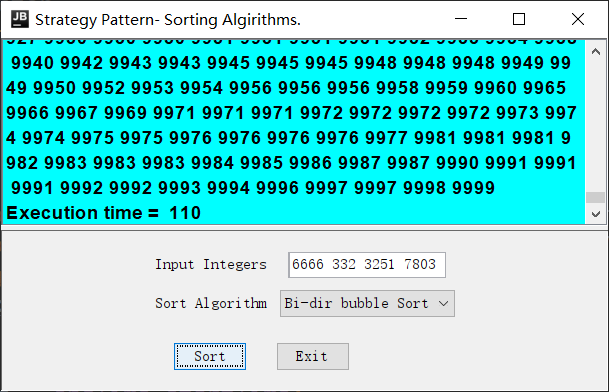
Yes

**2) If you have changed the code for a sort(a:int[],c:Context) method in a sub-strategy class in the strategy class hierarchy,**

* 1. **Do you have to change or recompile the** **Context class?**
  2. **Do you have to change or recompile the Client class ClientGUI?**

**不需要重新编译Context和ClientGUI类。**

**5. Test your program by outputing typical input and output from running your new program**



**Homework 9: 中介者模式(Mediator Pattern)**

**Homework Description：See the following class diagram for the Mediator pattern implementation of collaboration program for Hotel, Airline and Tour.**

**HotelGUI**

-m: BusinessMediator

+addCustomer(String c);

+addPossibleCustomer(String c)

+displayInfo(String txt)

+writeCusToXmlFile(String[] c)

+writeCandidateCusToXmlFile(c)

**TourGUI**

-m: BusinessMediator

**BusinessMediator**

companyList: ArrayList <ParticipantGUI>

+register(c: CompanyGUI)

+addPossibleCus(p: ParticipantGUI ,

text String)

+updateAllGuis(p: ParticipantGUI,

text String)

+writePossibleCusToXmlFile(p: ParticipantGUI ,

cusData: String[])

**AirlineGUI**

-m: BusinessMediator

**<<interface>>**

**ParticipantGUI**

*+addCustomer(String c);*

*+addPossibleCustomer(String c)*

*+displayInfo(String txt)*

*+writeCusToXmlFile(String[] c)*

*+writeCandidateCusToXmlFile(c)*

**Client**

+main()

+addCustomer(String c);

+addPossibleCustomer(String c)

+displayInfo(String txt)

+writeCusToXmlFile(String[] c)

+writeCandidateCusToXmlFile(c)

+addCustomer(String c);

+addPossibleCustomer(String c)

+displayInfo(String txt)

+writeCusToXmlFile(String[] c)

+writeCandidateCusToXmlFile(c)

HotelCustomer.xml

HotelPossibleCustomer.xml

**Airport**Customer.xml

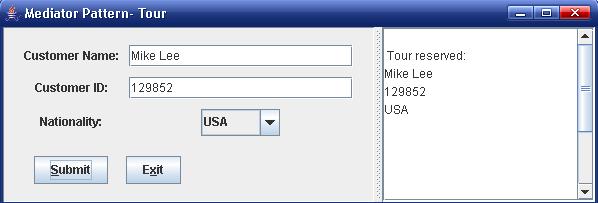
AirportPossibleCustomer.xml

TourCustomer.xml

TourPossibleCustomer.xml

**图1 利用中介者模式设计的信息共享程序**

**The three graphical user interfaces for the Hotel, Airline and Tour are as below.**



**图2 旅游公司输入客户信息**



**图3 宾馆界面同步获得旅游公司输入的客户信息**



**图4 机场界面同步获得旅游公司输入的客户信息**

当在一个用户图形界面，例如Tour，中输入Mike Lee，1239852，选择USA然后点击按钮“Submit”，则程序会通过BusinessMediator对象，将这些客户信息转发给Hotel和Airline对象，并且显示在相应的用户图形界面上。这些客户数据也被写到了以下的数据文件TourCustomer.xml，HotelPossibleCustomer.xml和AirportPossibleCustomer.xml中。

**Homework requirement:**

1. **Run the program before you go any further**
2. **(Coding) Add another class named** **TouriststoreGUI （旅游商店）such that the 4 classes HotelGUI, AirflightGUI, TourGUI and TouriststoreGUI will interact though the BusinessMediator in the same way as before (use the mediator pattern). Note that you need to create files TourStoreCustomer.xml and TourStorePossibleCustomer.xml and relavent customer information needs to be added to the above files.**

项目报告部分

**1. Describe your finished homework, including**

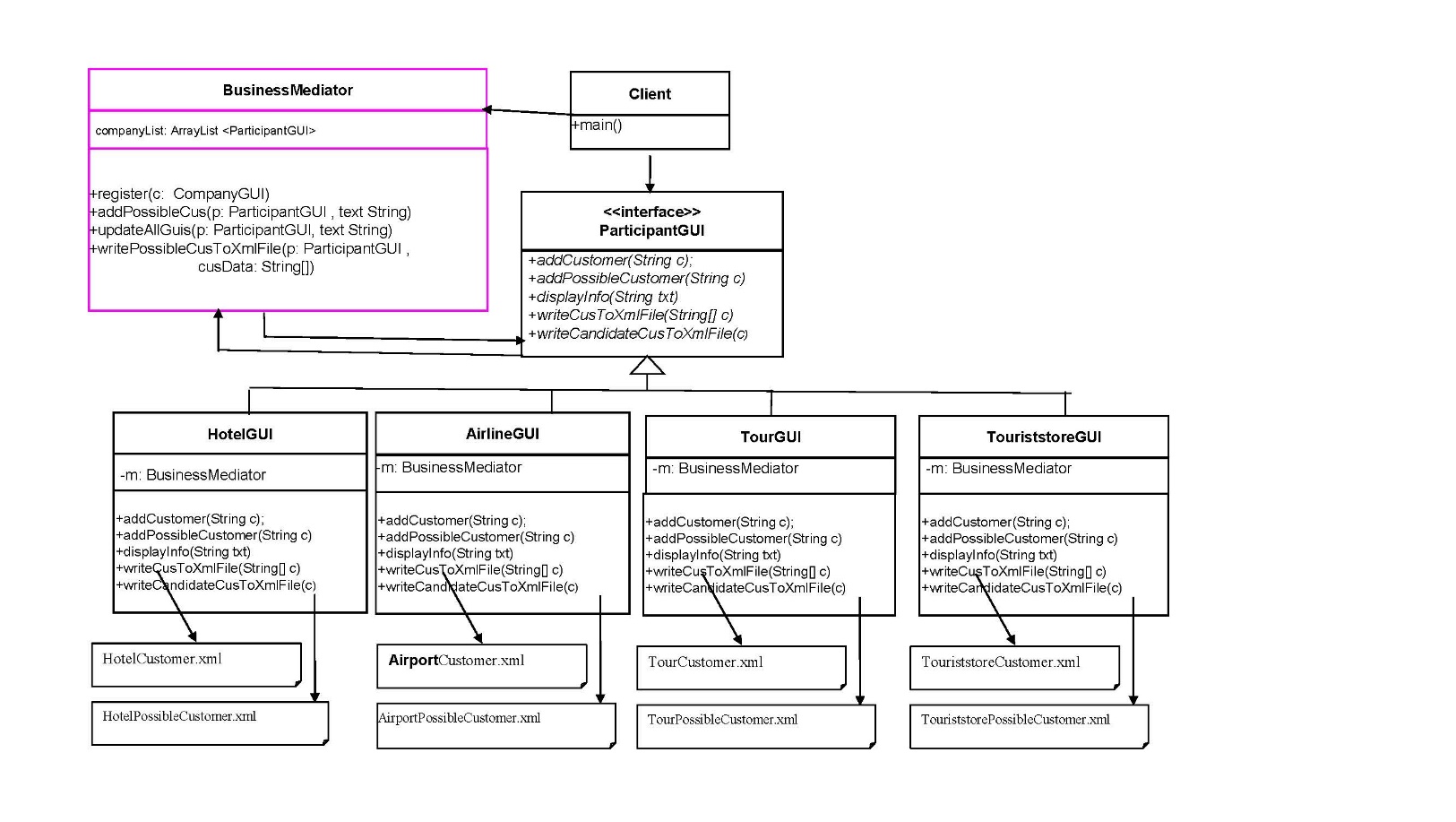
* 1. **What class has been added to the existing class hierarchy?**
  2. **The relationship between the existing classes and the newly added class**

**Answer:**

**a) TouriststoreGUIc class**

**b) The relationship between class ParticipantGUI and the newly added class is an inheritance relationship and the newly added class inherits from ParticipantGUI, and the relationship between class BusinessMediator and the newly added class is an aggregation relationship.**

**2. Draw your new complete class diagram here**



**3. In HotelGUI object, when you enter name entry as “Rose Black”, id number entry as “555888999”, and choose the nationality as “USA”, and then click on button “Submit” to save the customer information into file HotelCustomer.xml, what methods in what class will have been invoked, list all of them in correct order.**

**Answer:**

**The methods in HotelGUI**

* setCusName();
* setCusID();
* setCusNation();
* getCusName();
* getCusID();
* getCusNation();
* writeCusToXmlFile()；
* askMedSaveCusInfo();

**The methods in BusinessMediator**

* bMediator.writePossibleCusToXmlFile();

**The methods in HotelGUI**：

* displayInfo()；
* addCustomer();
* askMedAddCus();

**The methods in AirflightGUI：**

* addPossibleCustomer()；

**The methods in TourGUI：**

* addPossibleCustomer()；

**The methods in TouriststoreGUI：**

* addPossibleCustomer()；

**The methods in HotelGUI：**

* askMedUpdate()；

**The methods in BusinessMediator**

* updateAllGuis()；

**The methods in AirflightGUI：**

* displayInfo();

**The methods in TourGUI：**

* displayInfo();

**The methods in TouriststoreGUI:**

* displayInfo()

**4. Discussion**

1. **Describe the functionality of the** **BusinessMediator class.**
2. **Explain how the mediator pattern works?**
3. **How the method updateAllGuis(****ParticipantGUI p, String text) in the BusinessMediator class works?**

**Answer**

1. **The BusinessMediator will keep a reference of each of the interacting objects(HotelGUI, AirflightGUI, TourGUI and TouriststoreGUI) and receive the message from all the objects,and provide methods to call the participating objects(HotelGUI, AirflightGUI, TourGUI and TouriststoreGUI).**
2. **Firstly, Mediator object call the methods named *register* to save objects. A object calls a method of mediator object, and then the mediator object calls methods in other objects. These objects also call back the methods in Mediator object. Mediator object calls the object .Mediator object in turn call other objects**
3. **An Iterator is defined that iterates through the companyList elements (ParticipantGUIl instantiation object), and if the element is not p, the displayInfo method of the element (ParticipantGUIl instantiation object) is called to print the information.**

**5. [Extendibility Issue] Discussion:**

1. **After you add the new class TouriststoreGUI, do you have to modify class ParticipantGUI?**
2. **After you add the new class TouriststoreGUI, do you have to modify class HotelGUI?**
3. **After you add the new class TouriststoreGUI, do you have to modify class AirlineGUI?**
4. **After you add the new class TouriststoreGUI, do you have to modify class TourGUI?**
5. **After you add the new class TouriststoreGUI, do you have to modify class BusinessMediator?**

**Answer: I don’t need to modify all these classes.**

# Project 1: Encryption by Using Layered Architecture

**问题描述：The purpose of this project is to design an agent information system by using three-layer layered architecture. The agent information is encrypted and then saved into some specific text files. In the design below, the graphical user interface includes a class nemed ClientGUI for sending user’s request. The application layer contains a class hierarchy named Encryption to actually encrypt agent info and calls class TxtFileWriter, which saves the encrypted data into files Folding.txt, and Group-swap.txt. The agent information to be encrypted includes agent’s last name, first name, and agent code. The agent code can be mixed characters with digital numbers. The length of the code is 12.**

**ClientGUI**

**TxtFileWriter**

**Encryption**

**EncryptedInfo1**

**EncryptedInfo2**

Folding.txt

Group-swap.txt

表示层

应用层

永久数据存储层

**EncryContext**

**Fig 1. Software architecture for the agent information system**

**算法描述：**

**Class EncryptedInfo1 encrypts agent information using Encryption Algorithm1 (折叠算法) as below.**

**Precondition: a text file to be encrypted contains only English characters and digital numbers. The algorithm is:**

**a🡪z, b🡪y, ...m🡪n, y🡪b, z🡪a**

**0🡪9, 1🡪8, 2🡪7, 3🡪6, 4🡪5, 5🡪4, 6🡪3, 7🡪2, 8🡪1, 9🡪0**

**Upper case letters are also encrypted the same way (Upper case letters are encrypted into Upper case letters).**

**Class EncryptedInfo2 encrypts agent information using Encryption Algorithm2 (分组互换算法).**

**Precondition: a text file to be encrypted contains only English characters and digital numbers. The algorithm is:**

**a🡪b, b🡪a; c🡪d, d🡪c; e🡪f, f🡪e;, ..., y🡪z, z🡪y**

**0🡪1, 1🡪0; 2🡪3, 3🡪2; ...8🡪9, 9🡪8**

**Upper case letters are also encrypted the same way (Upper case letters are encrypted into Upper case letters) .**

**Class diagram is as Fig 2.**

**The source code for the above class diagram has been written in Java. The encrypted agent information from EncryptedInfo1 is saved into a file named Folding.txt and encrypted agent information from EncryptedInfo2 is saved into a file named Group-swap.txt. Both text files are in the same folder as the Java source files.**

**The class diagram is as below.**

**<<interface>>**

**Encryption**

*+encryptName(inputStr: String): String*

*+encryptCode(inputSt: Stringr): String*

*+log(lNm: String, fNm: String, cd:String): void*

**EncryptedInfo1**

+encryptName(inputStr: String): String

+encryptCode(inputSt: Stringr): String

+log(lNm: String, fNm: String, cd:String)

-writer: TxtFileWriter

-writer: TxtFileWriter

+encryptName(inputStr: String): String

+encryptCode(inputSt: Stringr): String

+log(lNm: String, fNm: String, cd:String)

**ClientGUI**

**EncryptedInfo2**

**TxtFileWriter**

+logMsg(encryptOption: String, lNm: String, fNm: String, cd:String)

Folding.txt

Group-swap.txt

**EncryContext**

+doEncryption(lastNm:String, firstNm: String, code: String, txt: JTextArea)

-en: Encryption

**Fig 2. The class diagram for the agent information system**

## 你的任务：

**1) 在以上已经存在的设计中，新增加第三种加密算法. Now you are required to write a class EncryptedInfo3 to encrypt agent’s last name, first name and code number.**

**a) The class should be in the same format as EncryptedInfo1 and EncryptedInfo2, and provides a different encryption method**

**b) EncryptedInfo3 should encrypt the last name, first name and code number as described below as CaesarCypher: To encrypt a word, we replace the letter a with b, b with c, and so on, up to z, which is replaced by a. This is called the rotate-1 Caesar cipher. When a digital number between 0 and 9 is encountered, the rule to rotate is 01, 12,…,89 and 90. For example:**

**Agent first name: Mike**

**Encrypted agent first name: Njlf**

**Agent last name: Sun**

**Encrypted agent last name: Tvo**

**Agent code: WildWolf7489**

**Encrypted agent code: XjmeXpmg8590**

**c) Modify the class ClientGUI to allow choice EncryptedInfo3 to be shown in the GUI.**

**d) The encrypted agent information should be saved into a text file Caesar.txt.**

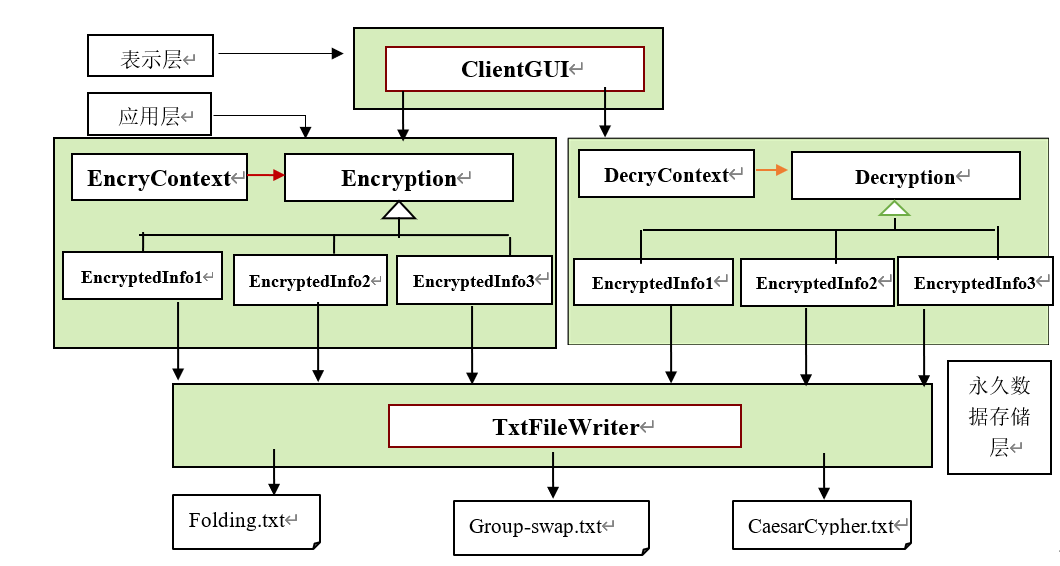
**2) 在应用层，增加相应的解密算法层次类Decryption，该类带有三个相应的解密子类。分别从三个加密文件中读出的agent加密信息，可以通过此层次类解密，然后显示在用户图形界面上。**

**3) 在用户图形界面上增加解密按钮，以便对各个加密文件所存的内容进行解密，然后显示的用户图形界面上。**

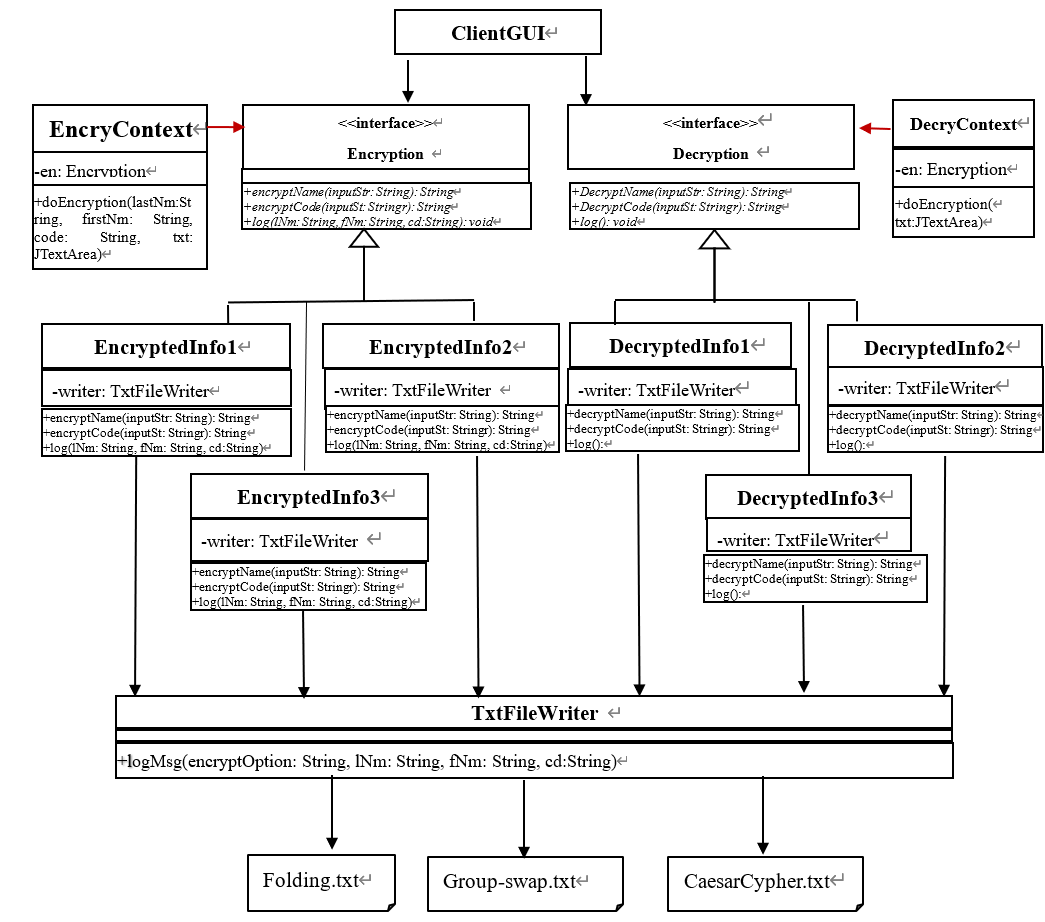
**【注】为了方便起见，你也可以使用另外一个文件夹包含全部的解密程序和text文档。**

## 项目报告部分

1. **Draw your software architecture for your new agent information encryption system**



1. **Draw class diagram for your new agent information encryption system**



1. **To implement all the required functionalities, what classes have you added to the application layer?**

**答：在应用层添加的类有EncryptedInfo3，新添加的编码方法类，DecryContext，Decryption接口，DecryptedInfo1, DecryptedInfo2, DecryptedInfo3,三个用于解码的方法类。**

1. **Which design pattern has been used in the application layer?**

**答：使用了桥接模式(Bridge Pattern)**

1. **Answer question: When you add a new class in the application layer, will the data file access layer be affected or not?**

**答： 不会。在应用层新加一个类的时候，因为数据层和应用层是隔离开封装的，不会影响数据层的信息。**

1. **Answer question: When you add a new class in the application layer, will the user interface layer be affected or not?**

**答：不会，图形界面层和应用层以及数据层之间都封装了，因此在应用层之间类方法等不会影响其他层次。**

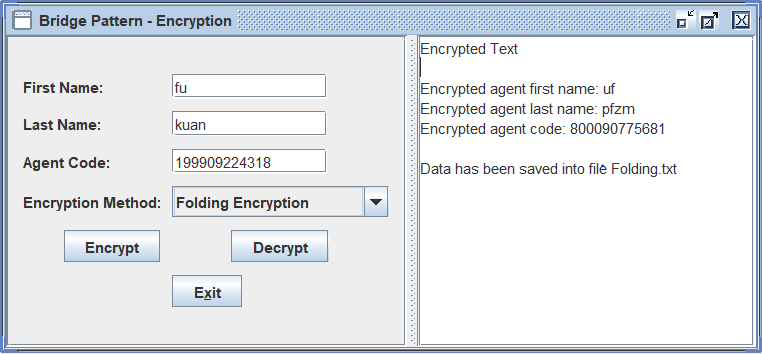
1. **Answer question: When you add a new method in the TxtFileWriter class, will the user interface layer or the application layer be affected or not?**

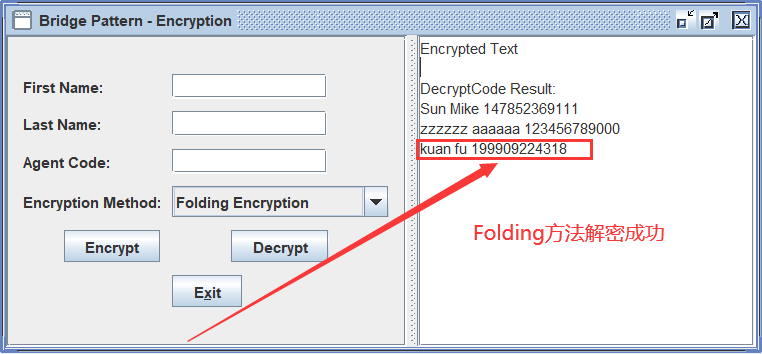
**答：不会。在TxtFileWriter class中添加方法的时候，除了TxtFileWriter class会受到影响之外，其余的图形界面层和应用层之间，是不会收到影响的，因为各自部分已经封装。**

1. **If you want the user interface layer to initialize a request to save some data into the data files, what classes need to call?**

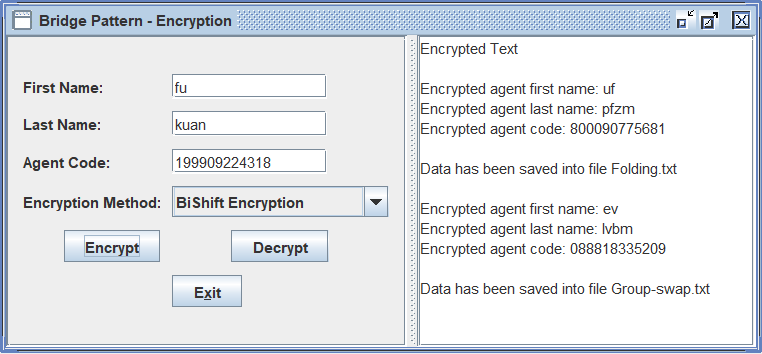
**答：需要调用的类有：ClineGUI类，Encrycontext类，Encryption类。然后加密算法类EncryptedInfo1，EncryptedInfo2，EncryptedInfo3，其中之一，最后使用TextFileWriter类和FileUtil类进行写入文件。**

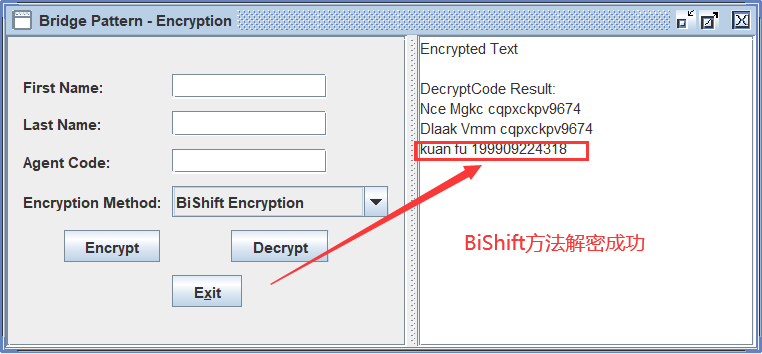
1. **In your program,** 
   1. **Does the application layer call the user interface layer?**
   2. **Does the data files access layer call the application layer or the user interface layer?**
2. **应用层会调用用户界面层获取用户选择进行对应的执行加密算法**
3. **数据文件访问层调用应用层，从应用层获取对应的加密好的信息，然后写入文件进行存储，不调用用户界面层。**
4. **测试与验证部分: List at least 3 group of typical input and output from running your program, corresponding to 3 ways to encrypt and decrypt agent information**
5. **Folding：**



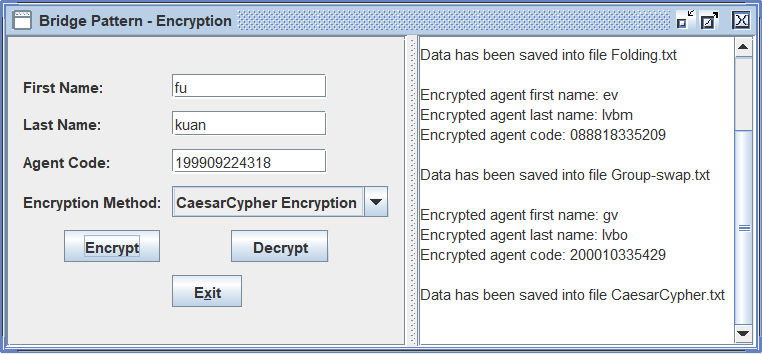


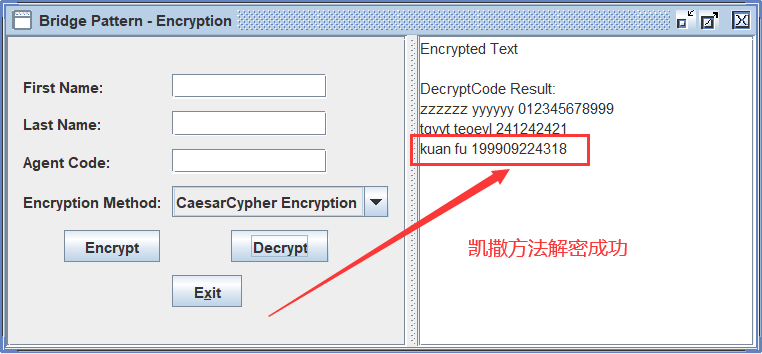
1. **BiShift：**





1. **Caesar：**





**Project 3: Text File Encryption in Pipes-and-filters Architecture**

## 项目需求部分：实现一个多次加密程序

**Do text file encryption by using the pipes and filters architecture. Suppose that we have 3 encryption algorithms as following:**

**Encryption Algorithm 1 (折叠算法). Precondition: a text file to be encrypted contains only English characters and digital numbers. The algorithm is:**

**a🡪z, b🡪y, ...m🡪n, y🡪b, z🡪a**

**0🡪9, 1🡪8, 2🡪7, 3🡪6, 4🡪5, 5🡪4, 6🡪3, 7🡪2, 8🡪1, 9🡪0**

**Upper case letters are also encrypted the same way (Upper case letters are encrypted into Upper case letters).**

**Encryption Algorithm 2(分组互换算法). Precondition: a text file to be encrypted contains only English characters and digital numbers. The algorithm is:**

**a🡪b, b🡪a; c🡪d, d🡪c; e🡪f, f🡪e;, ..., y🡪z, z🡪y**

**0🡪1, 1🡪0; 2🡪3, 3🡪2; ...8🡪9, 9🡪8**

**Upper case letters are also encrypted the same way (Upper case letters are encrypted into Upper case letters) .**

**Encryption Algorithm 3 (Caesar cipher, rotation 1): Precondition: a text file to be encrypted contains only English characters and digital numbers. The algorithm is: for the 26 English characters, you replace the letter a with b, b with c, and so on, up to z, which is replaced by a. When a digital number between 0 and 9 is encountered, the rule to rotate is 0🡪1, 1🡪2,…,8🡪9 and 9🡪0.**

**a🡪b, b🡪c; c🡪d, d🡪e; e🡪f, …,y🡪z, z🡪a**

**0🡪1, 1🡪2; 2🡪3, 3🡪4; 4🡪5, 5🡪6,6🡪7,7🡪8, 8🡪9, 9🡪0**

**Upper case letters are also encrypted the same way (Upper case letters are encrypted into Upper case letters)**

**The program should be designed using pipes and filters architecture, which contains filters as below:**

1. **InFilter reads the file to be encrypted in character stream format and parses the file. The parsed data is written to the output pipe as char stream;**
2. **EncrFilter1 reads character stream from its source pipe, use Encryption algorithm 1 to encrypt the incoming characters and numbers, and then write the resultant data into sink pipe as character stream;**
3. **EncrFilter2 reads character stream from its source pipe, use Encryption algorithm 2 to encrypt the incoming characters and numbers, and then write the resultant data into sink pipe as character stream;**
4. **EncrFilter3 reads character stream from its source pipe, use Encryption algorithm 3 to encrypt the incoming characters and numbers, and then write the resultant data into sink pipe as character stream;**
5. **OutFilter reads character stream from the source stream (through source pipe), and then write the resultant data into a file named Updated.txt in directory “EncryptedFiles” and onto the right textArea on the GUI (the original file content will be shown on the left textArea)**

**The logic design of the pipeline is as the following.**

**Sale1.txt**

**EncryptedSale1.txt**

**Screen**

**EncrFilter1**

**EncrFilter2**

**OutFilter**

**InFilter**

**EncrFilter3**

**Fig 1. The logical design of the encryption system**

**The class diagram design is as below.**

**Pipe**

-reader\_: PipedReader

-writer\_: PipedWriter

+read()

+write(c: int)

+closeReader()

+closeWriter()

**<<Interface>>**

***Runnable***

+run()

**Filter**

#input: Pipe

#output:Pipe

-isStarted:boolean

+Filter(input:Pipe,output:Pipe)

+start()

+stop()

+run()

#process()

**InFilter**

+process()

**EncrFilter1**

+process()

**EncrFilter2**

+process()

**OutFilter**

+process()

**EncrFilter3**

+process()

**AssemblyOfPipeLines**

**PipeFilterTestGui**

**Fig 2. The design of encryption program using pipes and filters architecture**

## 你的任务：

1. **在以上设计中，InFilter类，EncrFilter1类，和OutFilter类已经实现。你的任务是写代码实现EncrFilter2类和EncrFilter3类（提示：在写代码之前，运行程序，会对你的理解有所帮助）。**
2. **要求你的经过3次复合加密过滤器处理的文件，显示在图形界面右端的文本框中和相应的文件夹EncryptedFiles中的相应的文件EncryptedSalesman1, EncryptedSalesman2,或者EncryptedSalesman3之中。**
3. **与加密类似地，按照以上类似的管道-过滤器架构设计并且实现一个针对以上算法的解密系统，可以将以上的整个加密程序，包括用户图形界面类，复制为另外一个文件夹，然后少许修改代码，而成为一个解密系统。注意，解密系统的过滤器的顺序与加密系统过滤器的顺序相反。**
4. **对解密系统进行测试：使用已经被三次加密的数据，进行解密运算，如果最后得到未加密之前的原始数据，则说明你的加密系统是正确的（目的是验证你的加密算法的正确性）。**

## 项目报告部分(Report Part)

1. **Answer question: When you add a new sub class of Filter, what class needs to be modified in order for using the new sub class?**

**答：AssemblyOfPipeLines类需要修改。需要在这个类中创建新的管道，以及创建新的EncryFilter对象。并且使用多线程同时运行这些encryptFilter对象。**

1. **Answer question: What makes the data flow through the pipeline?**

**答：首先InFilter读入一个字符，写入Output端。之后EncrFilter从In的输出端读取一个字符，并且进行处理后写入输出端。每一个EncrFilter读取上一个的输出，之后处理发送到自己的输出端，最后由OutPutFilter读取并且输出到屏幕上。**

1. **Answer question: What “all the filters can work simultaneously” mean?**

**答：与批处理不同，批处理必须等前一个任务结束了，后面一个任务才能使用前面一个任务的输出作为自己的输入开始执行。他们的执行是依次序执行。 而管道过滤器则可以使用多线程的方式，多个处理程序同时运行。一次读取一个字符，之后处理并且输出。**

**4. 验证部分**

* 1. **List typical input and output from running your encryption program**
  2. **use your decryption system to decrypt the output file from encryption to get the original file (如果你解密以后得到了原始文件，则证明你的加/解密算法的实现是正确的).**

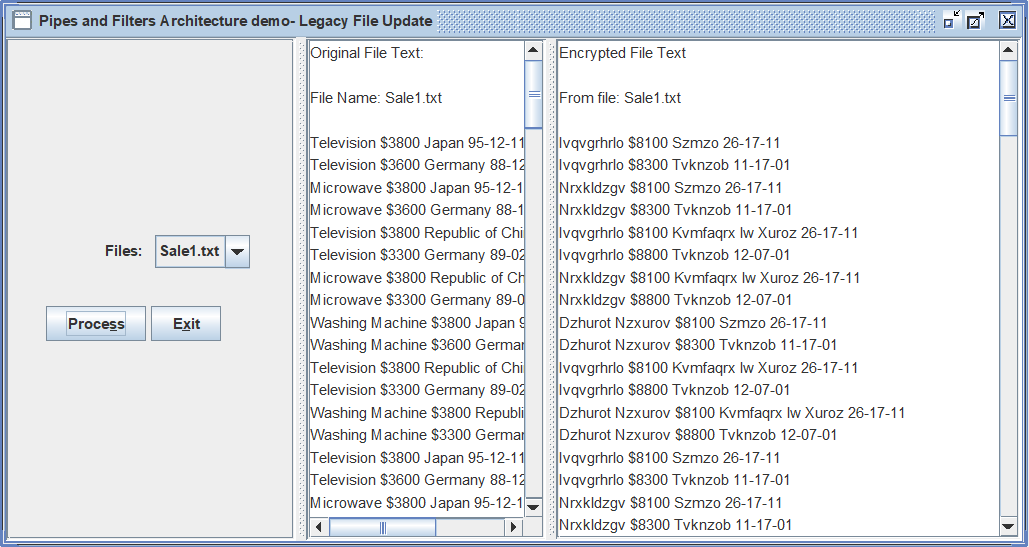


图 1 - 加密

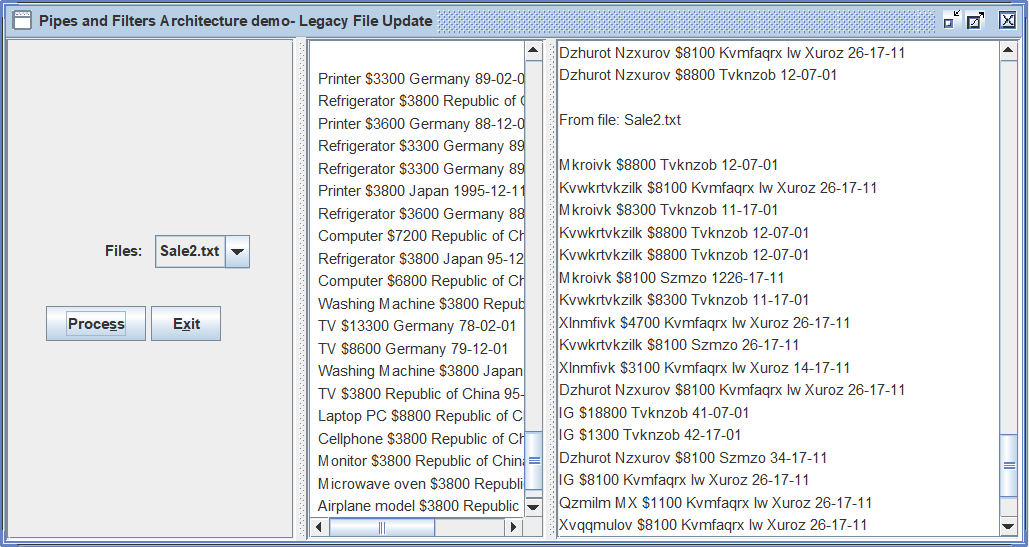


图 2 - 加密

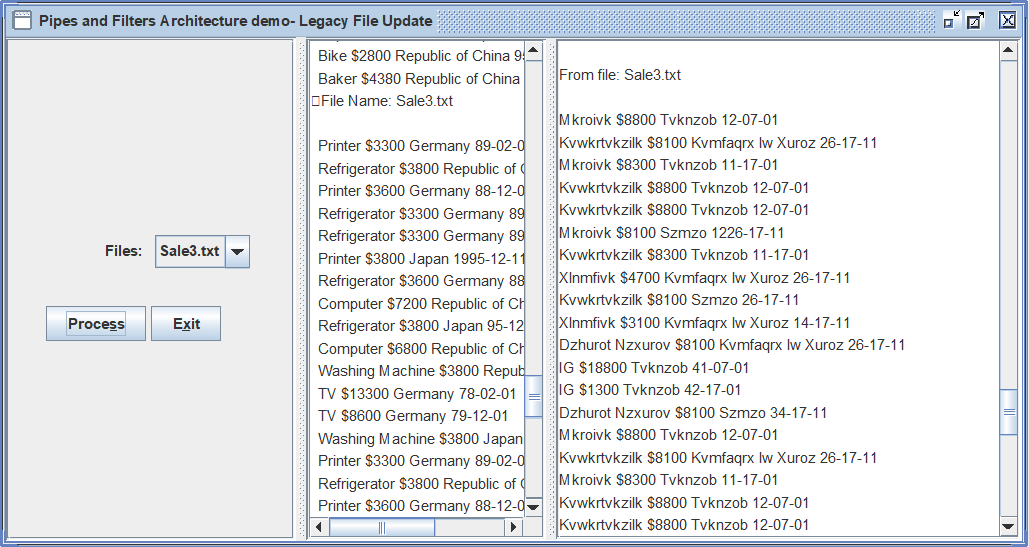


图 3 - 加密

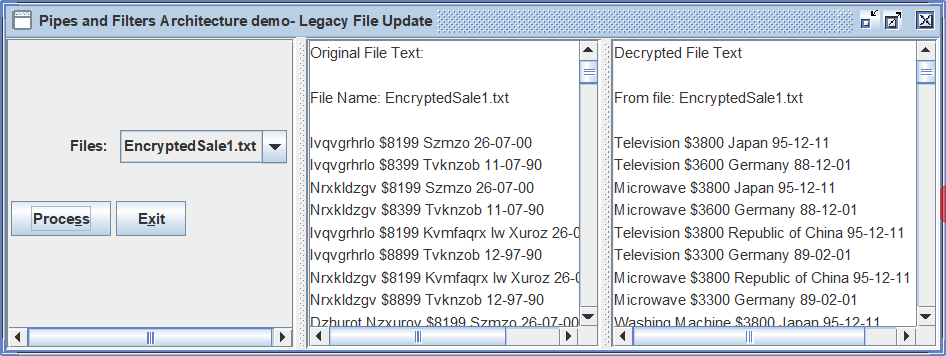


图 4 - 解密

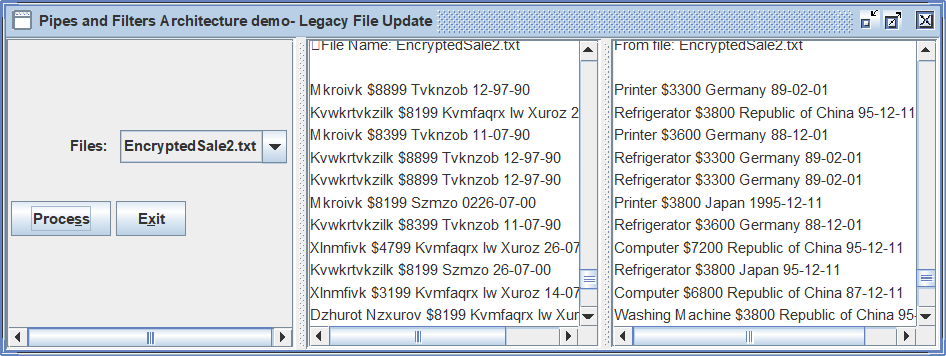


图 5 - 解密

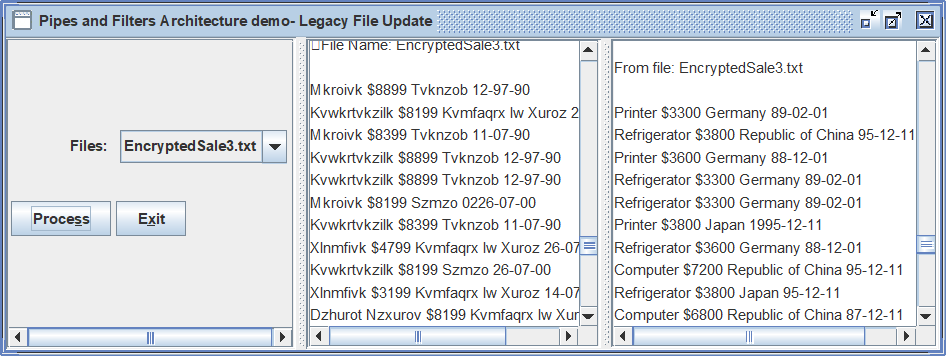


图 6 - 解密