CSCM94 2022

Q1 – 概念题

(a) – 莱斯格(Lessig)的四种规约

Question 1

a) Cloud computing has become the foundation upon which much of our frequently used digital apps are built upon. Platforms such as Amazon Web Services, Microsoft Azure and Google Cloud have changed the way that data is stored and the ability for systems to access enhanced computing power. Cloud computing can appear in our homes through smart speakers, supporting the social network services we use and everyday digital services such as calendars and emails.

Define and explain how to envision cloud computing interacting with each of Lessig's Four Modalities.

[8 marks]

> 定义云计算，并解释其如何与莱斯格（Lessig）的四种规约（法律、社会规范、市场、架构/代码）相互作用

Four Modalities

- **Code/architecture** – the physical or technical constraints on activities (e.g. locks on doors or firewalls on the Internet)
- **Market** – economic forces
- **Law** – explicit mandates that can be enforced by the government
- **Norms** – social conventions that one often feels compelled to follow

b – 软件危机

b) Explain what is meant by the term *Software Crisis*. In your answer you should also discuss approaches that can be employed during software engineering projects to lessen the impact of the issues surrounding the software crisis.

[5 marks]

c – 敏捷Scrum方法论

c) During the courseworks for CSCM94/CSCM94J, you completed a group design and implementation of a piece of software named *'Cafe94'*. Based on your experience, propose a software methodology that could have been used during this process. You should provide a clear justification and reasoning for your choice.

[3 marks]

Q2 – 分析代码

Consider the following Java class that acts as a controller for a graphical user interface with an application:

```
public class GuessController implements Initializable {

  private final Random random = new Random();
  private int randomNumber;
  private int guessCount = 0;

  @FXML
  private TextField guess;
  @FXML
  private TextField result;

  @Override
  public void initialize(URL url, ResourceBundle rB) {
  randomNumber = random.nextInt(10);
  }

  @FXML
  void checkGuess(ActionEvent event) {
      if(Integer.parseInt(guess.getText()) == randomNumber){
      result.setText("Correct guess!");
      }
      else {
      guessCount++;
          result.setText("Try again. guesses: " + guessCount);
      }
  }
}
```

a – 解释代码

a) Describe what the above code implements.

[6 marks]

b – 提高代码阅读性

b) Discuss how the code could be changed in order to improve its readability.

[3 marks]

c – 代码错误❌

c) Discuss a potential error that could arise from the above code and provide a potential solution to the issue.

[6 marks]

Q3 – 画UML图

a – Class Diagram 类图

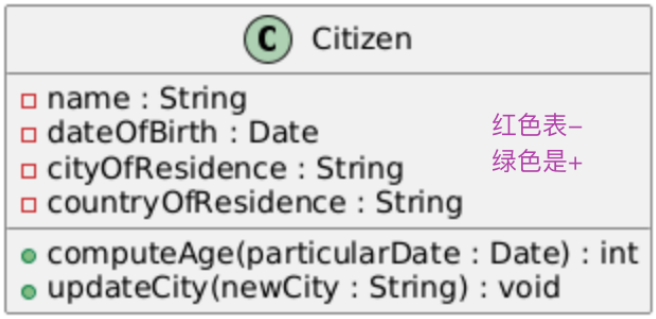
a) Consider the following class description:

The Citizen class holds information about a citizen's name, date of birth and city of residence, which are all required in the basic form of the class. Every instance of the Citizen class shares a piece of information called 'countryOfResidence'. The class provides functionality to compute a citizen's age based on a particular date and will return a whole number as a result of the function. The city of residence can be updated by another part of the system (such as another class) providing a new city is provided.

Draw a UML Class Diagram that provides a design for the class.

[7 marks]

1. **属性（Attributes）**
   - **name**：公民姓名（必需）。
   - **dateOfBirth**：公民出生日期（必需）。
   - **cityOfResidence**：公民所居住的城市（必需）。
   - **countryOfResidence**：公民所居住的国家（对于每个 Citizen 实例来说都需要保留该信息）。
2. **操作（Operations / Methods）**
   - **computeAge(particularDate: Date): int**
     根据传入的 particularDate 计算并返回公民的年龄。
   - **updateCity(newCity: String): void**
     将 cityOfResidence 更新为 newCity。题目说明此更新可能由系统中的其他部分触发。
3. **可选设计点**
   - 既然题目指出"每个实例都保存 countryOfResidence"，那么在类中直接定义此属性即可。
   - 对于日期类型，可以简单用 `Date` 表示，也可用其他更具体的类型（如 `LocalDate`）。
   - 在 UML 中通常将属性设置为私有（`-`），方法设置为公有（`+`）。当然，也可根据需求添加 getter/setter 方法。




```
@startuml
' 可以根据需要修改类名、属性、方法可见性与类型

class Citizen {
    - name : String
    - dateOfBirth : Date
    - cityOfResidence : String
    - countryOfResidence : String

    + computeAge(particularDate : Date) : int
    + updateCity(newCity : String) : void
}
@enduml
```

图形示例说明

1. `-` 表示私有属性（private）。
2. `+` 表示公有操作（public）。
3. 四个私有属性：`name`，`dateOfBirth`，`cityOfResidence`，`countryOfResidence`。
4. 两个公有方法：
   - `computeAge(...)`：根据特定日期计算返回年龄。
   - `updateCity(...)`：更新居住城市。

根据题目要求，**“每一个 Citizen 类的对象都有一块名为 countryOfResidence 的信息，并提供基于某个日期计算公民年龄的功能，以及可更新居住城市的功能”**，上述 UML 图已涵盖了这些需求。若在实现时需要更多的功能（如获取和修改国家的方法），可在类中继续补充相应的操作与属性访问器。

b – Use Case Diagram 用例图

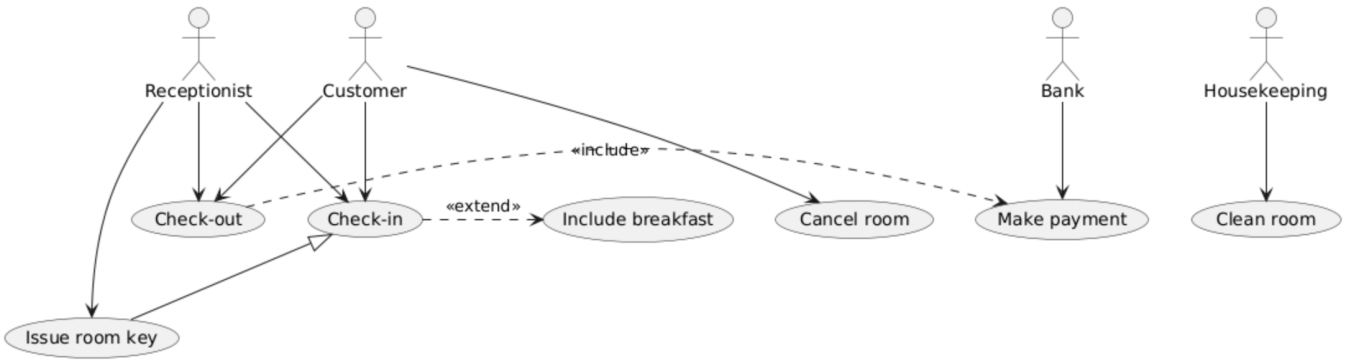
b) Consider the following scenario:

A client would like a hotel booking system to be designed and implemented. In the system customers should be allowed to check in and check out of a room, as well as cancelling a room. During the check in/out processes, a receptionist is also involved. The receptionist can also issue a room key for a room. When a customer checks in, they have the option of including a breakfast with then booking (though no compulsory). When a customer checks out, a payment must be made immediately that is processed through a bank service. A member of the housekeeping team performs cleaning services on rooms.

Draw a Use Case Diagram that represents a design of the system.

[12 marks]

1. **系统功能需求**：
   - 客户（Customer）可以执行以下操作：
     - 办理入住（Check-in）
     - 办理退房（Check-out）
     - 取消预订（Cancel room）
     - 选择包含早餐（Include breakfast，可选）
   - 接待员（Receptionist）参与以下操作：
     - 协助客户办理入住和退房
     - 发放房卡（Issue room key）
   - 退房时需通过银行服务（Bank）立即付款（Make payment）
   - 客房服务员（Housekeeping）负责房间清洁（Clean room）
2. **参与者（Actors）**：
   - Customer（客户）
   - Receptionist（接待员）
   - Housekeeping（客房服务员）
   - Bank（银行服务）
3. **用例（Use Cases）**：
   - Check-in（办理入住）
   - Check-out（办理退房）
   - Cancel room（取消预订）
   - Issue room key（发放房卡）
   - Include breakfast（包含早餐）
   - Make payment（付款）
   - Clean room（清洁房间）
4. **关系**：
   - 客户与入住、退房、取消预订、包含早餐直接相关。
   - 接待员与入住、退房、发放房卡相关。
   - 客房服务员与清洁房间相关。
   - 银行服务与付款相关。



改进点说明

1. **扩展关系（extend）**：
   - `Include breakfast` 是 `Check-in` 的可选扩展，用 `<<extend>>` 表示。
2. **包含关系（include）**：
   - `Make payment` 是 `Check-out` 的必需步骤，用 `<<include>>` 表示。
3. **泛化关系**：
   - `Issue room key` 是 `Check-in` 的一部分，用泛化关系（ `<|--` ）表示。

最终输出效果

改进后的图会更清晰地体现：

- 客户办理入住时可选早餐。
- 退房时必须付款。
- 房卡发放是入住的一部分。

CSCM94 2023

Q1 – 概念题

a – 莱斯格(Lessig)的四种规约

Question 1

a) The emergence of social media platforms has changed many aspects of societal life for many individuals globally..

Define each of Lessig's Four Modalities and for each modality give an example of how social media affects society, or how it has been affected by society..

[6 marks]

Four Modalities

- **Code/architecture** – the physical or technical constraints on activities (e.g. locks on doors or firewalls on the Internet)
- **Market** – economic forces
- **Law** – explicit mandates that can be enforced by the government
- **Norms** – social conventions that one often feels compelled to follow

b – 功能性与非功能性需求差异

b) Explain the differences between the functional and non-functional requirements. Provide an example of each requirement type.

[4 marks]

c – 敏捷Scrum方法论

c) Describe the Agile Scrum methodology. In your answer, you should discuss the stages and activities that take place.

[6 marks]

d – 重构（Refactoring）

d) Explain what you understand by the term Refactoring. Discuss when in a software engineering project this activity should take place and give two examples of a typical activity that happens during a refactoring process.

[4 marks]

Q2 – 分析代码

Consider the following Java class that acts as a controller for a graphical user interface with an application:

```
public class User {
  public String m;
  int printTimes = 1;

  /**
  * This method returns the name of the User
  * @return m - the name
  **/
  public void getName() {
  return m;
  }

  /**
  *  Takes the parameter newName and updates the m variable
  */
  public void setName(String newName) {
  m = newName;
  }

  public void SetPrintTime(int d) {
  this.printTimes = d;
  }

  public void print() {
        for(int i = 1; i < printTimes; i++) {
        System.out.println(m);
        }
  }
}
```

a – 代码规范

a) Consider the coding conventions from CSCM94. Discuss how this code does not comply with these rules and suggest improvements to address the issues.

[6 marks]

b – 代码错误

b) The User class is part of a wider system that allows other classes to make calls to the public methods. There is potential for bad data to be passed into one of these methods. Describe steps that could be taken within the user class and the calling class to handle such an error.

[3 marks]

Q3 – 画UML图

a – Use Case Diagram 用例图

a) Consider the following class description:

A client would like a hotel booking system to be designed and implemented. In the booking system, a customer can request a booking and a receptionist handles the request. A customer can also checkin and checkout of a room with a receptionist. During checkin, a customer can optionally request breakfast for each morning of their stay. The receptionist can also issue a room key for a room. When a customer checks out, a payment must be made immediately that is processed through a bank service.

Draw a Use Case Diagram that represents a design of the system.

[10 marks]

> 同2022年题目

b – Class Hierarchy 类结构层次

b) Consider the following class description:

A local shop wants to see an inventory system developed to keep track of stock it holds. The shop sells and buys CDs, Books, Magazines, Vinyls and Video Games

---

Identify candidate classes from the description box above and organise them into a class hierarchy. You should then draw out your hierarchy using standard UML conventions. You may be required to create classes that are not included in the description above.

If you need to write italic text, precede the text with a '/' symbol (e.g. /italicClass).

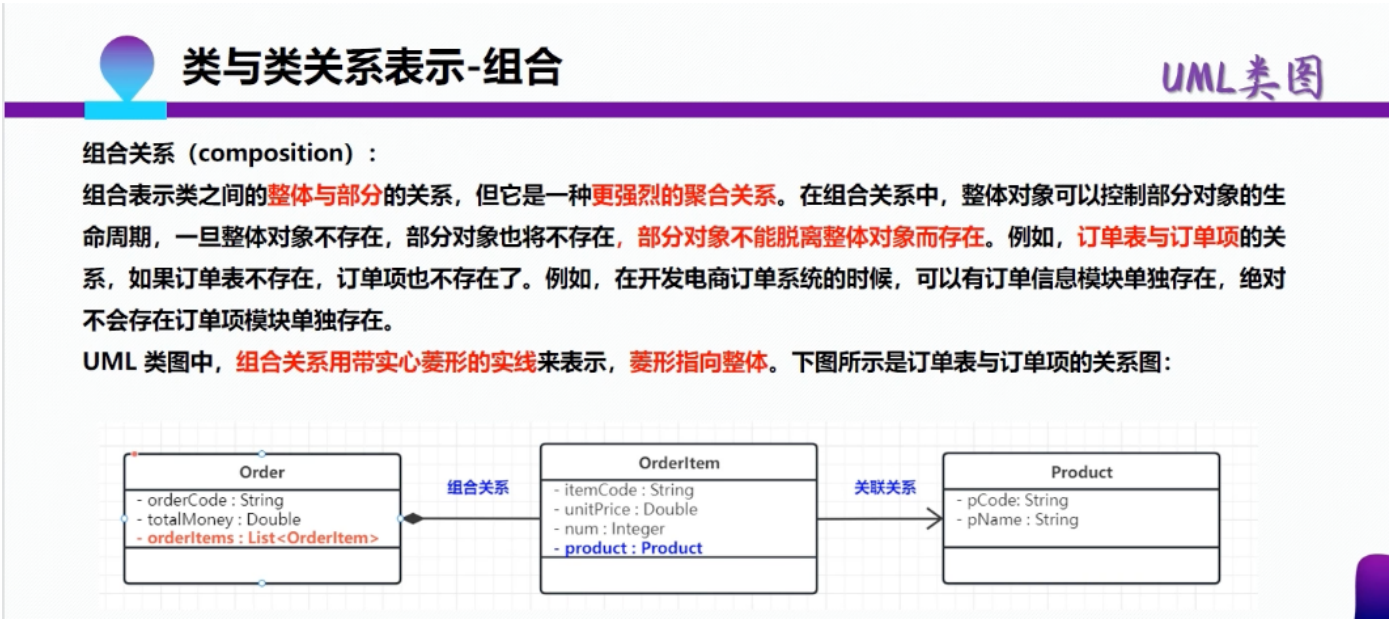
[7 marks]

1. **抽象超类** `Item`
   - 用途：将所有库存项目（CD、书籍、杂志、黑胶、游戏）共有的属性和行为提取到一个地方，避免重复定义。
   - 属性：
     - `id: int` —— 唯一标识每件库存物品。
     - `title: String` —— 标题或名称。
     - `price: double` —— 单价。
     - `quantity: int` —— 当前库存数量。
   - 操作（方法）：
     - `getDetails(): String` —— 返回物品的完整信息（如"ID、名称、价格、数量"等）。
     - `updateQuantity(delta: int)` —— 对库存数量进行增减操作，（例如进货时 `delta` 为正，销售时为负）。
   - UML 记法：
     - 使用 `abstract class Item` 表示这是个抽象类，不能直接实例化。
     - 在子类继承关系箭头上通常不画实心箭头，PlantUML 中用 `<|--` 表示泛化（Generalization）。
2. **具体子类：** `CD`、`Book`、`Magazine`、`Vinyl`、`VideoGame`
   - 都继承自 `Item`，自动获得超类中的属性和方法。
   - 可以在各自类中再定义专有属性 / 方法：
     - 例如 `Book` 可能有 `author: String`、`ISBN: String`；
     - `VideoGame` 可能有 `platform: String`、`releaseDate: Date`。
   - 继承关系的UML符号：

```lua
Item <|-- CD
Item <|-- Book
……
```

3. **管理类** `InventorySystem`

- 责任：对整个库存进行增删改查管理。
- 方法：
  - `addItem(item: Item)` —— 将一个新项目加入库存；
  - `removeItem(itemId: int)` —— 根据 ID 删除库存项；
  - `findItem(itemId: int): Item` —— 检索并返回指定 ID 的库存项；
  - `listAllItems(): List<Item>` —— 列出所有库存项。
- 属性（在图中未显式列出，但隐含有一个集合属性）
  - 例如内部可能维护 `items: List<Item>`。

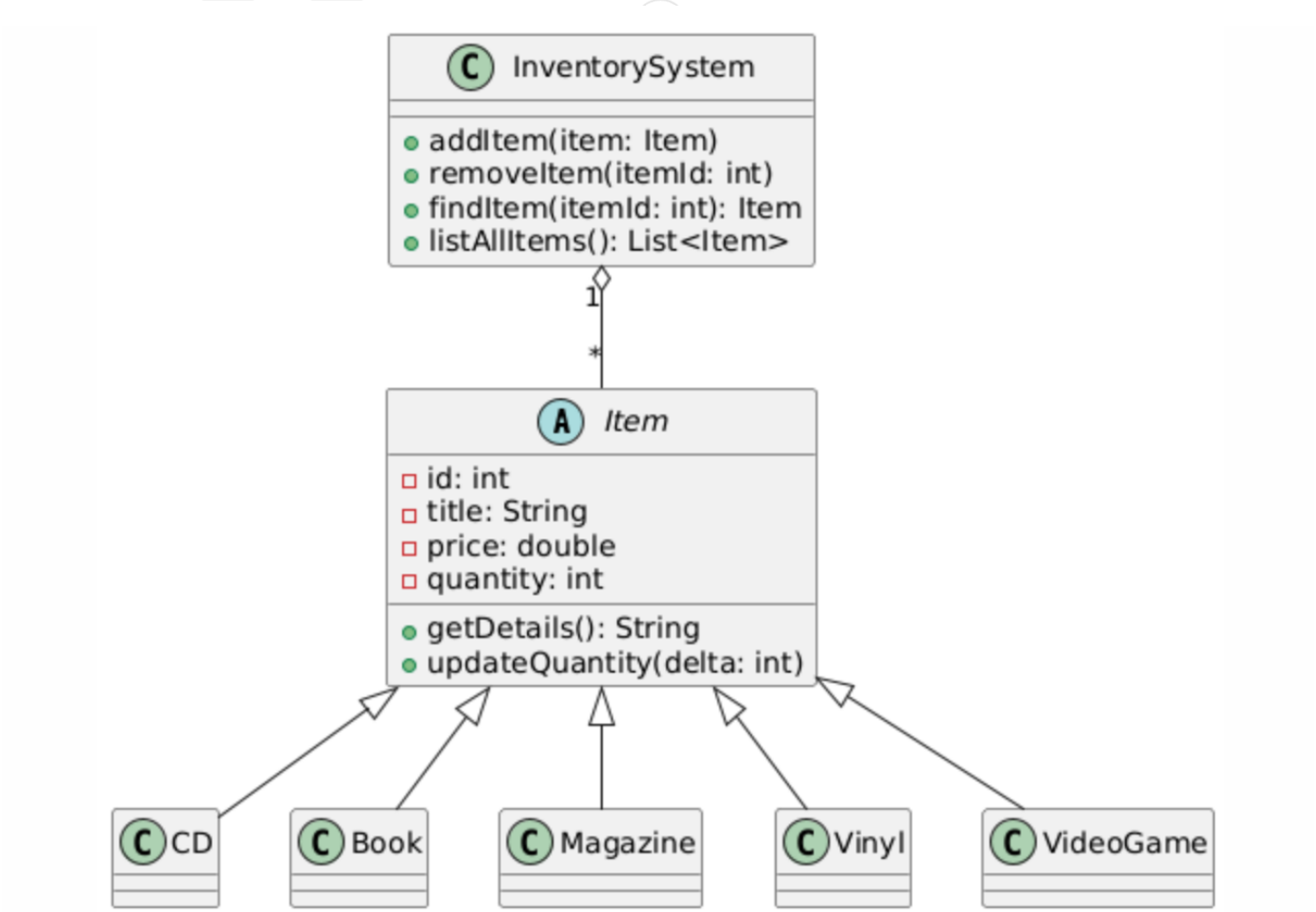


> UML类图设计–竟如此简单–1小时快速掌握
> https://www.bilibili.com/video/BV1H84y1j7HQ?spm_id_from=333.788.videopod.episodes&vd_source=7cf5ef44f3b13ee8494262e51e8fe7ba&p=8

4. **聚合 / 组合关系**
   - 在 PlantUML 中用：

```mathematica
InventorySystem "1" o-- "*" Item
```

   - 含义：
     - “1 o-- *” 表示一个 `InventorySystem` 对象包含（组合）多个 `Item` 对象。
     - 组合（filled diamond）表示 `Item` 的生命周期依赖于 `InventorySystem`，系统销毁则库存也被销毁。
5. **UML 符号小结**
   - **类** 用 `class Name { … }` 定义，抽象类前加 `abstract`。
   - **继承** 用空心三角箭头（PlantUML 的 `<|--`）连接子类指向父类。
   - **组合** 用黑色菱形（PlantUML 中 `o--`，也可用 `*--`）表示整体与部分的强关联，并用双引号标注多重性（`"1"` vs. `"*"`）。



c – (续)Class Hierarchy 类结构层次

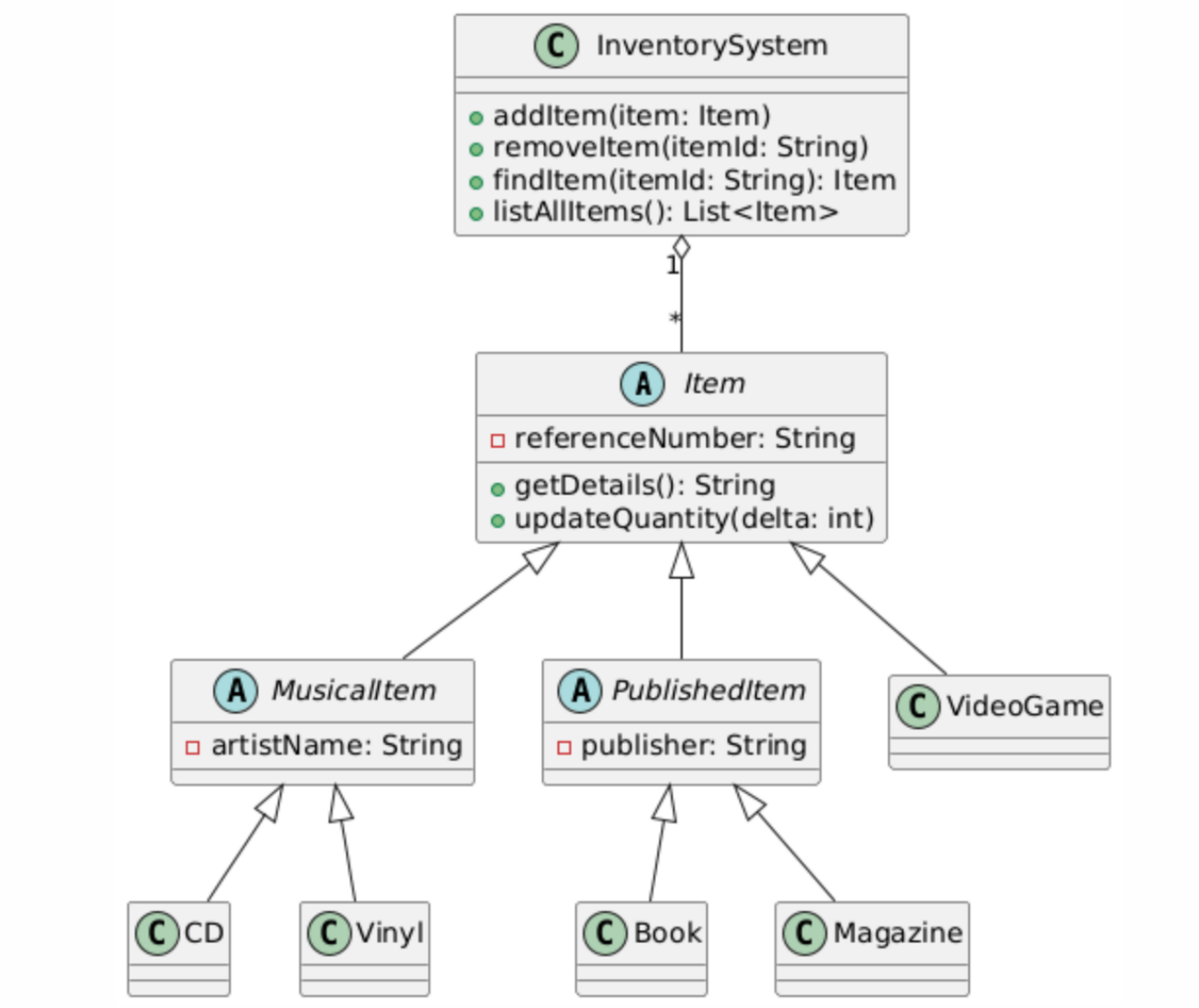
c) Consider your answer from 3b). The following responsibilities have been identified in the next stage of the design process:

- Store a unique reference number
- Store information about a music artists' name
- Store information about a publisher

Discuss a strategy for placing these responsibilities within your hierarchy and propose in which classes the newly identified responsibilities should be placed.

[4 marks]

1. 在抽象根类 `Item` 中放置通用责任——`referenceNumber`。
2. 通过中间抽象类 `MusicalItem` 和 `PublishedItem` 分别承载 `artistName` 与 `publisher`，并让具体的 `CD` / `Vinyl`、`Book` / `Magazine` 继承自它们，以实现职责的高内聚与重用。
3. 保持 `VideoGame` 等其他独立类别直接继承自根类。
4. 使用 `InventorySystem` 聚合所有 `Item`，管理增删查改操作。



CSCM94 2024

Q1 – 概念题

a – 敏捷模型和瀑布模型

(a) Agile and waterfall models are two categories of models for software engineering. Describe in 2 short sentences the primary differences between the two. **[2 marks]**

b – 功能需求与非功能需求

(b) Provide an example of a functional and of a non-functional requirement. Explain the difference between the two through these examples. **[3 mark**

c – 低保真与高保真原型

(c) Describe how Low Fidelity and High Fidelity prototypes can be used during a software engineering process. In your answer you should discuss when in the software engineering process they should be used. **[3 marks]**

Q2 – 画UML图

用例图 use case diagram

Consider the Amazwan new book and DVD online store that services South Wales and the Valleys! Its online store shall follow the following specification:

There are no accounts for Amazwan or logins. The clients of Amazwan are the most common type of persona for the system. They can search for products. They can also select products from the current screen and put them into their shopping cart. They can also pay for their items through the checkout functionality which spawns a process for them to enter their credit card number.

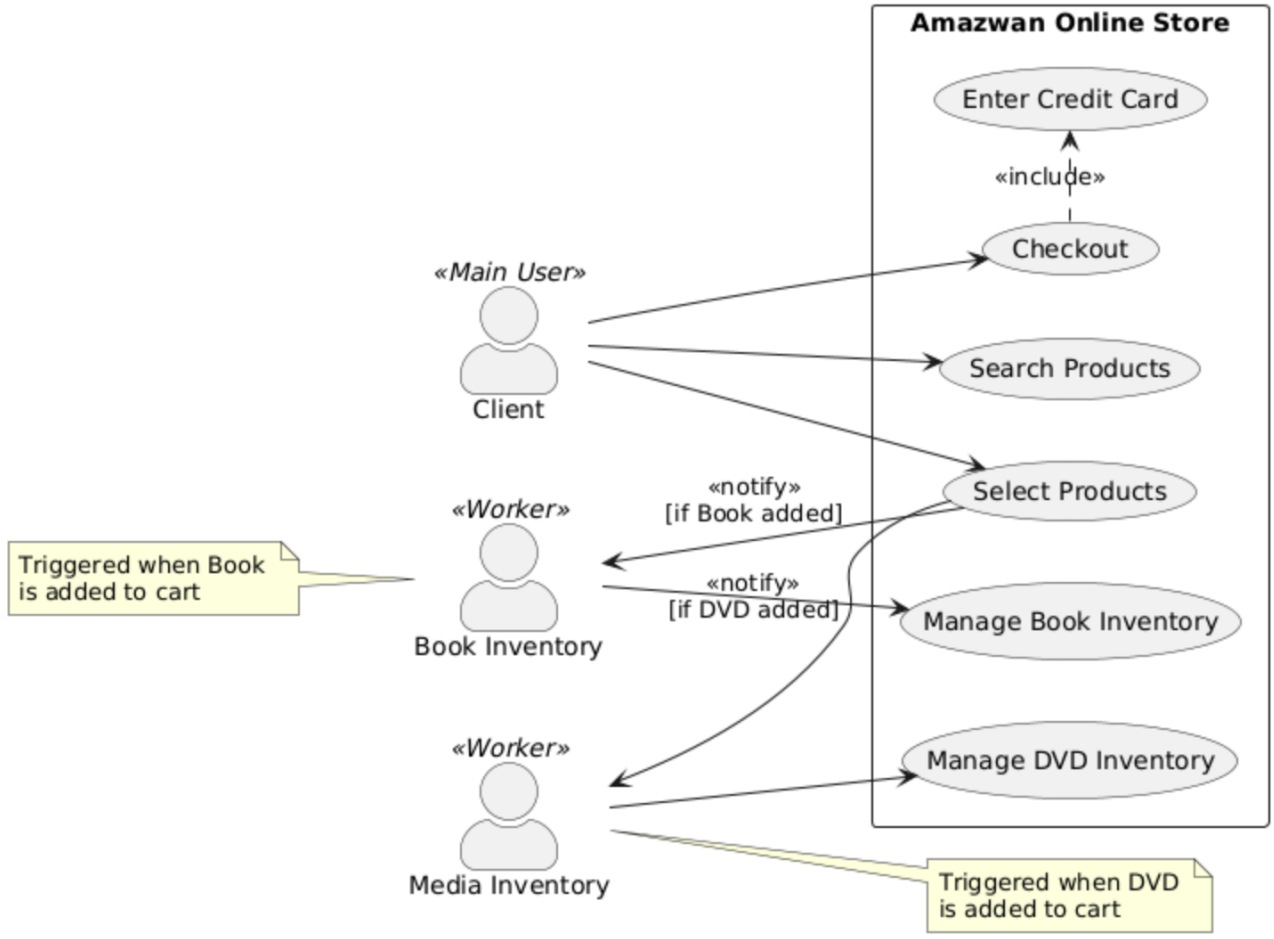
If the customer places a DVD into their shopping cart (a special case of putting an item), the Media Inventory persona is notified. A Media Inventory persona can also perform inventory on all of Amazwan's DVDs.

If the customer places a book into their shopping cart, the Book Inventory persona is notified. A Book Inventory persona can perform an inventory on all their books.

Draw a use case diagram that depicts the above scenario. **[8 marks]**

一、系统要素分析

1. **参与者（Actors）**：
   - `Client` （客户）：主要用户，无需登录
   - `Media Inventory` （媒体库存人员）：管理DVD库存
   - `Book Inventory` （书籍库存人员）：管理书籍库存
2. **核心用例**：
   - `Search Products` （搜索商品）
   - `Select Products` （选择商品）
   - `Checkout` （结账）包含 `Enter Credit Card` （输入信用卡）
   - `Manage DVD Inventory` （管理DVD库存）
   - `Manage Book Inventory` （管理书籍库存）
3. **特殊触发关系**：
   - 放入DVD到购物车时通知 `Media Inventory`
   - 放入书籍到购物车时通知 `Book Inventory`



三、设计要点说明

1. **关系处理**：
   - 使用 `<<notify>>` 扩展关系表示购物车操作触发通知
   - 结账与信用卡输入采用 `<<include>>` 包含关系
2. **布局优化**：
   - 主用户（Client）置于左侧
   - 工作人员角色分列两侧
   - 使用 `left to right direction` 改善可读性
3. **标记增强**：
   - 添加注释说明触发条件
   - 使用 `<<Worker>>` 标记工作人员角色类型
   - 通过 `actorStyle awesome` 优化图标显示
4. **边界明确定义**：
   - 用 `rectangle` 明确系统边界
   - 所有用例包含在系统范围内

Q3 – 画UML图

Class Collaborations

Consider the new mobile phone media service Msg94, which allows users to send and receive text, videos, and sound messages. Its specification is as follows:

Within the Msg94 application a traditional text message can be displayed and media forms of messaging can be both displayed and played. All messages can display their content and share a variable called 'messageCount' that is used to generate a unique ID for each message.

Text messages are classes that simply display the text of the message entered by the user. The class stores the text of the message and provides methods to set and retrieve this text.

Sound messages can both record audio provided by a user and also play audio messages that a user receives. Sound messages are recorded from a device microphone and stored in an array of integers called samples. The sound messages also have controls to lower and increase the volume of the message, with the volume being stored also as an integer to represent the volume level.

Video messages are recorded using a device camera and stored as an array of strings called videoStream. Like Sound messages, video can have volume levels increased and decreased. The video message class also allows for the rewind and fast-forwarding of content to be controlled, by skipping in increments of 5 seconds.

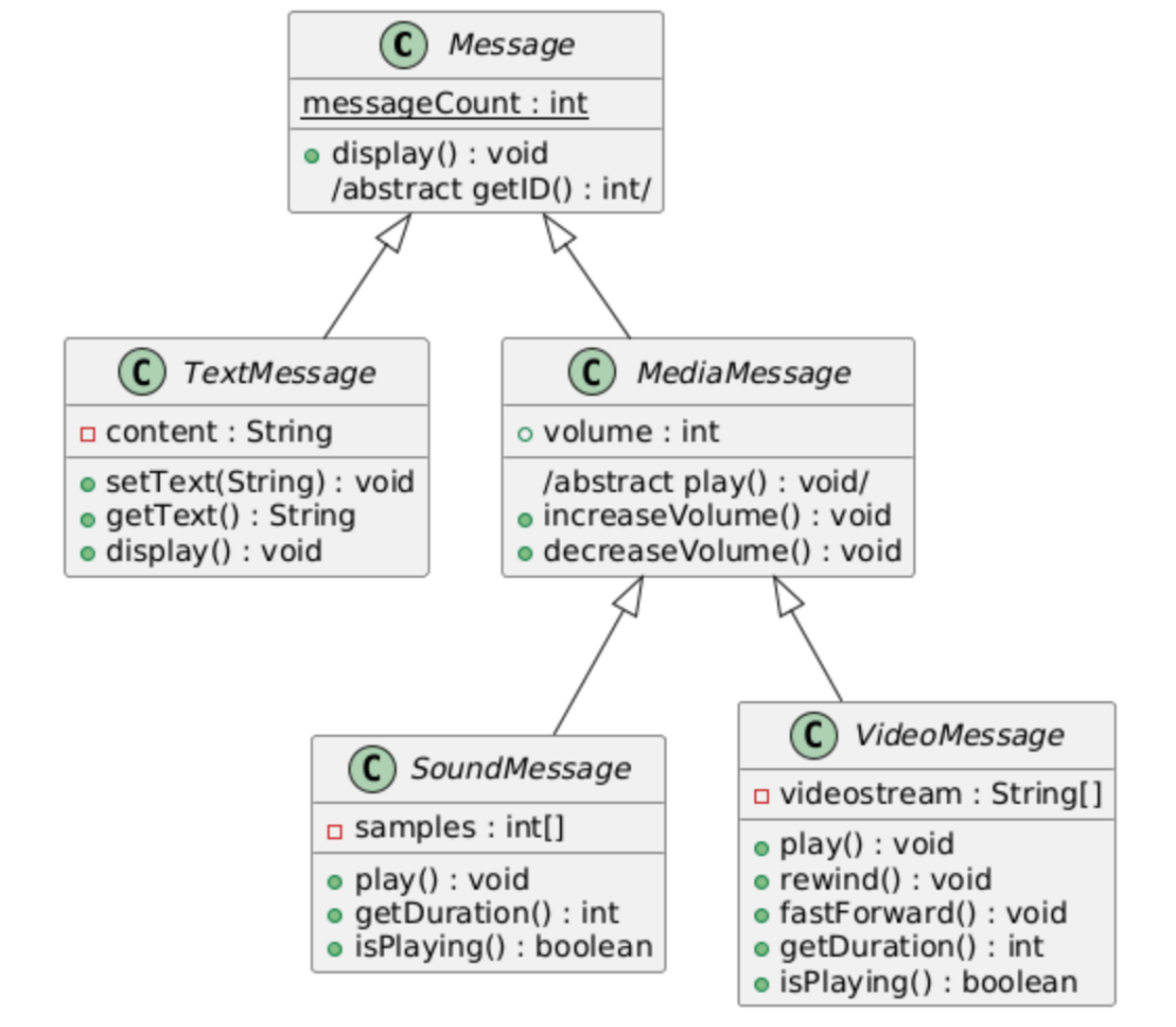
Video messages and Sound messages should be able to report on whether they are currently playing their audio or video content and provide information on the length of a recorded clip.

---

a – 类层次结构设计

(a) Specify a good class hierarchy for all classes in the above description.
If you need to write *italic* text, precede the text with a '/' symbol (e.g. /classname).
[3 marks]

1. **基类设计**：
   - `Message` 为抽象基类，包含：
     ✓ 静态变量 `messageCount` （实现唯一ID生成）
     ✓ 抽象方法 `getID()`
     ✓ 公共方法 `display()`
2. **继承关系**：
   - `TextMessage` 直接继承 `Message` ，实现纯文本功能
   - `MediaMessage` 作为抽象中间层，处理音视频共性：
     ✓ 音量控制方法
     ✓ 抽象播放方法
3. **具体媒体类**：
   - `SoundMessage` ：
     ✓ 用 `int[]` 存储音频采样
     ✓ 实现播放状态报告
   - `VideoMessage` ：
     ✓ 用 `String[]` 存储视频流
     ✓ 扩展播放控制（快进/快退）
4. **斜体表示规范**：
   - 抽象类和抽象方法使用 `/斜体/`
   - 示例： `/abstract play()/`

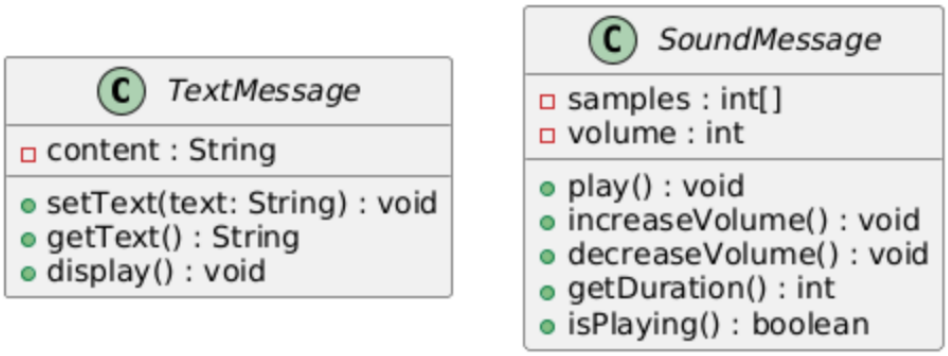


b – UML类图

(b) For text message and sound message only, draw full and correct UML class diagrams specifying all methods and attributes of these classes.

If you need to write *italic* text, precede the text with a '/' symbol (e.g. /classname).

[8 marks]



关键要素说明：

1. **文本消息类**：
   - 私有属性：`-content : String`（存储短信内容）
   - 公共方法：
     - `setText()` / `getText()`（内容存取）
     - `display()`（显示功能）
2. **声音消息类**：
   - 私有属性：
     - `-samples : int[]`（音频采样数组）
     - `-volume : int`（当前音量值）
   - 公共方法：
     - 核心功能：`play()`
     - 音量控制：`increaseVolume()` / `decreaseVolume()`
     - 状态查询：`isPlaying()` / `getDuration()`
3. **格式规范**：
   - 所有斜体文本（如抽象类/方法）需用 `/` 包裹
   - 示例：若需定义抽象类应为 `/abstract Message/`

Q4 – 分析代码

Coding Conventions

Consider the following program:

```
import java.util.Scanner;

public class Readout {

    public static void main(String[] args) {
        Scanner myScanner = new Scanner(System.in);
        System.out.println("Please enter a number:");
        int myNumber = myScanner.nextInt();
        System.out.println("Please enter a character");
        char myChar = myScanner.next().charAt(0);
        myScanner.close();

        for (int i = 0; i < myNumber; i++) {
            for (int j = 0; j <= myNumber; j++) {
                System.out.print(myChar);
            }
            System.out.println();
        }
    }
}
```

a – 解释代码

(a) Describe what this code implements. **[3 marks]**

b – 代码错误❌

(b) Discuss any potential errors that exist in the code. **[3 marks]**

c – 修复代码错误

(c) Rewrite the code to fix any errors in the code provided and to handle any errors that a user may introduce into the system, while providing meaningful information back to the user. **[5 marks]**

d – 代码注释

(d) Add appropriate documenting comments to the code provided. **[2 marks]**

Q5 – 代码测试

Software Implementation

Consider the following code sample:

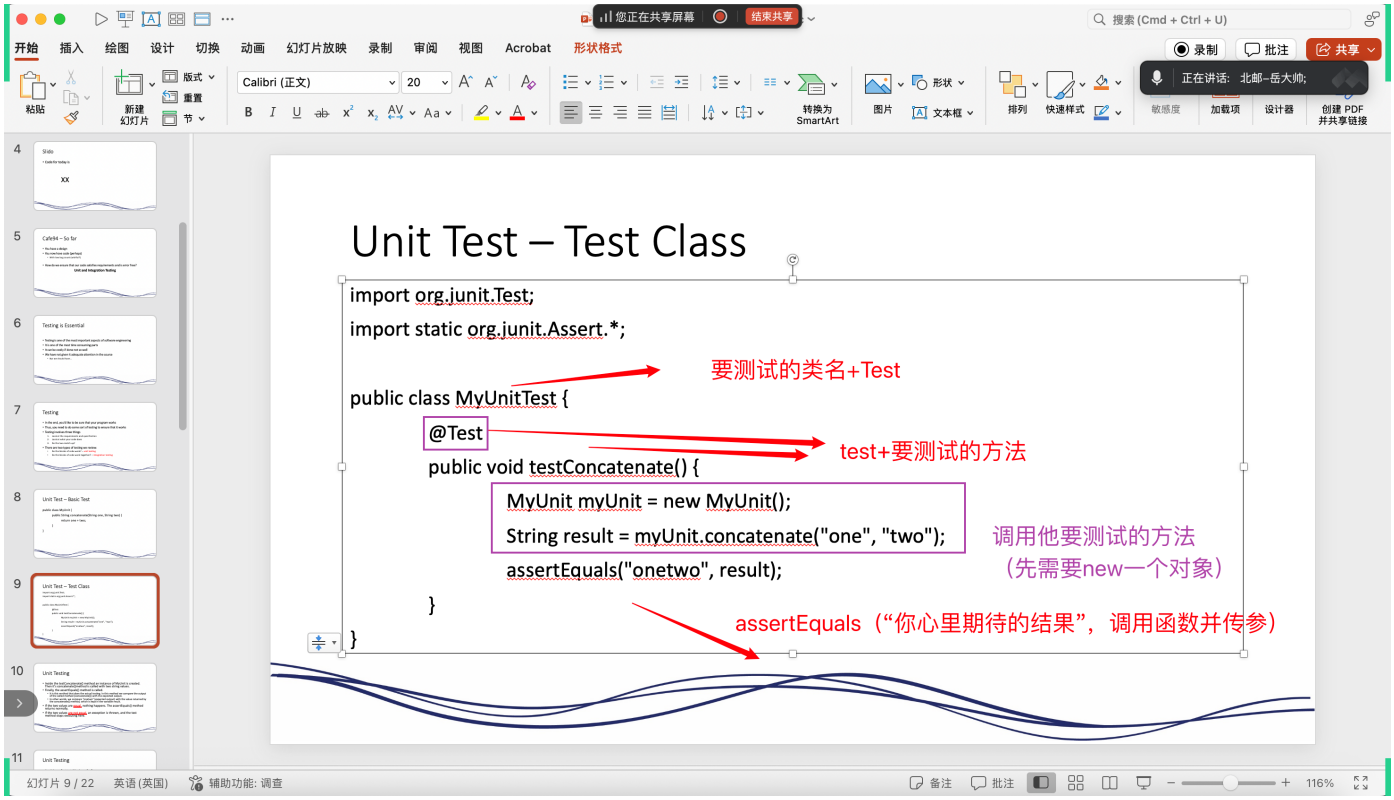
```
public class MyCalculator {
    public int add(int one, int two) {
        return one + two;
    }

    public double divide(double one, double two) {
        return two / one;
    }

    public int fancySum(int one, int two, int
       three){
        return (one + two) * three;
    }
}
```

a – 测试方法

(a) Write code to test each method to ensure they output expected results.
[7 marks]

Unit Test – Basic Test

```
public class MyUnit {
        public String concatenate(String one, String two) {
                return one + two;
        }
}
```



b – 大型程序测试策略

(b) Discuss a testing strategy that could be used if the code above were part of a larger program with several collaborations between classes.

[3 marks]

先单元测试 然后 集成测试