
物件導向程式設計 期末專題

第16組

組員：楊騰闊A113223008、楊少荃A113223009

班級：智網二甲

【系統概述】

一個簡單的訂餐管理系統，用來管理餐廳菜單、顧客訂單和訂單狀態。使用者可以瀏覽菜單、加入購物車、下訂單並列印明細。

【類別規格】

類別:Dish(菜品)

屬性:

- dishId(字串): 菜品的唯一識別碼, 不可為空或null
- name(字串): 菜品名稱, 不可為空或null
- price(浮點數): 菜品價格, 必須大於0
- isAvailable(布林): 菜品是否可訂購, 預設為true

行為:

- 建構子 Dish(dishId, name, price)
 - 若 dishId 或 name 為 null 或空字串, 拋出 IllegalArgumentException 訊息: "[參數名稱] cannot be null or empty"
 - 若 price <= 0, 拋出 IllegalArgumentException 訊息: "Price must be greater than 0"
- markAsUnavailable(): 將菜品標記為不可訂購
 - 若菜品已不可訂購, 拋出 IllegalStateException 訊息: "Dish is already unavailable"
- markAsAvailable(): 將菜品標記為可訂購
 - 若菜品已可訂購, 拋出 IllegalStateException 訊息: "Dish is already available"
- getDishId() / getName() / getPrice() / isAvailable(): 取得屬性值

類別: Order(訂單)

屬性:

- orderId(字串): 訂單唯一識別碼, 不可為空或 null
- dishes(ArrayList<Dish>): 訂單菜品列表
- status(字串): 訂單狀態, 預設為 "未處理"

行為:

- 建構子 Order(orderId, dishes)
 - 若任何參數為 null, 拋出 IllegalArgumentException
 - 訂單內容須全為可訂購菜品
- calculateTotal(): 計算訂單總價
- addDish(Dish dish): 加入菜品到訂單
 - 若菜品不可訂購, 拋出 IllegalArgumentException 訊息: "Dish is unavailable"
- removeDish(Dish dish): 從訂單中移除菜品
 - 若菜品不在訂單中, 拋出 IllegalArgumentException 訊息: "Dish is not in the order"
- markAsDelivered(): 將訂單標記為已送達
 - 若訂單已送達, 拋出 IllegalStateException 訊息: "Order cannot be marked as delivered"

類別: Restaurant(餐廳)

屬性:

- menu(ArrayList<Dish>): 餐廳菜單
- allOrders(ArrayList<Order>): 所有訂單紀錄

行為:

- addDishToMenu(Dish dish): 新增菜品
 - 若菜品 ID 已存在, 拋出 IllegalArgumentException 訊息: "Dish with this ID already exists"
- findDishById(String dishId): 以 ID 查詢菜品
 - 找到回傳 Dish 物件
 - 找不到回傳 null
- findDishesByKeyword(String keyword): 關鍵字搜尋菜品
 - 搜尋菜名包含關鍵字(不分大小寫)
 - 回傳符合條件的 ArrayList<Dish>
 - 無結果時回傳空的 ArrayList
- getAvailableDishes(): 取得所有可訂購菜品
 - 回傳 isAvailable 為 true 的菜品清單
- placeOrder(Order order): 接受訂單
 - 成功加入 allOrders

類別 : Main(系統操作入口)

行為 :

- 提供互動式選單：
 1. 瀏覽/搜尋菜單
 2. 加入購物車
 3. 結帳並列印明細
 4. 退出系統
- 使用者可輸入菜品 ID 加入購物車
- 結帳時列印訂單明細與總價
- 下訂單後購物車清空

【功能總結】

1. 使用者可以瀏覽菜單並搜尋菜品。
2. 使用者可以將菜品加入購物車並結帳。
3. 訂單包含菜品清單、總價和狀態(未處理 / 已送達)。
4. 餐廳可以管理菜單，新增菜品並防止重複 ID。
5. 所有訂單會被餐廳紀錄。

=====

訂餐管理系統 - 測試案例清單

版本: 1.0

=====

【Dish 類別測試】

TC-D001: 正常建立菜品

輸入: dishId="D001", name="炒飯", price=120.0

預期: 物件建立成功, isAvailable為true

TC-D002: dishId為null時拋出例外

輸入: dishId=null, name="炒飯", price=120.0

預期: 拋出IllegalArgumentException, 訊息為"dishId cannot be null or empty"

TC-D003: name為空字串時拋出例外

輸入: dishId="D001", name="", price=120.0

預期: 招出IllegalArgumentException, 訊息為"name cannot be null or empty"

TC-D004: price為0或負數時拋出例外

輸入: dishId="D001", name="炒飯", price=-10.0

預期: 招出IllegalArgumentException, 訊息為"Price must be greater than 0"

TC-D005: 正常標記菜品為不可訂購

前置: 建立可訂購的菜品

動作: 呼叫markAsUnavailable()

預期: isAvailable變為false

TC-D006: 嘗試重複標記已不可訂購的菜品

前置: 菜品已被標記為不可訂購

動作: 再次呼叫markAsUnavailable()

預期: 招出IllegalStateException, 訊息為"Dish is already unavailable"

TC-D007: 正常標記菜品為可訂購

前置: 菜品已被標記為不可訂購

動作: 呼叫markAsAvailable()

預期: isAvailable變為true

TC-D008: 嘗試重複標記已可訂購的菜品

前置: 菜品已被標記為可訂購

動作: 再次呼叫markAsAvailable()

預期: 招出IllegalStateException, 訊息為"Dish is already available"

【Order 類別測試】

TC-O001: 正常創建訂單

輸入: orderId="O001", dishes=[Dish 物件]

預期: 訂單創建成功, 並計算正確的總價

TC-O002: 添加不可訂購菜品至訂單

前置: 訂單已創建

動作: 嘗試加入不可訂購的菜品

預期: 抛出IllegalArgumentException, 訊息為"Dish is unavailable"

TC-O003: 移除未在訂單中的菜品

前置: 訂單已創建並包含菜品

動作: 嘗試移除不在訂單中的菜品

預期: 抛出IllegalArgumentException, 訊息為"Dish is not in the order"

TC-O004: 正常標記訂單為已送達

前置: 訂單狀態為"未處理"

動作: 呼叫markAsDelivered()

預期: 訂單狀態變為"已送達"

TC-O005: 嘗試重複標記已送達的訂單

前置: 訂單狀態為"已送達"

動作: 再次呼叫markAsDelivered()

預期: 抛出IllegalStateException, 訊息為"Order cannot be marked as delivered"

【Restaurant 類別測試】

TC-R001: 正常新增菜品至菜單

輸入: dishId="D001", name="炒飯", price=120.0

預期: 菜品成功新增至菜單

TC-R002: 新增重複菜品ID

前置: 菜單中已有D為"D001"的菜品

動作: 嘗試新增D為"D001"的菜品

預期: 扔出IllegalArgumentException, 訊息為"Dish with this ID already exists"

TC-R003: 查詢菜品ID

前置: 菜單中有菜品D為"D001"

動作: 呼叫findDishById("D001")

預期: 回傳該菜品物件

TC-R004: 查詢無結果的菜品D

動作: 呼叫findDishById("D999")

預期: 回傳null

TC-R005: 查詢所有可訂購菜品

前置: 菜單中有可訂購與不可訂購的菜品

動作: 呼叫getAvailableDishes()

預期: 回傳所有sAvailable為true的菜品

TC-R006: 顧客成功下訂單

前置: 菜單中有可訂購的菜品

動作: 呼叫placeOrder(order)

預期: 訂單成功新增至餐廳訂單清單

TC-R007: 關鍵字搜尋菜品(不分大小寫)

前置: 菜單中有"炒飯"與"牛肉麵"

動作: 呼叫findDishesByKeyword("FAN")

預期: 回傳包含"炒飯"的菜品清單

TC-R008: 關鍵字搜尋無結果

前置: 菜單中有菜品

動作: 呼叫findDishesByKeyword("不存在的菜名")

預期: 回傳空的ArrayList

yk57swMZX 帶我分析網址內容，
並幫我做出一個線上點餐系統

顯示思路 ▾

該網址

<https://hackmd.io/@chuckhu/Hyk57swMZX>
的內容主要是一個關於**「線上點餐系統」的專案開發筆記與需求規格書**。

根據該文件的內容，我為您分析了其核心功能架構，並為您準備了一套可以運行的簡易版線上點餐系統程式碼範例（使用 HTML/JS），您可以直接以此為基礎進行擴充。

幫我生成一個線上點餐系統，用
java 做，並且幫我做一個系統架
構表

這是一個根據您先前提供的規格風格，使用
Java 實作的線上點餐系統核心邏輯。我採用了
物件導向（OOP）的設計模式，將系統拆分為商
品、購物車、訂單管理三個核心區塊。

1. 系統架構表 (System Architecture)

在 Java 實作中，我們將系統分為 **模型層**
(Model)、**服務層 (Service)** 與 **資料儲存層**

(Data Repository)

層級	組件	職責說明
Model	Product , OrderItem , Order	定義基礎資料 結構（餐點、 訂單詳情、訂 單狀態）。
Service	OrderingSystem	處理核心業務 邏輯（下單驗 證、計算總金 額、變更訂單 狀態）。
Data	ArrayList (模擬 資料庫)	暫存菜單與訂 單資料。

動作 : findBooksByKeyword("java")

預期 : 回傳包含"Java Programming"的清單

TC-L002: 關鍵字搜尋無結果

動作 : findBooksByKeyword("C++")

預期 : 回傳空的ArrayList

TC-L003: 新增重複ISBN的書籍

前置 : 已有ISBN為"978-1"的書

動作 : 新增另一本ISBN為"978-1"的書

預期 : 拋出IllegalArgumentException

幫我寫出測試案例 上面是範例。這些測試直接對應規格書中的每一條規則。



以下是根據你提供的規格書，對 訂餐管理系統 各類別的測試案例。我會依照你要求的格式寫出每一個測試案例，並確保涵蓋系統的各項功能。

訂餐管理系統 - 測試案例清單

版本 : 1.0

【Dish 類別測試】

TC-D001: 正常建立菜品

輸入 : dishId="D001", name="炒飯", price=120.0

預期 : 物件建立成功, isAvailable為true

TC-D002: dishId為null時拋出例外

輸入 : dishId=null, name="炒飯", price=120.0

預期 : 拋出IllegalArgumentException，訊息為"dishId cannot be null or empty"

TC-D003: name為空字串時拋出例外

輸入 : dishId="D001", name="", price=120.0

預期 : 拋出IllegalArgumentException，訊息為"name cannot be null or empty"

TC-D004: price為0或負數時拋出例外

屬性 :

- name (字串) : 圖書館名稱
- books (ArrayList<Book>) : 館藏書籍
- members (ArrayList<Member>) : 註冊會員

行為 :

- addBook(Book book) : 新增書籍
 - 若 ISBN 已存在，拋出 IllegalArgumentException
 - 訊息 : "Book with this ISBN already exists"

- addMember(Member member) : 新增會員
 - 若會員編號已存在，拋出 IllegalArgumentException
 - 訊息 : "Member with this ID already exists"

- findBookByIsbn(String isbn) : 以ISBN搜尋書籍
 - 找到回傳 Book 物件
 - 找不到回傳 null

- findBooksByKeyword(String keyword) : 關鍵字搜尋
 - 搜尋書名或作者包含關鍵字的書籍（不分大小寫）
 - 回傳符合條件的 ArrayList<Book>
 - 無結果時回傳空的 ArrayList

- getAvailableBooks() : 取得所有可借閱書籍
 - 回傳 isAvailable 為 true 的書籍清單

按照這個格式生成一份訂餐管理系統的系統規劃書



這裡是根據你的要求，為 訂餐管理系統 設計的一份 功能規格書，遵循了圖書館管理系統的格式。

訂餐管理系統 - 功能規格書

版本 : 1.0

【系統概述】



再根據測試案例清單與上述規劃書，做出一個點餐系統



這三個類別都不能在IntelliJ IDEA 執行，我要在哪裡執行

顯示思路 ▾



根據您提供的「測試案例清單」與先前的規劃書，我為您開發了這套完整的 **Java 訂餐管理系統**。此版本嚴格遵循測試案例中的異常處理 (Exception Handling)、屬性驗證及業務邏輯 (如菜品狀態管理與關鍵字搜尋)。

訂餐管理系統 (Ordering Management System) 系統架構

由左向右滑動可切換內容

如果您在 IntelliJ IDEA 中無法執行這三個類別，通常是因為缺少了程式的進入點 `main` 方法，或者是類別分別存在於不同的檔案但沒有被正確組織在同一個專案 (Project) 架構下。

Java 程式需要一個 `public static void main(String[] args)` 作為引擎的起點才能運作。

1. 如何在 IntelliJ IDEA 執行？

我想在執行最後有一個明
細表，列出我下單的列表
與總金額



:

這是一個非常實用的功能！為了在執行最後產生一份清晰的**結帳明細表**，我們需要在「確認下單（選項 3）」的邏輯中，將購物車內的菜品逐一列出。

以下是修正後的 `Main.java`。我優化了明細表的視覺呈現，並根據 PDF 測試案例清單（例如 **TC-O001 驗證總價**）重新標註了精確的 `cite` 註解。

7. 未來擴充功能

- **會員系統**：累積點數、紀錄喜好餐點。
- **線上支付**：整合 LINE Pay、街口支付或綠界金流。
- **列印功能**：串接熱感應印表機，自動列印廚房單。

