第五章　物件變數與物件陣列

5-1 真實環境程式化

5-1-1 真實現象數位化

建構資訊系統主要目的，是要將真實環境可能出現的各種現象數位化，並能儲存於資訊系統內，再依照運作環境所需而編寫處理資料之程式，使其能自動化執行，這就是目前最流行的『電子化系統』，如電子化企業(E-enterprise)、電子化商務(E-commercer)...等。

由此可見，建構電子化系統，最基本的工作即是將所欲處理的『事務』數位化，才可利用資訊系統處理。但這裡所說的『事務』並不一定是有形的真實事務，也有可能虛擬幻象的『抽象』現象。簡單的說，吾人必須想盡辦法將任何可能出現的『人』、『事』、『地』、『物』，且無論是有形或無形的，都要將其數據化，使其可放入資訊系統中處理。

如何將真實環境數位化，基本方法是將該『事物』(或實體，Entity) 可能出現的現象，收集成若干個屬性 (attribute)。每一種屬性利用一個變數 (某種資料型態) 儲存該屬性的數值，這些屬性的集合，則是該『事物』現象的數位資料。譬如，如欲描述真實環境中的『汽車』現象，也許會選用廠牌、價錢、車型、排氣量、馬力、門數...等屬性，每一種屬性由一個變數儲存其數值或資料，有了這些數位化資料的格式後，則可將許多類型汽車依照其屬性儲存於資訊系統。如果有很多類型汽車，每種汽車都選用相同屬性但屬性內的資料不同，吾人就可以利用這些資訊來作各廠牌之間汽車效能比較、搜尋所喜歡車種...等等電子化處理程序。

另一方面來講，描述『事物』的屬性越多的話，描述得越仔細，雖然可得到的資訊越正確，但資訊系統則需要更多的資源來處理；如果『屬性』越少的話，系統處理速度可能較快，但可能會不能滿現況所需，或出現處理不周的現象。到底應該如何描述才是最理想，這就考驗程式設計師的能耐了。但最起碼可區分下列兩大重點：

(A) 同一『現象』不同描述：

當某一種真實『現象』需要不同的處理時，可能會出現不同的描述，每一種描述則需要各種不同的屬性。譬如，真實環境中的『人』就可能出現許多描述的類型，如圖 5-1 所示。建構醫院管理系統時，描述每一個『人』的屬性可能包含姓名、身分證號、病歷代號、地址、電話；而建構銀行信用卡系統時，『人』的信用卡資料可能包含姓名、會員代號、地址、電話、信用等級、所屬銀行；而銀行開戶系統中，銀行可能會需要登錄『人』的姓名、帳戶代號、地址、電話、存款等等。由此可見，同樣的『事物』進入不同系統處理時，所需描述的屬性可能會相差很大。而且就算是相同功能的資訊系統但由不同程式設計師製作時，因個人考量不同，要選擇哪些屬性來描述『現象』也不盡相同，何況是不同應用系統呢？

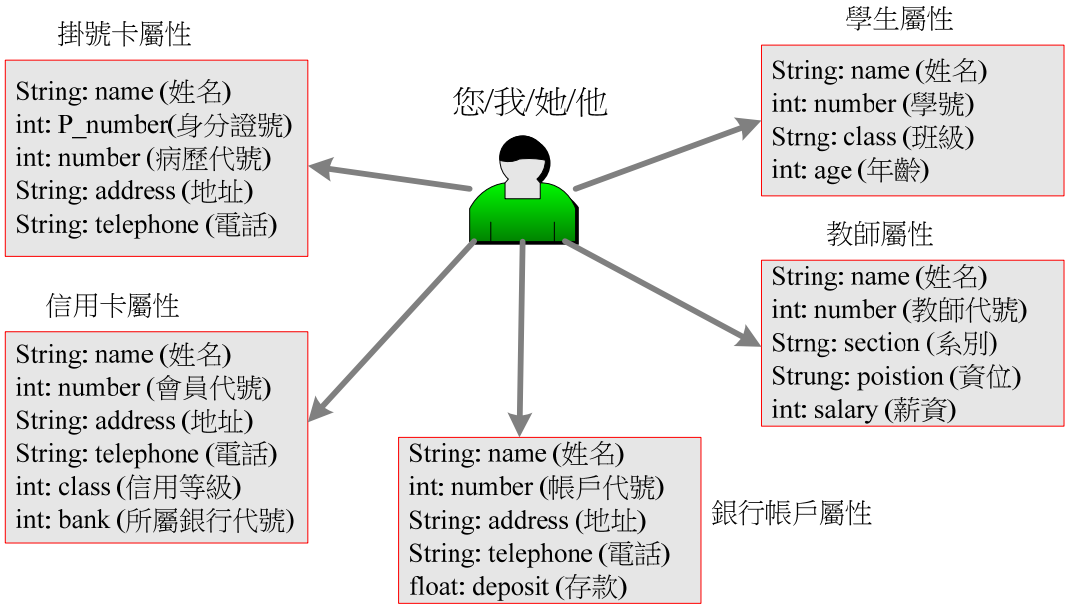

圖 5-1 同一現象不同描述

（B）相同描述，辨識不同個體：

選定某些『屬性』來描述特定『事物』之後，又如何來辨識相同事物現象之間的不同個體呢？譬如，系統如何在眾多信用卡之中，辨識哪一張是『春嬌』、哪一張是『志明』？簡單的方法是選取某一個能夠辨識不同個體的屬性，如果是『人』的話，可以選用『身分證字號』作為屬性其中之一，因為幾乎不可能出現相同身分證字號的『人』，如果找不到現有可辨識個體的屬性，則可能要自行創立。以圖 5-2 圖書館系統為例，描述書籍則非常有可能出現相同的出版社、書名、作者，為了要能辨識個體，只好加入一個『編號』屬性。基本上，該『編號』屬性與所描述書籍並沒直接關聯，僅為了分辨不同書籍而已，因此，他的內容需由管理員自行編制，但需保證不可以重複。

圖 5-2 相同描述辨識不同個體

5-1-2 被動式描述技巧 – 結構變數

傳統語言描述真實環境的『實體』(Entity，或稱事物) 現象，大多利用幾個基本資料型態 (int, float, char 等等) 所構成的結構變數 (如 C 語言的 structure 變數)。亦是，結構變數內包含了若干個元素，每一元素可能由基本資料型態，或另一個結構所產生的變數。基本上，結構是描述實體的結合體，至於會採用元素的多寡，這必須視如何分辨出不同實體之間的差異，與實體的運用範圍。

利用結構變數所描述真實環境，『實體』之間大多是獨立的，很難找出它們之間的關連性。譬如，一般公司員工大略可區分為工作員與經理員兩群組，如利用 C 語言的結構來製作員工資料時，則可能產生兩個獨立分別描述工作員與經理員兩種結構變數，兩者之間也很難找出關連性。其實公司裡的員工許多資料都是相同的，資料型態之間應該有許多相似的地方。譬如，當工作員工升等成經理階層，則他的資料可能需要重新製作。如果改利用物件來宣告，就能輕易地取得兩者之間的連帶關係。另外，某些變數可能會有一些限制，譬如，經理級的薪資可能有最低基本薪額，當我們使用結構變數變更資料時，很難去處理這些特殊限制，但運用物件變數卻很容易達成。

(A) 結構變數描述

吾人用一個簡單範例說明傳統語言描述『實體』的方法。假設某家製造工廠欲建立薪資管理系統，針對某位員工選擇描述的屬性如圖 5-3 所示。以 C 語言為範例，宣告產生『結構變數』與『陣列結構變數』的語法如下：

圖 5-3 結構變數的被動描述

描述工作員的結構變數如下：(C 語言範例)

```
struct worker {
        int ID;                 // 員工代號
        char name[15];          // 姓名
        int age;                // 年齡
        char department[10];    // 服務部門
        int payment;            // 底薪
        int  position;          // 職務加級
}
struct worker Liu;
struct worker Employ[30];
/* 給予 Liu 變數內容 (描述 Liu 個體) */
        Liu.ID = 31012;
        Lin.name = "張大年";
        Liu.age = 27;
        Liu.department = “銷售”;
        Liu.payment = 50000;
        Liu.overtime = 15000;

/* 給予陣列結構變數內容 (描述某一員工屬性) */
        Employ[0].ID = 32011;
        Employ[0].name = "劉有志";
        Employ[0].age = 30;
        Employ[0].department ="製造";
        Employ[0].payment = 40000;
        Employ[0].overtime = 10000;
```

（B）結構變數可能出現問題

(1) 同一資訊系統內，描述某一『實體』的資料結構於不同應用軟體之間是否能保持一致性。譬如，員工資料當然不會僅出現於薪資管理系統，同一公司的其他系統也會使用到員工資料，可能出現同一員工在相同資訊系統裡出現多種描述結構，而這些描述變數之間沒有任何關聯性，常會造成資料相衝突。

(2) 當資訊系統處理模式變更，需要改變描述實體的結構變數時，也許所有資訊系統的結構變數都需要變更。譬如，期望在每一員工的資料裡加入『電話』，則所有結構變數都必須變更。

(3) 對於描述實體的屬性內容無法管制。譬如，經理級的員工薪資有最低保障薪資、最高加班時數等等；資料本身無法判斷正確與否，必須仰賴程式裡加以限制。也就是，結構變數對於某些較重要的屬性無法隱藏與限制。

當然，物件導向程式語言並非僅針對上述缺點來設計，不只解決了上述問題，更加入許多功能。但無論如何，我們還是由這方面切入探討物件的基本功能，之後再研究他額外增加的功能，如此一來，對於學習物件導向的觀念，也許會較容易。

5-1-3 主動式描述技巧 – 物件變數

在物件導向程式設計下，描述真實現象『實體』的物件，不再僅由若干個基本型態的變數所構成。意即不僅包含某些變數成員，同時也包含處理這些變數的方法成員。方法成員基本運用是作為整個物件的輸入與輸出介面，又稱為『輸入/輸出埠口』，如圖 5-4 所示。當某一物件被引用之後，其內部的方法也可能隨之被呼叫執行，然而執行方法成員時，也可能需要獨立的記憶體空間(譬如，程式執行當中會產生的變數) 不可與原主程式記憶體空間相衝突。由此可見，物件是獨立的個體，背負某一特殊任務所構成，因此稱之為『Object』。圖 5-4 為主動式物件變數的概念圖，物件內包含：變數（Variable）與方法（Method）兩種成員。變數成員如同結構變數一般，可用來描述真實環境，某些變數 (隱藏式變數，如 ID 變數) 必須透過『方法』成員才可處理，但較無關鍵性變數則可直接處理 (非隱藏式，如 pay、name 變數)。

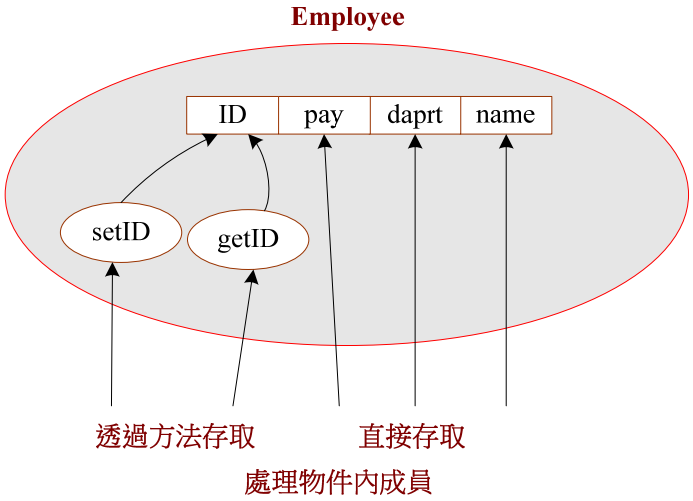

圖 5-4 主動式描述

5-2 物件的概論

5-2-1 物件的屬性

何謂『物件』(Object) 是許多初學者迫切想知道的答案，如果從抽象的哲學觀點來講，可以說成一大篇文章，但相信也很難聽得懂。我們用最簡單的方法來說明『物件』在程式設計裡扮演的角色，了解之後再來推演它的哲學觀念，可能會比較容易懂。首先我們將物件的屬性歸納如下：

(1) 物件是針對某一『實體』(Entity) 的資料表示，並可依照環境需求，制定實體『資料』的規格，實體可以是無形或有形、真實或抽象現象，譬如：人、動物、天氣、感受、圖形、聲音、色彩、語意、個性...等等。簡單的說，真實環境裡任何有形或無形的事項，都可以由規格化的物件來表示。

(2) 為了確保物件規格的完整性，物件實體裡除了有表現實體 (Entity) 的資料外，可能還包含某些程式，透過適當程式來判斷操作者是否超越規格。

(3) 為了確保物件在程式執行當中不被侵犯，物件實體內有獨立的變數、程式與記憶體空間。

5-2-2 物件的產生

規劃物件的藍本即是『類別』(class)。同一個類別可以透過 new() 命令產生多個物件，並給予物件適當名稱，如圖 5-5 所示。簡單的說，『物件』是類別的某一『驗例』(instance，特定範例的意思)。

圖 5-5 物件的產生

類別具有繼承性功能，可由某一類別擴充成另一個類別。譬如，可由學生類別產生『張同學』的學生物件 (透過 new() 命令); 另一方面，也可以由學生類別擴充成『班長類別』(透過 extends 命令)，再由它產生『張班長』的班長物件。

5-2-3 物件的成員

我們可以了解為了確保物件所描述真實現象的完整性，物件 (或類別) 裡必須包含兩種主要成員，如下：

(a) 變數成員 (Variable member)：功能是描述真實現象的資料。

(b) 方法成員 (Method member)：功能是確保物件規格的完整性。

吾人利用變數成員描述真實世界的狀況(無論真實或虛擬世界、有形或無形事務)，又使用方法成員來限定存取變數成員的規則，如此才能規範物件的完整性與主動性。

5-2-4 物件的種類

由上所敘，類別是描述某一真實現象的藍本。為了規範藍本的規格，必須包含著變數與方法兩種成員。但真實運用上並非如此，有些『Entity』並不需要嚴格的規格限制，不需要另外制定『方法』成員；另一方面，物件也可能是一種軟體工具，也可能不需變數成員來描述 Entity 的樣式。因此，物件可能出現下列三種格式：(如圖 5-6 所示)

(1) 傳統物件變數(被動式物件)：物件內僅包含變數成員，並沒有方法成員。我們可以將它當成傳統程式語言(如 C 語言)的結構變數使用，本章將介紹此物件的運用方法。

(2) 物件導向(主動式物件)：物件 (或類別) 內包含變數與方法兩種成員，可主動性的描

述任何真實現象，第 6～8 章介紹。

(3) 物件方法套件(物件方法庫)：物件 (或類別) 內僅包含方法成員，不具有變數成員 (如果有也不是描述事件使用)，本書第 9 章介紹。

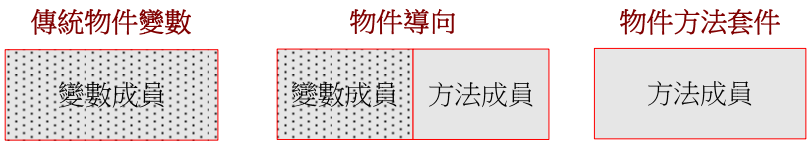

圖 5-6 物件的三種型態

5-2-5 類別宣告與物件產生

利用物件變數來描述真實環境。首先必須建構一個可以產生物件變數的類別，再利用此類別來宣告產生物件。僅有變數成員的類別宣告語法如下：

	語　　法	範　　例

類別
宣告
	class 類別名稱 {
	　變數型態 變數名稱_1;
	　變數型態 變數名稱_2;
	　…..
	}
		class Employee {
		　int ID;
		　String name;
		　String depart;
		　int payment;
		　int duty;
		}

物件 產生	類別名稱 物件名稱 = new 類別名稱(); 物件名稱.變數名稱_1 = 數值;	Employee Liu = new Employee(); Liu.ID = 2345; Liu.name = "張大名";

(1) class 類別名稱 {….}：宣告類別的語法，其中 class 為關鍵字，左右大括號內為變數成員。每一個變數成員可獨立宣告變數型態，甚至另一種類別型態 (如 String)。

(2) 類別名稱 物件名稱 = new 類別名稱()：利用類別宣告並產生物件。譬如 Employee Liu = new Employee() 表示利用 Employee 宣告 Liu 物件型態，再利用 new 產生一個 Employee 型態的物件，並填入 Liu 物件變數內(如同 int data = 5; 的意思相同)

(3) 物件名稱.變數名稱_1：變數成員表示。譬如所產生的 Liu 物件變數，其成員有：Liu.ID、Liu.name、Liu.depart...等

圖 5-7 為宣告 Employee 類別，並由它產生物件的運作情況。圖中包含兩個類別：Employee 與 Ex5_1，其中 Employee 類別僅有變數成員，可以宣告產生物件；Ex5_1 是靜態類別（容後介紹），可不經由 new 產生物件，而直接呼叫執行（即是中介碼，Bytecode）。

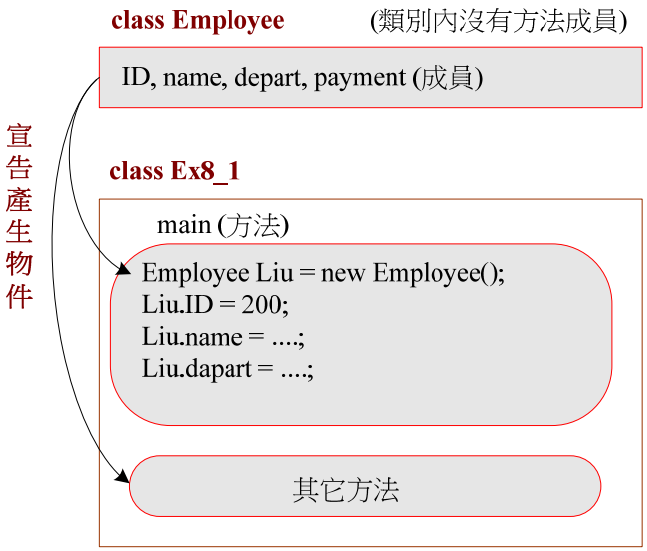

圖 5-7 物件變數宣告產生範例

5-3 物件的傳統運用

5-3-1 範例研討：規劃員工薪資資料

（A）程式功能：Ex5_1.java

『志明電器公司』希望建立電子化企業管理系統，人事部門所需的薪資管理子系統，需要規劃員工薪資資料，其中包含有：員工代號、姓名、部門、底薪與職務加級。請設計相關資料型態，並測試輸入/輸出結果，期望操作介面如下：

```
D:\Java2_book\chap5\Ex5_1>javac Ex5_1.java

D:\Java2_book\chap5\Ex5_1>dir/b
Employee.class                          // Employee 類別
Ex5_1.class                             //Ex5_1 類別
Ex5_1.java

D:\Java2_book\chap5\Ex5_1>java Ex5_1
```

```
***** 輸入員工資料 *****
輸入員工姓名 =>張大名
輸入員工所屬部門 =>資訊部
輸入員工代號 =>430
輸入員工底薪 =>25000
輸入薪資加級 =>12000

***** 印出員工資料 *****
員工代號 = 430
員工姓名 = 張大名
所屬部門 = 資訊部
底　　薪 = 25000
職務加級 = 12000
```

備註：Ex5_1.java 經由 javac 編譯後，產生兩只中介檔案（Bytecode）檔案，一則由 class Employee{ …} 程式區塊所產生的 Employee.class 檔案；另一則由 class Ex5_1 {….} 區塊所產生的 Ex5_1.class 檔案。

（B）製作技巧研討：

當利用物件變數描述真實現象，最困擾的是應具有哪些變數成員，才能夠明確的表示該『實體』的屬性。其實，採用多少或選用哪些變數成員，並沒有一定的規範，重點是應用系統裡需要處理哪些資料，或實體之間需要哪些資料才能辨識。『志明電器公司』員工應該具有哪些資料呢？首先須思考哪些資料可以辨識不同員工，假設每一員工都有唯一的員工代號，就可以用來區分每一位員工之間的不同點。另一方面，需合乎該員工資料是要提供給哪種系統使用，假設是薪資管理系統，則可能需要登錄員工的底薪、職務加級、以及加班時數等等。因此，吾人必須利用一個類別（Employee）來描述員工薪資資料，主程式再利用它產生員工物件，也利用所產生的員工物件，儲存某位員工（emp）各種屬性的數值，如圖 5-8 所示。

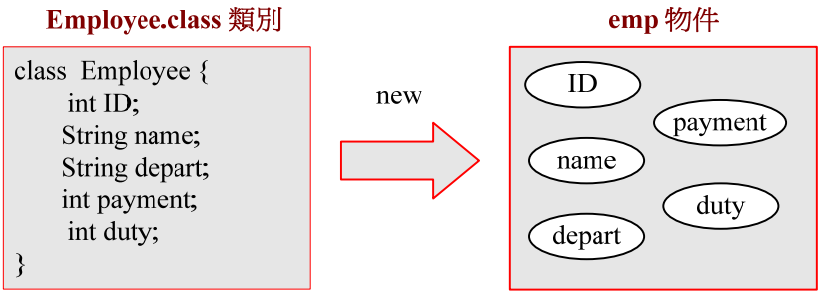

圖 5-8 利用 Employee 類別產生員工物件

（C）程式範例

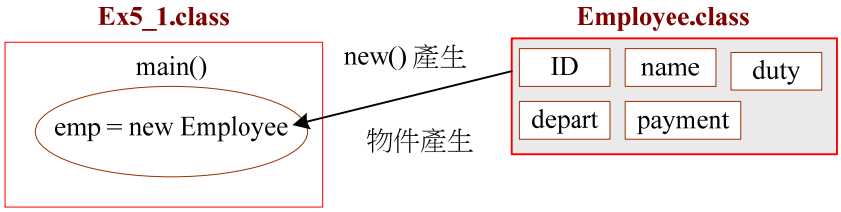

圖 5-9 Ex5-1 程式架構

```
01 //Ex5_1.java
02 /* 請建立一套電子化企業管理系統，
03
04     其中包含:員工代號、姓名、部門、底薪與職務加級 */
05
06 import java.util.*;
07 /* 宣告類別 */
08
09 class Employee {
10         int ID;                 //員工代號
11         String name;            // 員工姓名
12
13         String depart;      // 服務部門
14         int payment;            // 底薪
15
16         int duty;               // 職務加級
17 }
18 public class Ex5_1{
19       public static void main(String args[]) {
20             Scanner keyin = new Scanner(System.in);
21
22             /* new 產生物件變數 */
23
24             Employee emp = new Employee();
25             System.out.printf("***** 輸入員工資料 *****\n");
```

```
26              System.out.printf("輸入員工姓名 =>");
27              emp.name = keyin.nextLine();
28              System.out.printf("輸入員工所屬部門 =>");
29              emp.depart = keyin.nextLine();
30              System.out.printf("輸入員工代號 =>");
31              emp.ID = keyin.nextInt();
32              System.out.printf("輸入員工底薪 =>");
33              emp.payment = keyin.nextInt();
34              System.out.printf("輸入薪資加級 =>");
35              emp.duty = keyin.nextInt();
36
37              System.out.printf("\n***** 印出員工資料 *****\n");
38              System.out.printf("員工代號 = %d\n", emp.ID);
39              System.out.printf("員工姓名 = %s\n", emp.name);
40              System.out.printf("所屬部門 = %s\n", emp.depart);
41              System.out.printf("底　　薪 = %d\n", emp.payment);
42              System.out.printf("職務加級 = %d\n", emp.duty);
        }
    }
```

（D）程式重點分析：

(1) 第 5~11 行：『class Employee { ….} 』。宣告所欲產生物件變數的類別，該程式區塊會獨立產生 Employee.class 檔案，也可讓其他程式引用。

(2) 第 12~40 行：『class Ex5_1 { …} 』。主類別程式區塊，經由 javac 編譯後，會獨立產生 Ex5_1.class 檔案。

(3) 第 19 行：『Employee emp = new Employee(); 』。利用 Employee 類別宣告產生 emp 物件變數，並產生 Employee 物件（new Employee()）填入其中。該行敘述欲執行有效，必須 Employee.class(類別檔案)是在同一目錄底下，不然就需利用 import 命令指定位於哪一個目錄下並導入。

(4) 第 21~ 31 行：鍵盤輸入填入 emp.ID、emp.name、emp.depart...等變數成員內。

(5) 第 34~39 行：螢幕輸出 emp.ID、emp.name、emp.depart...等變數成員內。

5-3-2 範例研討：薪資扣繳所得稅

（A） 程式功能：Ex5_2.java

『志明電器公司』會計部門需要計算幫每位同仁預扣所得稅，稅額是總領薪資的10%，請製作一套薪資扣繳所得稅系統，期望操作介面如下：

```
D:\Java2_book\chap5\Ex5_1>javac Ex5_2.java

D:\Java2_book\chap5\Ex5_1>dir/b
Employee.class
Ex5_1.class
Ex5_1.java
Ex5_2.class
Ex5_2.java

D:\Java2_book\chap5\Ex5_1>java Ex5_2

***** 列印員工稅額 *****

員工代號= 43210 姓名= 張大名 部門= 製造部 薪資= 65000 預扣稅額= 6500
```

備註：Employee.class 類別可與其他程式共用。

（B）製作技巧研討：

吾人利用原來薪資管理系統所建立的 Employee.class 類別，再來製作薪資扣繳所得稅系統。於同一電子化企業系統內，使用相同型態的物件變數，資料較能保持一致性，但類別檔案必須存放於相同目錄下，否則必須用 import 導入。

（C）程式範例：

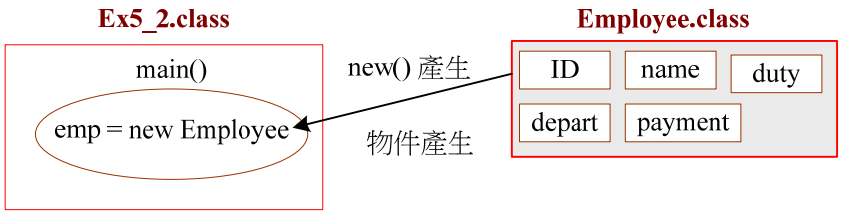

圖 5-10

```
01 //Ex5_2.java
02
```

```
03 /*   Employee.class 檔案需存在於同目錄下 */
04 public class Ex5_2 {
05       public static void main(String args[]){
06             int tax;
07             /* new 產生物件變數 */
08             Employee emp = new Employee();
09             emp.name = "張大名";
10             emp.depart = "製造部";
11             emp.ID = 43210;
12             emp.payment = 50000;
13             emp.duty = 15000;
14
15             System.out.printf("\n***** 列印員工稅額 *****\n");
16             System.out.printf("員工代號= %d ", emp.ID);
17             System.out.printf("姓名= %s ", emp.name);
18             System.out.printf("部門= %s ", emp.depart);
19             System.out.printf("薪資= %d ", emp.payment+emp.duty);
20             tax = (int)((emp.payment+emp.duty)*0.1);
21             System.out.printf("預扣稅額= %d\n", tax);
22       }
23 }
24
```

（D）程式重點說明：

(1) 第 7 行：『Employee emp = new Employee();』。利用 Employee.class 類別宣告產生 emp 物件變數。

5-3-3 自我挑戰：產生商品資料

（A）程式功能：PM5_1.java

請您幫『春嬌生鮮超市』建立商品資料登錄系統，為了配合倉庫管理系統、銷售管理系統、會計管理系統使用，每一商品必需登錄有：編號（String）、名稱（String）、單價（int）、庫存量（int）、單位（String）、製造商（String）、安全庫存量（int）。請編寫一系統，讓管理員可登錄商品資料，登錄後再由螢幕顯出結果。期望操作介面如下：

```
D:\Java2_book\chap5\PM5_1>javac PM5_1.java
```

```
D:\Java2_book\chap5\PM5_1>dir/b
Article.class
PM5_1.class
PM5_1.java

D:\Java2_book\chap5\PM5_1>java PM5_1
*****  輸入商品資料  *****
請輸入商品編號  =>A10134
請輸入商品名稱  =>肉燥麵
請輸入單價  =>20
請輸入庫存量  =>200
請輸入安全庫存量  =>50
請輸入單位  =>包
請輸入製造商  =>統一食品股份有限公司

*****  印出商品資料  *****
商品編號  = A10134
商品名稱  =  肉燥麵
單        價  = 20
庫  存  量  = 200
安全庫存量  = 50
單        位  =  包
製  造  商  =  統一食品股份有限公司
```

（B）製作技巧提示：

將所欲規描述商品的『屬性』規劃成類別，再利用它產生物件變數，如圖 5-9 所示。接著，輸入商品各種屬性的內容，再將物件變數內容印出即可。虛擬碼提示如下：

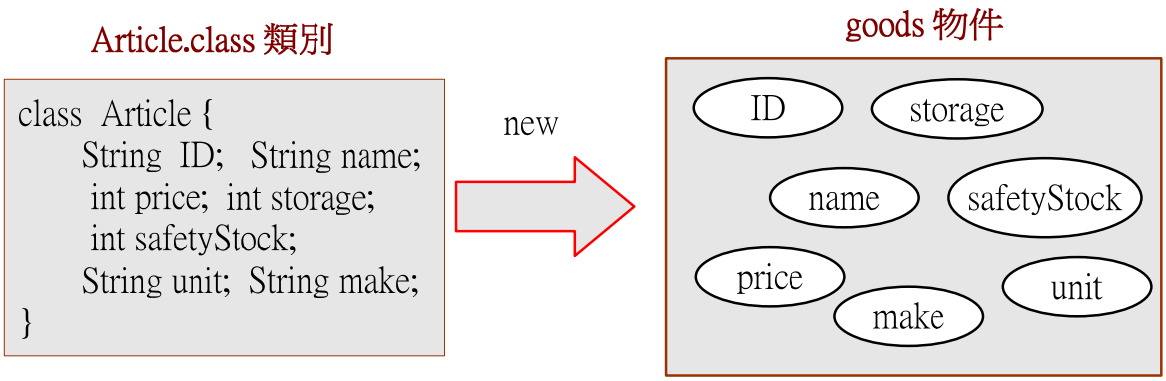

圖 5-11 利用 Article 類別產生商品物件

程式架構如圖 5-11 所示。

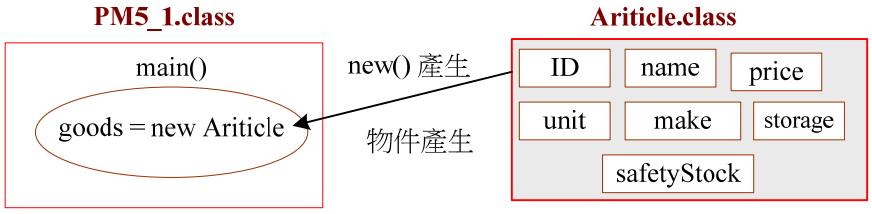

圖 5-12 PM5_1 程式架構圖

```
/* 宣告產品類別 */
class Article {
        String ID;              // 產品編號
        String name;            // 產品名稱
        int price;              // 單價
        int storage;            // 庫存量
        int safetyStock;        // 安全庫存量
        String unit;            // 單位
        String maker;           // 製造商
}
….
/* 宣告產生物件變數 */
Article goods = new Article();
輸入商品資料並填入物件的變數成員內；
輸出顯示物件的變數成員；
```

5-4 物件陣列的運用

5-4-1 物件陣列的宣告

雖然利用二維陣列也可以描述真實環境，但陣列內所有欄位的資料型態必須相同，如此一來，所能描述的功能會受到很大的限制。一般非物件導向（如 C 語言），利用結構陣列來擴充結構變數的功能，使其能描述多個屬性相同的真實現象；同樣的，Java 也可以利用物件陣列來表示多個屬性相同的真實現象。宣告語法如下（如圖 5-8 所示）：

	物件陣列宣告語法	範　　例

類別
宣告
	class class_name {
	…..
	}
		class Employee {
		….
		}

物件陣列 宣告	class_name[] object_name; object_name = new class_name[n];	Employee[] worker; worker = new Employee[20];

物件
產生
	object_name[k] = new class_name();
	…
	object_name[k].membe r= value;
		worker[k] = new Employee();
		….
		worker[k].ID = 71209;

(1) 敘述『class_name[] object_name;』（如 Employee[] worker;）。宣告產生類別 class_name 的物件陣列，名稱為 object_name。

(2) 敘述『object_name = new class_name[n];』（如 worker = new Employee[20];）。預留產生 n 個元素的物件陣列。

(3) 敘述『object_name[k] = new class_name();』（如 worker[k] = new Employee();）。 產生 class_name 類別的物件，填入物件陣列的第 k 個元素內。

(4) 敘述範例『worker[k].ID = 71209;』。將數值 71209 填入陣列物件的第 k 個元素的 ID 變數成員內。

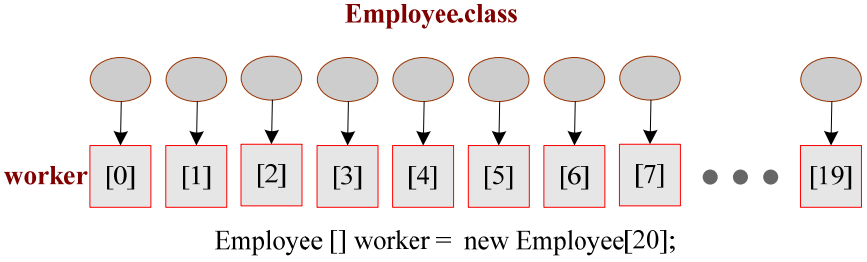

圖 5-13 物件陣列的宣告與產生

5-3-2 範例研討：簡單薪資管理系統

（A）程式功能：Ex5_3.java

請幫志明電器製造公司建立一套薪資管理系統，允許輸入多筆員工資料 (員工代號、姓名、服務部門、底薪與職務加級)，並印出員工薪資表，包含有：員工代號、姓名、服務部門、薪資總額、扣繳稅額 (= 總額 * 0.1) 與應領金額 (= 薪資總額 – 扣繳稅額)。期望操作介面如下：

```
D:\Java2_book\chap5\Ex5_3>javac Ex5_3.java
D:\Java2_book\chap5\Ex5_3>dir/b
Employee.class
Ex5_3.class
Ex5_3.java
D:\Java2_book\chap5\Ex5_3>java Ex5_3
是否繼續輸入員工資料 (yes/no) =>yes
***** 輸入員工資料 *****
輸入員工姓名 =>羅大仙
輸入所屬部門 =>製造課
輸入員工代號 =>720
輸入員工底薪 =>50000
輸入薪資加級 =>12000
是否繼續輸入員工資料 (yes/no) =>yes
***** 輸入員工資料 *****
輸入員工姓名 =>張真人
輸入所屬部門 =>業務課
輸入員工代號 =>610
輸入員工底薪 =>70000
輸入薪資加級 =>12000
是否繼續輸入員工資料 (yes/no) =>no
          ***** 印出員工薪資表 *****
 員工資料                    薪資總額          預扣稅額          應領薪資
 羅大仙 (720 製造課)         62000             6200              55800
```

張真人 (610 業務課)	82000	8200	73800

（B）製作技巧研討：

首先必須設計員工薪資各種屬性的資料型態，並將其宣告成類別（class Employee {…}），經過編譯後會產生一個獨立的中介碼（Employee.class）。再利用此類別宣告產生物件陣列（worker[]），期望員工資料儲存方式如圖 5-10 所示；每一行表示一筆員工資料，各個欄位表示所描述員工的屬性，但並非二維陣列的元素，而是描述員工物件的變數成員。

	員工代號	員工姓名	工作單位	底　薪	職務加級
worker[0]	71209	張志明	製造部	40000	25000
worker[1]	71210	羅大仙	製造部	43000	16800
worker[2]	71214	陳明郎	生管部	28000	12000
worker[3]	71219	陳中心	倉管部	32000	17000
worker[4]	71242	劉新明	行銷部	28000	15000

圖 5-14 物件陣列範例

假設系統最高可以儲存 20 筆員工資料，又系統可選擇連續輸入多筆資料，因此吾人必須利用 while 迴圈判斷是否超過界線，或繼續輸入資料；另外利用 k 變數紀錄目儲存了幾筆資料，或願意繼續輸入資料的話，則再產生一個物件變數填入 k 所指的陣列元素上(worker[k] = new Employee();)。輸入完畢後，再依照 k 所紀錄的資料數量，分別計算薪資總額（= 底薪 + 職務加級）、扣繳稅額（= 薪資總額 * 0.1）與應領薪資（= 薪資總額 – 扣繳稅額），並輸出到螢幕上。

（C）程式範例：

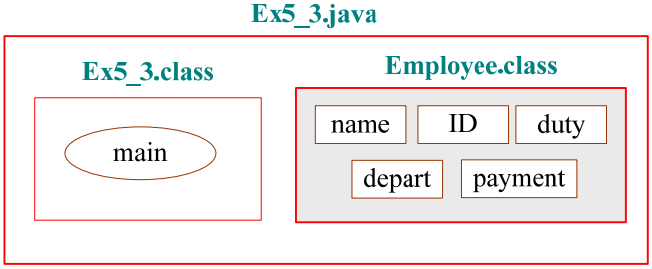

圖 5-15 Ex5_3 程式架構

```
01 //Ex5_3.java
02 /* 請建立一套薪資管理系統，允許輸入員工資料，並印出薪資表 */
03
04
05 import java.util.*;
06 class Employee {          // 員工資料類別
07         int ID;              // 員工代號
08
09         String name;         // 員工姓名
10
11         String depart;     // 服務部門
12
13         int payment;         // 底       薪
14
15         int duty;            // 職務加級
16 }
17
18 public class Ex5_3 {
19         public static void main(String args[]){
20                 Scanner keyin = new Scanner(System.in);
21                 Employee[] worker;
22                 worker = new Employee[20];
23                 int value, tax, k=0;
24                 String sel, temp;
25
26                 System.out.printf("是否繼續輸入員工資料 (yes/no) =>");
27                 sel = keyin.nextLine();
28                 while(sel.equals("yes") && (k <20)) {
29                         worker[k] = new Employee();
30                         System.out.printf("***** 輸入員工資料 *****\n");
31                         System.out.printf("輸入員工姓名 =>");
32                         worker[k].name = keyin.nextLine();
33                         System.out.printf("輸入所屬部門 =>");
34                         worker[k].depart = keyin.nextLine();
35                         System.out.printf("輸入員工代號 =>");
36                         worker[k].ID = keyin.nextInt();
37                         System.out.printf("輸入員工底薪 =>");
38                         worker[k].payment = keyin.nextInt();
39                         System.out.printf("輸入薪資加級 =>");
40                         worker[k].duty = keyin.nextInt();
41                         keyin.nextLine();
42
43                         System.out.printf("是否繼續輸入員工資料 (yes/no) =>");
44                         sel = keyin.nextLine();
45
46
```

```
47                      k = k+1;
48              }
49
50              System.out.printf("              ***** 印出員工薪資表 *****\n");
51              System.out.printf("  員工資料\t\t 薪資總額\t 預扣稅額\t 應領薪資\n");
52              for (int i=0; i<k; i++) {
53                      System.out.printf(" %s ", worker[i].name);
54                      System.out.printf("(%d ", worker[i].ID);
55                      System.out.printf("%s)\t", worker[i].depart);
56                      value = worker[i].payment+worker[i].duty;
57                      System.out.printf("    %d\t\t", value);
58                      tax = (int)(value * 0.1);
59                      System.out.printf("%d\t\t", tax);
                        value = value - tax;
                        System.out.printf("%d\n", value);
                }
        }
}
```

（D）程式重點說明：

(1) 第 20 行：『int k=0;』。利用變數 k 紀錄目前資料的比數有多寡；宣告時給予初始值 0。

(2) 第 28~45 行：『while(sel.equal(“yes”) && (k<20)) { …. sel = keyin.readLine(); k=k+1;}』。判斷如果輸入 yes 或 k 還小於 20，則繼續處理迴圈內敘述區塊。

(3) 第 30 行：『worker[k] = new Employee()』。產生一個 Employee 類別的物件，並填入 worker 陣列的第 k 個元素內。

(4) 第 32 行：『worker[k].name = keyin.readLine();』。讀入一行字串，並存入物件 worker[k] 的 name 變數成員內。

5-4-3 範例研討：超商販賣系統

（A）程式功能：Ex5_4.java

請幫『春嬌生鮮超市』建立一套販賣系統，電腦視窗能出現各項商品名稱與單價，操作者連續點選客人購買商品與數量，最後顯示出購買商品清單，並計算出購買金額。

假設僅販賣：可口餅乾（20 元）、味全鮮乳（30 元）、御便當（50 元）、黑松汽水（20 元）、頻果西打（30 元）與脆笛酥（20 元），期望操作介面如下：

```
D:\Java2_book\chap5\Ex5_4>javac Ex5_4.java

D:\Java2_book\chap5\Ex5_4>dir/b
Element.class
Ex5_4.class
Ex5_4.java

D:\Java2_book\chap5\Ex5_4>java Ex5_4
  **** 歡迎光臨  春嬌超商 請選擇購買商品  ****
(1) 可口餅乾 20 元      (2) 味全鮮乳 30 元      (3) 御便當 50 元
(4) 黑松汽水 20 元      (5) 頻果西打 30 元      (6) 脆迪酥 20 元
(7) 結算金額
        請輸入選項 =>2
        購買數量 =>5
  **** 歡迎光臨  春嬌超商 請選擇購買商品  ****
(1) 可口餅乾 20 元      (2) 味全鮮乳 30 元      (3) 御便當 50 元
(4) 黑松汽水 20 元      (5) 頻果西打 30 元      (6) 脆迪酥 20 元
(7) 結算金額
        請輸入選項 =>3
        購買數量 =>2
  **** 歡迎光臨  春嬌超商 請選擇購買商品  ****
(1) 可口餅乾 20 元      (2) 味全鮮乳 30 元      (3) 御便當 50 元
(4) 黑松汽水 20 元      (5) 頻果西打 30 元      (6) 脆迪酥 20 元
(7) 結算金額
        請輸入選項 =>7
***** 購買清單如下 *****
品名            單價    數量    小計
味全鮮乳        30      5       150
御便當          50      2       100
總計 = 250
```

（B）製作技巧研討：

系統要求可連續選擇多項商品，再計算出總金額及印出購買清單，因此，必須準備一只物件陣列，存放客戶所購買產品名稱及數量。吾人規劃產品物件包含有：產品名稱、單價、數量、以及小計，其中小計表示該樣產品的金額。客戶選購產品時，則需產生陣列內的物件元素，再填入該產品的屬性，如圖 5-11 所示。

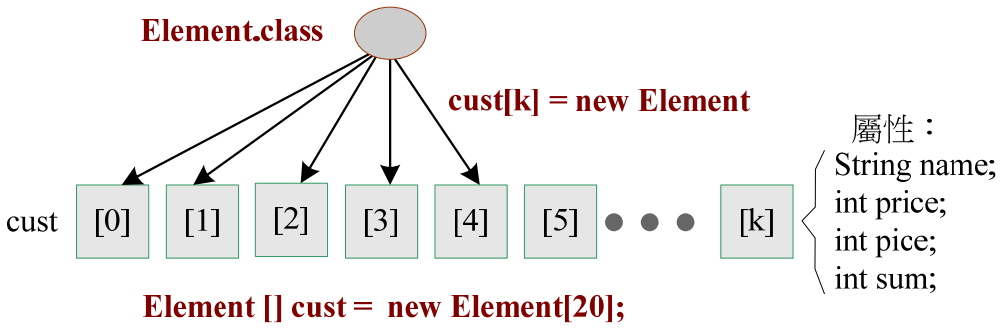

圖 5-16 Element 物件填入 cust 陣列中

另外，製作本系統重點提示如下：

(1) 系統操作時隨時由螢幕點選購買產品及數量，因此需將顯示產生清單編寫成一個獨立的函數，隨時呼叫執行（disp_Element()）。

(2) 需要一個陣列物件隨時登錄客戶所購買產品及數量，並規劃該物件的類別內容（class Element{ …} ）。

(3) 利用 while 迴圈與 switch/case 判斷選擇店員的操作輸入，並假設客戶最多購買 20 樣產品。

（C）程式範例：

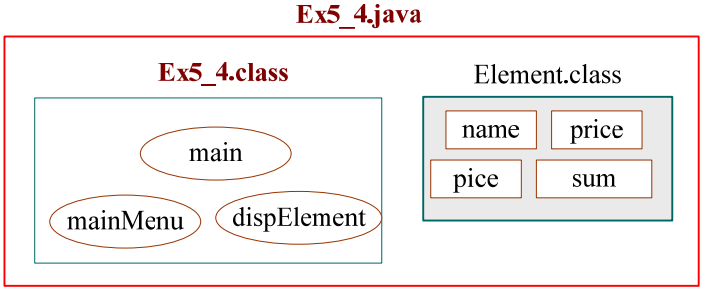

圖 5-17 Ex5_4 程式架構

```
01 //Ex5_4.java
02 /* 請建立一套超商販賣系統，能顯示商品名稱與單價，
03
04     提供客人購買，最後印出結果 */
05
06 import java.util.*;
07 class Element {
08       String name;     // 商品名稱
09
10       int price;       // 單價
11
12       int pice;        // 購買數量
13
14       int sum;         // 小計
15 }
16 public class Ex5_4 {
17     public static void main(String args[]) {
18           Scanner keyin = new Scanner(System.in);
19
20            Element[] cust = new Element[20]; // 登錄客戶購買產品
21            int select, k, no, total;
22            dispElement();
23            select = keyin.nextInt();
24            k=0;
25            while ((select != 7) && (k<20)) {
26                  System.out.printf("\t 購買數量 =>");
27                  no = keyin.nextInt();
28                  switch (select) {
29                        case 1:
30                              cust[k] = new Element();
31                              cust[k].name = "可口餅乾";
32                              cust[k].price = 20;
33                              cust[k].pice = no;
34                              k = k + 1;
35                              break;
36                        case 2:
37                              cust[k] = new Element();
38                              cust[k].name = "味全鮮乳";
39                              cust[k].price = 30;
40                              cust[k].pice = no;
41                              k = k + 1;
42                              break;
43                        case 3:
44                              cust[k] = new Element();
45                              cust[k].name = "御便當";
46
```

```
47                              cust[k].price = 50;
48                              cust[k].pice = no;
49                              k = k + 1;
50                              break;
51                      case 4:
52                              cust[k] = new Element();
53                              cust[k].name = "黑松汽水";
54                              cust[k].price = 20;
55                              cust[k].pice = no;
56                              k = k + 1;
57                              break;
58                      case 5:
59                              cust[k] = new Element();
60                              cust[k].name = "頻果西打";
61                              cust[k].price = 30;
62                              cust[k].pice = no;
63                              k = k + 1;
64                              break;
65                      case 6:
66                              cust[k] = new Element();
67                              cust[k].name = "脆迪酥";
68                              cust[k].price = 20;
69                              cust[k].pice = no;
70                              k = k + 1;
71                              break;
72                      default:
73                              System.out.printf("請重新正確選擇 \n");
74                              break;
75              }
76              dispElement();
77              select = keyin.nextInt();
78      }
79
80      /* 計算各產品小計與購買總金額 */
81      total = 0;
82      for (int i=0; i<k; i++) {
83              cust[i].sum = cust[i].price * cust[i].pice;
84              total = total + cust[i].sum;
85      }
96
87      /* 列印購買清單 */
88      System.out.printf("***** 購買清單如下 *****\n");
89
90
91
92
```

```
93              System.out.printf("品名\t\t 單價  \t 數量\t  小計\n");
94              for (int i=0; i<k; i++)
95                      System.out.printf("%s%10d%10d%10d\n", cust[i].name,
96                              cust[i].price, cust[i].pice, cust[i].sum);
97              System.out.printf("總計  = %d\n", total);
98      }
99
100     /* 顯示客戶購買商品選項 */
101     public static void dispElement(){
102        System.out.printf("   ** 歡迎光臨   春嬌超商  請選擇購買商品   ****\n");
103             System.out.printf("(1) 可口餅乾  20  元\t");
104             System.out.printf("(2) 味全鮮乳  30  元\t");
105             System.out.printf("(3) 御便當  50  元\n");
106             System.out.printf("(4) 黑松汽水  20  元\t");
                System.out.printf("(5) 頻果西打  30  元\t");
                System.out.printf("(6) 脆迪酥  20  元\n");
                System.out.printf("(7) 結算金額\n");
                System.out.printf("\t 請輸入選項  =>");
    }
}
```

（D）程式重點說明：

(1) 第 4~9 行：『class Element{ …} 』。 宣告物件陣列的原始類別。

(2) 第 15 行：『Element[] cust = new Element[20];』。產生準備登錄客戶謀買商品的物件陣列，假設最高可儲存 20 樣產品。

(3) 第 19 行：『k=0;』。利用 k 變數紀錄客戶購買商品的筆數；其初始值為 0。

(4) 第 20~75 行：『while(..) { switch(..) { ….}} 』。利用 while 迴圈與 switch/case 判斷選項，構成系統主要執行區域。

(5) 第 25 行：『cust[k] = new Element();』。變數 k 是紀錄目前存放資料的筆數，又它是由 0 開始計算，因此 k 所指的是下一個資料的位置。此敘述功能是產生一個 Elment 物件後，存入陣列 cust 的第 k 位置（cust[k]）。

5-4-4 自我挑戰：超商庫存管理系統

（A）程式功能：PM5_2.java

請您幫『春嬌生鮮超市』建立一套庫存管理系統，該系統功能有：盤點庫存量（顯示所有商品資料）、進貨登錄功能（點選商品編號、再輸入進貨量）、出貨登錄功能（點選商品編號，再輸入出貨量）。假設該超商僅有：可口餅乾（編號 A1001）、味全鮮乳（編號 A1002）、御便當（編號 A1003）、黑松汽水（編號 A1004）、頻果西打（編號 A1005）與脆笛酥（編號 A1006），功能如下：

(1) 期望操作介面有 5 個功能選項

```
D:\Java2_book\chap5\PM5_2>java PM5_2

 ** 歡迎光臨  春嬌超商 庫存管理系統  **
(1) 盤點庫存量      (2) 進貨登錄系統   (3) 出貨管理系統   (4) 離開系統
          請輸入選項 =>
```

(2) 選擇進貨登錄系統(選擇 2)操作如下：

```
** 歡迎光臨  春嬌超商 庫存管理系統  **
(1) 盤點庫存量      (2) 進貨登錄系統   (3) 出貨管理系統   (4) 離開系統
          請輸入選項 =>2
(1)可口餅乾(A1001)        (2)味全鮮乳(A1002)        (3)御便當  (A1003)
(4)黑松汽水(A1004)        (5)頻果西打(A1005)        (6)脆迪蘇  (A1006)
          請輸入選項 =>3
請輸入進貨數量 =>100
```

(3) 選擇盤點庫存量(選擇 1)操作如下：

```
** 歡迎光臨  春嬌超商 庫存管理系統  **
(1) 盤點庫存量      (2) 進貨登錄系統   (3) 出貨管理系統   (4) 離開系統
          請輸入選項 =>1
*****  盤點庫存量如下  *****
產品編號          產品名稱          庫存量
```

```
A1001        可口餅乾        300
A1002        味全鮮乳        30
A1003        御便當          100
A1004        頻果西打        50
A1005        頻果西打        20
A1006        脆迪酥          45
```

(4) 選擇出貨管理系統(選擇 3 與 1)操作如下：

```
** 歡迎光臨   春嬌超商 庫存管理系統   **
(1) 盤點庫存量      (2) 進貨登錄系統   (3) 出貨管理系統   (4) 離開系統
            請輸入選項 =>3
(1)可口餅乾(A1001)        (2)味全鮮乳(A1002)        (3)御便當   (A1003)
(4)黑松汽水(A1004)        (5)頻果西打(A1005)        (6)脆迪蘇   (A1006)
            請輸入選項 =>2
請輸入出貨數量 =>20

 ** 歡迎光臨   春嬌超商 庫存管理系統   **
(1) 盤點庫存量      (2) 進貨登錄系統   (3) 出貨管理系統   (4) 離開系統
            請輸入選項 =>1
***** 盤點庫存量如下 *****
產品編號        產品名稱        庫存量
A1001        可口餅乾        300
A1002        味全鮮乳        10
A1003        御便當          100
A1004        頻果西打        50
A1005        頻果西打        20
A1006        脆迪酥          45
```

（B）製作技巧提示：

本系統較特殊的地方是，系統開始時需將所預定的 6 樣產品資料填入物件陣列中，吾人利用一個二維陣列（article[][]）儲存這些資料，再寫一個簡單程式將他複製到物件陣列的

產品名稱與編號上。其他程式規劃就較為簡單。

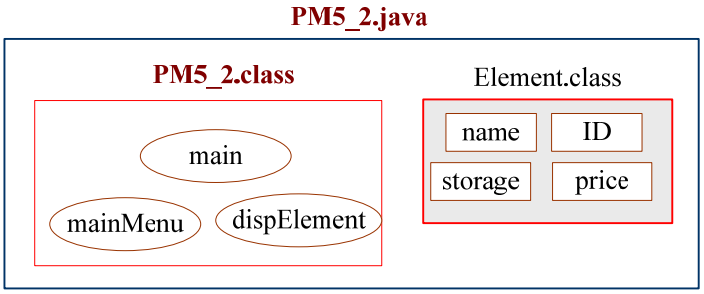

圖 5-18 PM5_2 程式架構

虛擬碼提示如下：

```
宣告描述商品的類別：
        class Element {
            String name;          // 商品名稱
            String ID;            // 商品編號
            int storage;          // 庫存數量
            int price;            // 單價
        }
宣告主類別範圍（ PM5_2 ）：{
  主方法（main() ）範圍{
    宣告物件陣列（ Element[] cust = new Element[20] ）；  // 庫存陣列
    宣告商品名稱與編號陣列：
            String[][] article = {{"可口餅乾", "A1001"},
                                  {"味全鮮乳", "A1002"},
                                  {"御便當   ", "A1003"},
                                  {"頻果西打", "A1004"},
                                  {"頻果西打", "A1005"},
                                  {"脆笛酥   ", "A1006"}};
    將商品名稱與編號填入物件陣列中：
            for (int i=0; i<6; i++) {
                cust[i] = new Element();
                cust[i].name = article[i][0];
```

```
                    cust[i].ID = article[i][1];
                }
        顯示並選擇輸入主工作目錄（mainMenu(), select）
        while ((select != 4)) {
            switch (select) {
                case 1:
                    顯示庫存量（顯示 cust[] 物件陣列內容）;
                case 2:
                    顯示商品選單及輸入（disp_Element(), art）;
                    讀取進貨數量（number）;
                    累增至庫存量（cust[art-1].storage =+ number）;
                case 3:
                    顯示商品選單及輸入（disp_Element(), art）;
                    讀取進貨數量（number）;
                    扣除庫存量（cust[art-1].storage =- number）;
                default:
                     顯示錯輸入；
            }
            顯示並選擇輸入主工作目錄（mainMenu(), select）
        }
    }
     顯示存管理系統工作選項函數（mainMenu()）;
     顯示庫存產品函數（disp_Element()）;
}
```

程式片段如下：

```
01  .....
02  class Element {
03      String name;          // 商品名稱
04
05      String ID;            // 商品編號
06      int storage;          // 庫存數量
07
08      int price;            // 單價
09  }
10  public class PM5_2 {
11      public static void main(String args[]) {
```

```
12          Scanner keyin = new Scanner(System.in);
13          Element[] cust = new Element[20]; //  庫存資料檔案
14
15
16          String[][] article = {{"可口餅乾", "A1001"},
17                                {"味全鮮乳", "A1002"},
18                                {"御便當   ", "A1003"},
19                                {"頻果西打", "A1004"},
20                                {"頻果西打", "A1005"},
21                                {"脆迪酥   ", "A1006"}};
22          int select, art, number;
23          for (int i=0; i<6; i++) {
24                cust[i] = new Element();
25                cust[i].name = article[i][0];
26                cust[i].ID = article[i][1];
27          }
28          mainMenu();
29          select = keyin.nextInt();
30          while ((select != 4)) {
31                switch (select) {
32                      case 1:
33                            System.out.printf("*  盤點庫存量如下  **\n");
34                            System.out.printf("產品編號\t 產品名稱\t 庫存量\n");
35                            for (int i=0; i<6; i++)
36                                  System.out.printf("%s\t\t%s\t%d\n", cust[i].ID,
37                                        cust[i].name, cust[i].storage);
38                            break;
39                      case 2:
40                            disp_Element();
41                            art = keyin.nextInt();
42                            System.out.printf("請輸入進貨數量  =>");
43                            number = keyin.nextInt();
44                            cust[art-1].storage = cust[art-1].storage + number;
45                            break;
46                      case 3:
47                            disp_Element();
48                            art = keyin.nextInt();
49                            System.out.printf("請輸入出貨數量  =>");
50                            number = keyin.nextInt();
51                            if(number > cust[art-1].storage)
52                                System.out.printf("庫存不足, 拒絕出貨\n");
53                            else
54
55
56
57
```

```
58                                  cust[art-1].storage = cust[art-1].storage - number;
59                              break;
60                      default:
61                              System.out.printf("請重新正確選擇 \n");
62                              break;
63                  }
64                  mainMenu();
65                  select = keyin.nextInt();
66              }
67      }
68
69      /* 顯示庫存管理系統工作選項 */
70      public static void    mainMenu(){
71              .......
72      }
73
74      /* 顯示庫存產品 */
        public static void disp_Element() {
                ......
                .....
        }
}
```

5-4 專題製作 – 真健康美食餐廳

公司接受『真健康美食餐廳』委託建立菜單管理系統。起初雙方溝通很困難，因此，公司希望一步一步的建立，每建立一個步驟後雙方達成共識後再繼續下一個步驟，以下是依照業者需求按步驟完成。

5-4-1 範例研討：建立『菜單價目表』

(A) 系統功能：Ex5_5_1.java

首先，請您幫他建立一套『菜單價目表』(Menu)，並允許列印菜單或張貼於網路上。預計公布菜單如下所示：

菜　名	價　格	卡洛里
蠔油香菇	300	800
蒜泥白肉	250	500

筍干扣肉	320	700
五味花枝	200	600
紅燒鮮魚	400	600
麻婆豆腐	100	200
白玉珍丸	150	300
快炒青菜	100	150
青菜豆腐湯	100	100
水晶蝦餃	200	300
紅油炒手	200	400
蜜汁叉燒酥	200	400
藕斷絲連	200	300
龍門炒米粉	300	500
瑤柱炒飯	300	500
龍蝦鮮麵	400	400
白飯	20	100
足料觀湯	500	800
香菜皮蛋湯	200	300
酸辣海鮮湯	200	300
清湯	20	10
南瓜西米露	150	200

期望製作出來的結果如下：

```
D:\Java2_book\chap5\Ex5_5>javac Ex5_5_1.java

D:\Java2_book\chap5\Ex5_5>java Ex5_5_1
   菜單          價格      卡路里
蠔油香菇         300       800
蒜泥白肉         250       500
筍干扣肉         320       700
五味花枝         200       600
```

紅燒鮮魚	400	600
麻婆豆腐	100	200
白玉珍丸	150	300
快炒青菜	100	150
青菜豆腐湯	100	100
水晶蝦餃	200	300
紅油炒手	200	400
蜜汁叉燒酥	200	400
藕斷絲連	200	300
龍門炒米粉	300	500
瑤柱炒飯	300	500
龍蝦鮮麵	400	400
白　　飯	20	100
足料觀湯	500	800
香菜皮蛋湯	200	300
酸辣海鮮湯	200	300
清　　湯	20	10
南瓜西米露	150	200

(B) 系統分析

首先我們針對菜單內項目做一個 Item 類別，在主類別內宣告一個以 Item 物件為元素的 Menu 陣列，並預留 100 個空間。其實 Menu 菜單是由後台輸入的，為了方便驗證系統的正確性，我們給予初值。系統將菜單初值(name[]、price[]、colary[]) 讀入後，再分項印出即可。

(C)程式範例

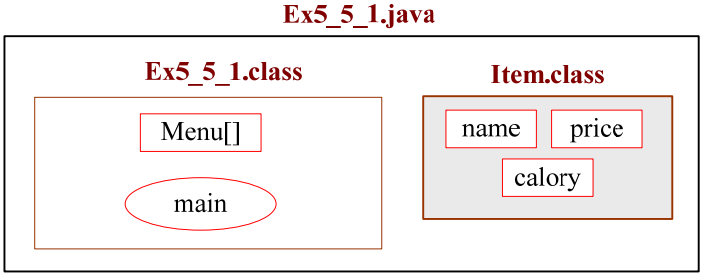

圖 5-19 Ex5_5_1 程式架構

```
01  //Ex5_5_1.java
02  /* 請您幫他建立一套『菜單價目表』( Menu ) ,
03
04      請製作一個菜單項目 Item，再產生 Menu 物件陣列
05      時立即並給予下列初值內容，再依照設計格式印出價目表 ) */
06
07
08  class Item {
09        String name;
10        int price;
11        int calory;
12  }
13
14  public class Ex5_5_1{
15          static Item[] Menu = new Item[100];
16          public static void main(String[] args){
17                  // 給予菜單初值
18                  String name[] = {"蠔油香菇","蒜泥白肉","筍干扣肉","五味花枝",
19                                   "紅燒鮮魚","麻婆豆腐","白玉珍丸","快炒青菜",
20                                   "青菜豆腐湯", "水晶蝦餃", "紅油炒手",
21                                   "蜜汁叉燒酥","藕斷絲連","龍門炒米粉",
22                                   "瑤柱炒飯", "龍蝦鮮麵","白       飯","足料觀湯",
23                                  "香菜皮蛋湯","酸辣海鮮湯", "清       湯",
24                                  "南瓜西米露"};
25                  int[] price = {300,250,320,200,400,100,150,100,100,
26                                 200, 200,200,200,300, 300,400,20,500,
27                                 200,200,20,150};
28                  int[] calory = {800,500,700,600,600,200,300,150, 100,
29                                 300,400,400,300,500, 500,400,100,800,
30                                 300,300,10,200};
31
32                  for (int i=0; i<name.length; i++) {
33                          Menu[i] = new Item();
34                          Menu[i].name = name[i];
35                          Menu[i].price = price[i];
36                          Menu[i].calory = calory[i];
37                  }
38                  System.out.printf("    菜單\t\t 價格\t 卡路里\n");
39                  for (int i=0; i<name.length; i++) {
40                        System.out.printf("%s\t", Menu[i].name);
41
42
43
44
```

```
45                      System.out.printf("%d\t", Menu[i].price);
46                      System.out.printf("%s\t", Menu[i].calory);
                    System.out.printf("\n");
                }
            }
        }
```

5-4-2 範例研討：建立『點菜系統』

(A) 系統功能：Ex5_5_2.java

雙方對『菜單價目表』感到滿意之後，接下來需製作『顧客點菜系統』，可以由顧客自行點菜或由服務生經手。顧客點菜過程中，會隨時累計消費金額與卡洛里，完成後，除了會列印點菜清單給客戶之外，也會列印一張菜單給廚房準備上菜。期望操作介面如下：

```
D:\Java2_book\chap5\Ex5_5>java Ex5_5_2
請輸入桌次 =>5                【輸入顧客桌次】
(1)蠔油香菇  (2)蒜泥白肉  (3)筍干扣肉  (4)五味花枝  (5)紅燒鮮魚
(6)麻婆豆腐  (7)白玉珍丸  (8)快炒青菜  (9)青菜豆腐湯  (10)水晶蝦餃
(11)紅油炒手  (12)蜜汁叉燒酥  (13)藕斷絲連  (14)龍門炒米粉  (15)瑤柱炒飯
(16)龍蝦鮮麵  (17)白      飯  (18)足料觀湯  (19)香菜皮蛋湯  (20)酸辣海鮮湯
(21)清      湯  (22)南瓜西米露  (0) 結束點菜
請輸入菜名的編號 =>1          【輸入菜單】
累計 300 元，800 卡洛里、請輸入菜名的編號 =>5        【輸入菜單】
累計 700 元，1400 卡洛里、請輸入菜名的編號 =>3       【輸入菜單】
累計 1020 元，2100 卡洛里、請輸入菜名的編號 => 20    【輸入菜單】
累計 1220 元，2400 卡洛里、請輸入菜名的編號 =>0      【結束點菜】
5 桌顧客菜單如下：
   菜名          價格     熱量
蠔油香菇         300      800
紅燒鮮魚         400      600
筍干扣肉         320      700
酸辣海鮮湯       200      300
總計             總價格:1220      總熱量:2400
```

廚房增加菜單如下：

5 桌次 蠔油香菇

5 桌次 紅燒鮮魚

5 桌次 筍干扣肉

5 桌次 酸辣海鮮湯

(B) 系統分析

我們引用 Ex5_5_1.java 繼續擴充。我們類別 Fare_item 來宣告客戶每點一樣菜的物件，並利用 Fare[] 物件陣列來儲存所有顧客所點的菜單。並宣告 Item_No 與 Fare_No 兩個類別變數來記錄目前菜單與客人點菜的數目有多寡。

(C) 程式範例

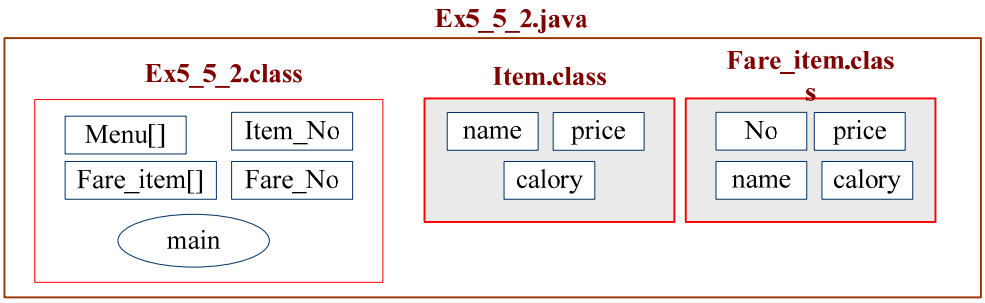

圖 5-20 Ex5_5_2 程式架構

```
01 //Ex5_5_2.java
02 /* 延續上題，請建立一套點餐系統，系統允許客戶以選單方式，選擇所喜歡的菜色。客戶
03  * 點餐之前需輸入所在桌次，再依序選擇菜單，完畢後系統會印出菜單、總價與本桌包含
04  * 熱量多寡。*/
05
06
07 import java.util.*;
08 class Item {
09       String name;              // 菜單項目名稱
10       int price;                // 單價
11
12       int calory;               // 熱量
13 }
14 class Fare_item{
15       int No;                    // 桌次號碼
16
17       String name;               // 菜名
```

```
18      int price;                  // 價格
19
20      int calory;                 // 熱量
21  }
22  public class Ex5_5_2{
23      static Item[] Menu = new Item[100];          // 菜單價目表
24
25      static int Item_No = 0;                      // 菜單項目數量
26
27      static Fare_item[] Fare = new Fare_item[20]; // 顧客點菜項目
28
29      static int Fare_No = 0;                      // 顧客點菜項目數量
30      public static void main(String[] args){
31          Scanner keyin = new Scanner(System.in);
32          // 給予菜單初值
33
34          String name[] = {"蠔油香菇","蒜泥白肉","筍干扣肉","五味花枝",
35                           "紅燒鮮魚","麻婆豆腐","白玉珍丸","快炒青菜",
36                           "青菜豆腐湯", "水晶蝦餃","紅油炒手","蜜汁叉燒酥",
37                           "藕斷絲連","龍門炒米粉","瑤柱炒飯","龍蝦鮮麵",
38                           "白　　飯","足料觀湯","香菜皮蛋湯","酸辣海鮮湯",
39                           "清　　湯","南瓜西米露"};
40          int[] price = {300,250,320,200,400,100,150,100,100,200,200,
41                         200,200,300,300,400,20,500,200,200,20,150};
42          int[] calory = {800,500,700,600,600,200,300,150,100,300,400,
43                          400,300,500,500,400,100,800,300,300,10,200};
44          for (int i=0; i<name.length; i++) {
45              Menu[i] = new Item();
46              Menu[i].name = name[i];
47              Menu[i].price = price[i];
48              Menu[i].calory = calory[i];
49          }
50          // 給予菜單初值結束、點菜作業開始
51
52       int table_No, count, item, sel;
53          int total_m=0, total_c=0;
54          System.out.printf("請輸入桌次 =>");
55          table_No = keyin.nextInt();
56          for(int i=0; i<Item_No; i++){
57              System.out.printf("(%d)%s    ", (i+1), Menu[i].name);
58              if ((i+1)%5 == 0)
59                  System.out.printf("\n");
60          }
61          System.out.printf("(0) 結束點菜 \n");
62
63
```

```
64              System.out.printf("請輸入菜名的編號 =>");
65              item = keyin.nextInt();
66              count = 0;
67              while(item != 0) {
68                      Fare[Fare_No] = new Fare_item();
69                      Fare[Fare_No].No = table_No;
70                      Fare[Fare_No].name = Menu[item-1].name;
71                      Fare[Fare_No].price = Menu[item-1].price;
72                      Fare[Fare_No].calory = Menu[item-1].calory;
73                      total_m = total_m + Fare[Fare_No].price;
74                      total_c = total_c + Fare[Fare_No].calory;
75                      Fare_No = Fare_No + 1;
76                      count = count + 1;
77                  System.out.printf("累計%d 元，%d 卡洛里、請輸入菜名的編號 =>",
78                                          total_m, total_c);
79                      item = keyin.nextInt();
80              }
81              System.out.printf("%d 桌顧客菜單如下：\n", table_No);
82              System.out.printf("    菜名\t\t 價格\t 熱量\n");
            for(int i=0;i< count;i++)
                    System.out.printf("%s\t%3d\t%3d\n",Fare[i].name,Fare[i].price,
                                        Fare[i].calory);
            System.out.printf("總計\t\t 總價格:%d\t 總熱量:%d\n",total_m,total_c);

             System.out.printf("廚房增加菜單如下：\n");
            for(int i=0;i<count;i++)
                    System.out.printf("%3d 桌次\t %s \n",Fare[i].No, Fare[i].name);
        }
    }
```

5-4-3 自我挑戰：建立『餐廳管理系統』

(A) 系統功能：PM5_3

經過 Ex5_5_1 與 Ex5_5_2 討論後，雙方都能接受此運作模式，接下來必須將它整合一系統，我們希望具有 7 個功能選項。

(1) 進入系統後出現 7 個功能選項，如下

```
D:\Java2_book\chap5\PM5_3>javac PM5_3.java

D:\Java2_book\chap5\PM5_3>java PM5_3
** 真健康美食餐廳 管理系統 **
(1) 匯入菜單項目
(2) 新增菜單項目
(3) 列印所有菜單
(4) 顧客點菜作業
(5) 會計收款作業
(6) 列印所有點菜
(7) 結 束 作 業
        請選擇輸入 =>
```

(2) 選擇匯入菜單並顯示其結果(選擇 1 再選 3)操作如下：

```
        請選擇輸入 =>1
** 真健康美食餐廳 管理系統 **
(1) 匯入菜單項目
(2) 新增菜單項目
(3) 列印所有菜單
(4) 顧客點菜作業
(5) 會計收款作業
(6) 列印所有點菜
(7) 結 束 作 業
        請選擇輸入 =>3
目前有：22
  菜單          價格    卡路里
蠔油香菇        300     800
蒜泥白肉        250     500
筍干扣肉        320     700
五味花枝        200     600
紅燒鮮魚        400     600
麻婆豆腐        100     200
```

```
白玉珍丸          150     300
快炒青菜          100     150
青菜豆腐湯        100     100
水晶蝦餃          200     300
紅油炒手          200     400
蜜汁叉燒酥        200     400
藕斷絲連          200     300
龍門炒米粉        300     500
瑤柱炒飯          300     500
龍蝦鮮麵          400     400
白      飯        20      100
足料觀湯          500     800
香菜皮蛋湯        200     300
酸辣海鮮湯        200     300
清      湯        20      10
南瓜西米露        150     200
```

(3) 選擇『新增菜單項目』再觀察其結果(選擇 2 與 3)的操作如下：

```
            請選擇輸入 =>2
請輸入菜單名稱 =>炒青菜
請輸入價格 =>150
請輸入熱量 =>100
** 真健康美食餐廳 管理系統 **
(1) 匯入菜單項目
(2) 新增菜單項目
(3) 列印所有菜單
(4) 顧客點菜作業
(5) 會計收款作業
(6) 列印所有點菜
(7) 結 束  作 業
            請選擇輸入 =>3
```

```
目前有：23
   菜單            價格      卡路里
蠔油香菇           300       800
蒜泥白肉           250       500
筍干扣肉           320       700
.....
南瓜西米露         150       200
炒青菜   150       100
```

(4) 選擇『顧客點菜作業』(選擇 4)，結果會出線顧客點菜明細表，以及給廚房的增加菜單，操作如下：

```
            請選擇輸入 =>4
請輸入桌次 =>12
(1)蠔油香菇  (2)蒜泥白肉  (3)筍干扣肉  (4)五味花枝  (5)紅燒鮮魚
(6)麻婆豆腐  (7)白玉珍丸  (8)快炒青菜  (9)青菜豆腐湯  (10)水晶蝦餃
(11)紅油炒手  (12)蜜汁叉燒酥  (13)藕斷絲連  (14)龍門炒米粉  (15)瑤柱炒飯
(16)龍蝦鮮麵  (17)白      飯  (18)足料觀湯  (19)香菜皮蛋湯  (20)酸辣海鮮湯
(21)清      湯  (22)南瓜西米露  (23)炒青菜  (0) 結束點菜
請輸入菜名的編號 =>1
累計 300 元，800 卡洛里、請輸入菜名的編號 =>3
累計 620 元，1500 卡洛里、請輸入菜名的編號 =>5
累計 1020 元，2100 卡洛里、請輸入菜名的編號 =>7
累計 1170 元，2400 卡洛里、請輸入菜名的編號 =>15
累計 1470 元，2900 卡洛里、請輸入菜名的編號 =>0
12 桌顧客菜單如下：
   菜名            價格      熱量
蠔油香菇           300       800
筍干扣肉           320       700
紅燒鮮魚           400       600
白玉珍丸           150       300
瑤柱炒飯           300       500
```

```
總計                總價格:1470        總熱量:2900

廚房增加菜單如下：
 12 桌次   蠔油香菇
 12 桌次   筍干扣肉
 12 桌次   紅燒鮮魚
 12 桌次   白玉珍丸
 12 桌次   瑤柱炒飯
```

(5) 選擇『會計收款作業』(選擇 5)，要求輸入桌號後，會出現明細表與應收金額，操作如下：

```
          請選擇輸入 =>5
請輸入買單桌號 =>12
菜名：蠔油香菇   單價：300
菜名：筍干扣肉   單價：320
菜名：紅燒鮮魚   單價：400
菜名：白玉珍丸   單價：150
菜名：瑤柱炒飯   單價：300
合     計 = 1470
```

(6) 選擇『列印所有點菜』(選擇 6)，則會出現到目前所有出菜清單，與總營業額多寡，操作如下：

```
          請選擇輸入 =>6
桌次       菜單項目
12         蠔油香菇
12         筍干扣肉
12         紅燒鮮魚
12         白玉珍丸
12         瑤柱炒飯
9          筍干扣肉
9          紅燒鮮魚
```

```
9          白玉珍丸
9          快炒青菜
9          青菜豆腐湯
9          水晶蝦餃
9          瑤柱炒飯
9          南瓜西米露
總共收入：3190
```

(B) 系統分析

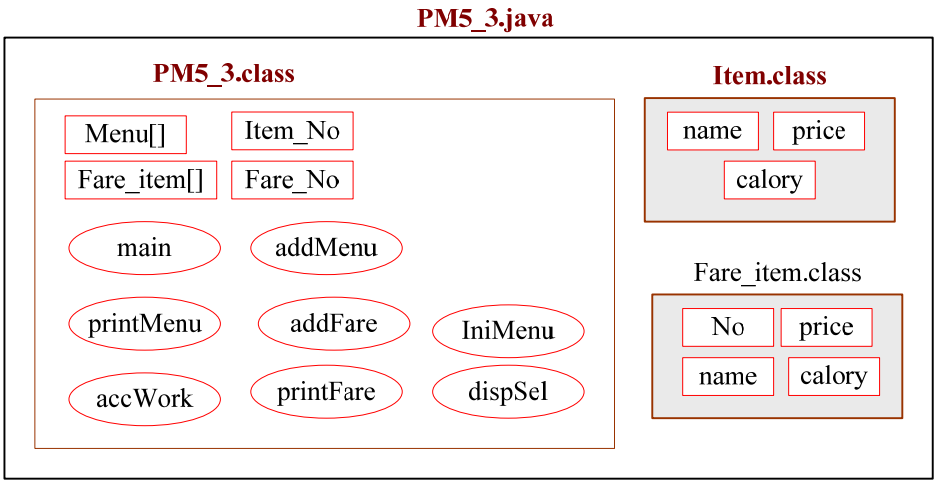

圖 5-21 PM5_3 程式架構

(C)製作提示

Items.java 程式範例如下：

```
01  //Items.java
02  /* 產生 Item 與 Fare_item 兩類別，以供產生物件陣列
03
04   * 經由 D:> javac Items.java 命令則產生
05   * Item.class 與 Fare_item.class 兩類別 */
06
07
08  //菜單 Menu 陣列內各元素的物件
09  class Item {
10        String name;
11        int price;
12        int calory;
13  }
14
```

```
15  // 點菜 Fare 陣列內各元素的物件
16
17  class Fare_item{
18      int No;
19      String name;
20      int price;
        int calory;
    }
```

PM5_3.java 程式片段如下：

```
01  ……
02  …..
03  import java.util.*;
04  public class PM5_3{
05      static Item[] Menu = new Item[100];
06      static int Item_No = 0;
07      static Fare_item[] Fare = new Fare_item[20];
08      static int Fare_No = 0;
09      public static void main(String[] args){
10          Scanner keyin = new Scanner(System.in);
11          int sel;
12          dispSel();
13          sel = keyin.nextInt();
14          while (sel != 7) {
15              switch (sel) {
16                  case 1:
17                      IniMenu();
18                      break;
19                  case 2:
20                      addMenu();
21                      break;
22                  case 3:
23                      printMenu();
24                      break;
25                  case 4:
26                      addFare();
27                      break;
28                  case 5:
29                      accWork();
30                      break;
31                  case 6:
32                      printFare();
33                      break;
34                  default:
35                      System.out.printf("錯誤輸入 !! 請重新選擇\n");
36                      break;
```

```
37                      }
38                      dispSel();
39                      sel = keyin.nextInt();
40              }
41      }
42      static void addMenu() {
43              Scanner keyin = new Scanner(System.in);
44              Menu[Item_No] = new Item();
45              .....
46              .....
47              Item_No = Item_No + 1;
48      }
49      static void printMenu() {
50              System.out.printf("目前有：%d\n", Item_No);
51              .....
52              .....
53      }
54      static void addFare() {
55              Scanner keyin = new Scanner(System.in);
56              int table_No, count, item, sel;
57              int total_m=0, total_c=0;
58              System.out.printf("請輸入桌次 =>");
59              .....
60              ......
61      }
62      static void accWork() {
63              Scanner keyin = new Scanner(System.in);
64              int table_No;
65              System.out.printf("請輸入買單桌號 =>");
66              ......
67              ......
68      }
69      static void printFare() {
70              int count=0;
71              System.out.printf("桌次\t 菜單項目\n");
72              ......
73              ......
74      }
75      static void dispSel(){
76             System.out.printf("** 真健康美食餐廳 管理系統 **\n");
77              .....
78              .....
79      }
80
81
82
```

```
83          // 給予菜單初值
84
85          static void IniMenu() {
96                  .....
87                  .....
88                  for (int i=0; i<name.length; i++) {
89                          Menu[i] = new Item();
90                          Menu[i].name = name[i];
91                          Menu[i].price = price[i];
92                          Menu[i].calory = calory[i];
93                          Item_No = Item_No + 1;
94                  }
            }
}
```