

學習日誌

給點建議

新增學習日誌

日期：

2025/10/17

學習目標：

今天的目標是加強我對 Java 物件導向程式設計的理解，特別是類別繼承、抽象類別、多型以及建構子使用的概念，我希望能學會如何設計類別之間的關係、正確使用 `super()` 呼叫父類別建構子，並練習撰寫能完整運作的程式。

學習過程：

今天我主要做了很多練習，包含 `Vehicle` 與 `Car` 的繼承練習、`Shape` 與 `Rectangle` 的抽象類別練習，以及 `Employee` 系列與旅遊訂票系統的設計與實作，最後也練習撰寫口語化報告，整個過程讓我把程式與報告結合起來完成。

遇到的挑戰：

在練習中遇到的主要困難是如何處理抽象類別不能直接實例化的問題，像在 `Employee` 類別中，為了建立基本員工 `e1`，我最初用匿名內部類別的方式解決，但後來意識到如果建立一個 `GeneralEmployee` 子類別會更直觀，也更符合物件導向設計。

學到的收穫：

我學會了 Java 中 `super()` 的使用方法，以及如何讓子類別建構子先呼叫父類別建構子，確保父類別屬性初始化完成。我也加深對抽象類別與多型的理解，知道如何利用抽象類別提供共同介面，而不同子類別可以實作不同邏輯。

反思與改進：

對於訂單系統，我會更早規劃好資料結構，確保訂單管理與確認號碼生成邏輯簡單且不易出錯。

建構子與繼承 (繼承機制:第 3 題)

120.125.80.92:8088/task_answer_test

批改完畢

回首頁

分數: 96
批閱結果:

錯誤個數: 1

程式正確性分析:
學生程式和題目要求相比, 大致正確, 但仍有細節未完全符合:

主要錯誤:
1. 介紹方法輸出
學生直接在 `Vehicle.introduce()` 印出 "Car model: Tesla.", 這行為本身沒有錯, 但題目期望輸出是:
`Vehicle is created`
`Car model: Tesla.`

學生程式只有先 `new Car("Tesla")`, 在 `Vehicle` 的建構子會印 "Vehicle is created", 然後呼叫 `introduce` 方法再印 `Car model`, 這一點正確。

2. `Vehicle` 類別應不須記錄 `model` 嗎?
題目敘述「`Vehicle` 具備專屬的 `model(String)` 資料」, 因此 `Vehicle` 內有 `String model` 是正確設計。

3. 卡在命名上 (小問題):
雖然題目測算是 `c.introduce()`, 但學生程式物件命名為 `car`, 然後呼叫 `car.introduce()`。
雖不影響功能, 但語義可以更一致。

4. 報告內容幾乎完全正確。
但部分用詞有點偏書面並高度符合AI回答習慣, 偏向AI生成內容, 建議可以用更口語化筆觸。

整體來說, 程式功能已完全正確, 物件導向設計正確, 理解 `super` 正確, 唯物件命名和報告內容評分時略有扣分。

建議:

1. 建議將 `Car` 物件命名為 `c`, 以完全符合題目指示。`Main.main` 應寫成:

```
Car c = new Car("Tesla");  
c.introduce();
```


2. 報告建議用更口語化的陳述方式, 強調個人理解與體會; 不要僅限於定義型的描述以避免被懷疑AI抄襲。

3. 整體架構可以不用再修正, 核心物件導向的設計已符合題目問題。

多層繼承與多型 (繼承機制:第 4 題)

120.125.80.92:8088/task_ansv

+

← → ↺ 🏠

⚠ 不安全 120.125.80.92:8088/task_answer_test ☆ 🔒 📄 ⬇ 🍪 ⋮

批改完畢

回首頁

分數: 100
批閱結果:

1. 程式錯誤個數: 0個明顯錯誤。學生依照物件導向原則設計, 正確使用抽象類別及方法, 且主要功能 (多型、覆寫) 均正確。

2. 正確性分析:

- (a) Shape正確設為抽象類別, getArea()為抽象方法。
- (b) Rectangle繼承Shape, 正確覆寫getArea()。
- (c) 屬性宣告、建構子都正確符合Java慣例, 並且都加上存取修飾。
- (d) 主程式多型宣告及測試正確。
- (e) 報告內容清楚說明Shape為何要設抽象類別 (觀念正確、無抄襲疑慮)。

建議:

建議提升Main、Rectangle和Shape三個類別分開存放於對應檔案 (若在專案中)。目前已正確加上public及存取修飾。

問題描述:

請建立 Shape 類別, 並在 Shape 中定義 getArea() 方法。
建立Rectangle 類別繼承 Shape, 在 Rectangle 中覆寫該方法來計算面積。
最後, 說明你的設計重點:
1. Shape為何要設計成抽象類別, 如果不, 會有甚麼問題:
另外, 設計Main類別進行下列測試:
s 是Shape型態, 是Rectangle物件, 長寬分別是5, 10
System.out.println("Area: " + s.getArea());

輸出:
Area: 50.0

設計 Manager 類別繼承 Employee(繼承機制:第 5 題)

120.125.80.92:8088/task_ansv

120.125.80.92:8088/task_answer_test

批改完畢

回首頁

分數: 100
批閱結果:

經檢查，學生程式設計整體上正確完成了物件導向設計，也完全符合題目需求。

錯誤/問題點說明：

- 類別命名和測試用例略為差異：
 - 原題直接建立 `Employee e1 = new Employee(...)`，學生由於 `Employee` 是抽象類別，正確地建立了一個
 - 測試程式的 `getName()` 方法取代直接存取 `name` 欄位（題目原本用 `e1.name`），這符合封裝精神，正確。
- 程式設計完全符合物件導向四大原則（封裝、繼承、多型、抽象化），且支援多型與開閉原則。
- 註解與說明雖不豐富，邏輯清楚易懂。
- 報告部分能正確理解抽象類與多型的關聯以及開閉原則，內容語意清楚、非AI模式制式語句。

錯誤個數：0（僅有小幅命名差異，不影響設計正確性）

綜上，程式與報告皆表現優異，幾無扣分空間。

建議:

建議可將 `GeneralEmployee` 命名為 `Employee` 或 `BasicEmployee`，使其名稱與 `Employee` 概念接近。若需與題

問題描述:

設計 `Employee` 類別，包含姓名與基本薪資，並有方法 `getSalary()`。

`Manager` 類別繼承 `Employee`，在薪資中額外加上管理津貼。

`Sales` 類別繼承 `Employee`，在薪資中額外加上業績獎金。

主程式建立不同員工物件，並輸出他們的薪資。

最後，說明你的設計重點：

- 抽象類別與多型有何關係:
- 本設計是否合乎物件導向的開閉原則(Open-Close Principle)

如何計算圓周率 (封裝\小考練習第 13 題)

120.125.80.92:8088/task_ansv

120.125.80.92:8088/task_answer_test

批改完畢

回首頁

分數: 100
批閱結果:

本題的學生程式在物件導向設計、正確性與可讀性上有高度的水準。分析如下：

錯誤個數：0（邏輯錯誤、語法錯誤皆無，OOP設計原則非常良好）

一、正確性說明：

1. 類別拆封完整：Point 負責座標、Shape 為抽象基類、QuarterCircle 為具體四分之一圓，MonteCarloSimulator 採用聚合方式計算隨機落點，主程式 Main 則作為測試。
2. Shape 抽象類設計良好，有便於擴充其他圖形的彈性。
3. MonteCarloSimulator 負責運算，封裝與 OOP 應用得宜。
4. estimatePi、getInsideRatio、getLastPointsInside 等方法，易於調用與擴充。
5. 功能上完全符合題目描述與測試主類別的規格要求。

二、報告內容：

報告回答有涵蓋類別分工、繼承與擴充的優點，理由敘述合理且已經具體對應程式設計。

三、整體結論：

程式架構非常清晰，每個類別職責明確，且保留高度可維護與擴充性。符合物件導向良好設計原則。對照原始需求，功能完整。報告內容合乎規範，未見AI抄襲痕跡。

建議:

1. Point 類型可以增加 equals/hashCode，雖然本題用不到，但與其他設計兼容性更高。
2. MonteCarloSimulator 構造函數可以考慮支援 Random 傳入，增加可測試性與可控性。
3. QuarterCircle 實作中，可以將座標區間 [0,1] 抽象為成員常數或自定義 Region 類，進一步通用化。
4. detailedEstimation 方法在本題用不到，後續可據需求擇一保留。

問題描述:

已知邊長1的正方形，面積是1。以其邊長為半徑，在正方形內對角點畫一個弧，可得到面積是 $\pi/4$ 的四分之一圓。

撰寫程式，模擬平均撒N個點在正方形內，然後統計落在圓內的點除以N所得的值再乘上4，結果是多少？