第二章　文獻探討

本研究以問題解決的觀點與自我解釋策略，做為程式語言教學活動設計的理論基礎。因此分別對問題解決的理論基礎、自我解釋策略、程式語言學習等相關研究進行探討，以做為本研究設計時的參考，最後則總結文獻探討之內容並提陳具體之結論與研究目標。

第一節　問題解決的理論基礎

每一天，我們都會遇到許多問題，有的問題很簡單，像是「今天中餐要吃什麼？」，有的問題卻很困難，例如「什麼是我的人生目標？」。由於問題解決在日常生活中的重要性，因此許多學者專家們對問題解決的本質及歷程提出各自的想法與觀點，本節將就問題解決的意義，問題解決的歷程模型，及如何促進問題解決學習作深入的探討。

壹、　問題解決的意義

Chi 和 Glaser(1985)認為「問題解決」是指當學習者無法自然地找出問題的解決辦法時，嘗試以各種可能方式完成最終目標的一種歷程；在學習者遇到問題時，會先從自已的舊經驗中尋找可能的解題方式，在不斷的嘗試之後，成功的連結會將問題所涉及的新概念(concept)及解題所用到的法則(rule)進一步地轉變成高層次的技能。Kahney(1986)則認為「問題解決」是學習者為了適應情境的需要，運用過去所習得的知識技能以尋求解答的過程；在解決問題中，學習

者經由探索、分析、搜尋等不同的方式獲得與問題相關的資訊，並企圖解答疑問，消除困惑，直到達到目標為止。由於問題代表著一種未知而陌生的情境，因此 Hunt(1994)將「問題解決」視為一種高層次的心智活動模型。綜合以上的觀點，問題解決可說是找尋問題之有效解決方案或可能結果的過程，在這個過程中個人會針對問題情境選擇適當的先備知識及經驗，為完成該目標不斷找尋可能的解決路徑，並對相關的資訊進行重組以發展出可行之方法。由於在問題解決的過程中，學習者原本習得的舊法則將轉變成更高層次的技能，因此問題解決可以提高學習者的知識、能力並提升認知結構之層級。

在科學教育或是資訊教育的研究上，培養並提昇學生的問題解決能力一直是一個很重要的目標(行政院國家科學委員會科學教育發展處，民 85)，而我國九年一課程改革中也指出：培養獨立思考與反省的習慣，有系統地研判問題，並能有效解決問題和衝突為現階段學校教育的目標(教育部，民 87)。因此「問題解決」在學習中占有舉足輕重的地位。

貳、 問題解決的歷程模型

問題解決是一種複雜的心智歷程，從問題的產生到解決，牽涉到許多高層次的心理歷程，因此從認知心理學的觀點探討問題解決的歷程模型成為眾多學者的研究方向；此外，問題解決歷程模型的建立也有助於了解問題解決在人類心智中運作的情形，以下整理各家學者所提出的歷程模型。

一、Dewey 的問題解決歷程模型

美國的教育學家 Dewey(1910)視問題解決為有意識且深思熟慮的心智過程，此過程會自然地伴隨著一連串的心理活動。Dewey 認為問題解決有五個基本的心智步驟：

1. 呈現問題：首先問題解決者要先能辨識出已存在的問題，才能進行問題解決。由於辨識問題需要具備與問題相關的先備知識且需花費相當長的時間，因此是相當難的一個步驟。
2. 定義問題：接著問題解決者要確認問題的本質以及問題的解答存在那些重要的限制。
3. 形成假設：在此步驟問題解決者要提出一個或多個合理的問題解決方案。
4. 測驗假設：然後在這些解題方案中測驗出那些方案是可行的，並找出那一個是最可能成功的解題方案。
5. 選擇最佳的假設：最後成功的找出一個最佳的假設，並了解每一種解決辦法的相對優勢及缺點。

二、Newell 和 Simon 的問題解決歷程模型

資訊處理理論(Information processing theory)在認知心理學中占有舉足輕重的地位(Banikowski & Mehring, 1999)，Newell 和 Simon(1972)即以資訊處理模式分析人類的問題解決過程，並提出了「一般性問題解決模型」(general problem

solver)。

Newell 和 Simon 認為整個問題解決的過程包含了問題初始狀態和最後的目標狀態，這些狀態組合起來就稱為「問題空間」(problem space)，在問題空間及資訊處理理論的前提下，一般的問題解決過程可分成「了解問題」以及「找尋解決方法」兩個階段：

1. 了解問題：在這個階段中，問題解決者將問題轉換成心智表徵(mental representation)，以命題(propositions)，圖像(image)等方式置放在工作記憶體(working memory)中，表徵以內在的形式存放在內心中，或以外在的方式呈現；例如：解題者可以將表徵中的相關資訊記錄在紙上、寫在白板上、甚至儲存在電腦之中。
2. 找尋解決方法：接著這些儲放在工作記憶體的資訊開始活化個體長期記憶體(long-term memory)中的知識，使問題解決者得以提取與問題相關的知識，並從中挑選解決問題的策略與方法，當個體找不到有效的問題解決方法時，就會開始修正起初的問題表徵。解題者若能成功解決問題，則表示內在的問題表徵足以代表問題本身，因此問題表徵就能進一步深植於長期記憶體中，成為個體的新知識。

三、Bransford 和 Stein 的問題解決歷程模型

Bransford 和 Stein(1984)延續之前的研究，進一步將問題解決的過程分成了「確認問題(identifying)、擬定解決計劃(defining)、蒐集與探索資訊(exploring)、

實際行動(acting)、及評估結果(looking back)」等五個步驟，也就是所謂的 IDEAL 模型。

四、Gick 的問題解決歷程模型

Gick(1986)綜合了這些問題解決模型提出了一個精簡的模型，模型內容如圖 1 所示：

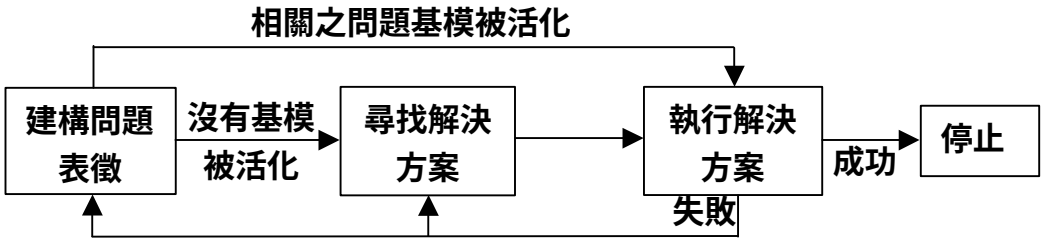


圖 1 問題解決的精簡模型

1. 建構問題表徵： 在問題解決的第一階段，解題者首先要建構出問題的心智表徵，例如：目前的問題是什麼？在怎樣的情境之下發生了這樣的問題？這是屬於那一類的問題？這個過程涉及了對問題的解析及分類。
2. 尋找解決方案：根據問題表徵，從經驗及相關知識中找尋可能的解題方案，或是透過創造力引發出一個具創意的解決方式。
3. 執行解決方案：接下來解題者開始實行並測試這個解題方案，假如此方法行不通的話，解題者將會修改其原先的問題表徵或尋找其它可行的解題方法。這個過程會不斷持續，直到找到一個成功的解決方案為止。值得注意的是：此問題解決模型並不強調解題者要找出多個解題方案，因為只要一個解法能成功地解決問題，整個問題解決歷程便宣告結束。

五、Greenfield 的問題解決歷程模型

Greenfield(1987)則歸納了以往的問題解決研究，提出三項與問題解決有關的規則與策略：

1. 問題解決的一般規則：全盤了解狀況暫不下判斷、應用思考模式或策略、改變表徵系統、提出適切問題、以及抱持懷疑心等。

2. 問題解決的策略步驟：

（1） 界定問題

（2） 構思(分析問題Ǧ搜集資料)

（3） 計畫(提出解決方案)

（4） 執行計畫(嘗試解決)

（5） 檢討。

3. 研擬問題解決過程中，準備階段和執行階段的策略：

（1） 準備階段：(A)區別相關與無關的資訊(B)作圖分析問題(C)組織歸納已有之資料。

（2） 執行階段：(A)找出與問題解決有關事項(B)對問題解決的相關事項進行排序(C)考驗對立假設(D)推論(E)由果溯因。

参、 促進問題解決學習的相關教學設計

在了解問題解決的認知歷程之後，接下來可從學習歷程探討如何以適當的教學設計促進問題解決的學習成效。

學習通常是為了要理解某些知識或是獲取某種技能。Anderson(1987)認為為了要理解新知識，學習者必需將外在的資訊加以編碼(encode)，這稱為「知識獲取階段」(knowledge acquisition)；為了將知識獲取階段的知識應用到實際的問題解決上，學習者必需將編碼後的陳述性知識編譯(compile)成相關的程序或技能，這稱為「問題解決階段」(problem solving)。整個知識學習歷程如圖 2 所示。

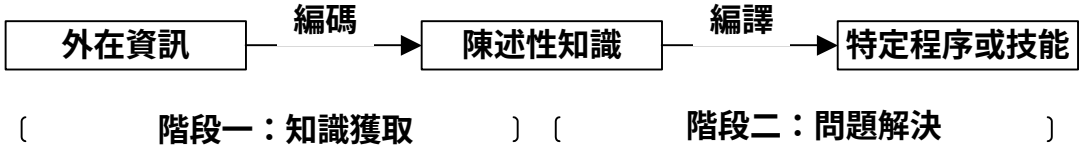


圖 2 知識學習歷程

在知識獲取階段，學習者直接由教學中獲取初步的陳述性知識。學習者根據先備知識將教材中的文章及範例轉化成心智表徵，這些表徵包括了特定領域的事實性概念及原則，並以陳述性知識的形態儲存在學習者的長期記憶體中。

在問題解決階段，學習者會根據問題的種類及形態，從長期記憶體中尋找可能的解決程序或相關技能，然而這樣的問題解決過程可能會因為找不到合適的解題程序而受到阻礙，這時，就要從陳述性知識中，找尋相關的解題概念或原則，藉由陳述性知識的運作解決所遭遇的問題，如此一來，在知識獲取階段所初步獲取的知識便會緩慢地轉變成更具效率的解題程序及技能。

從 Anderson(1987)的觀點可以得知：問題解決階段的學習成效取決於知識獲取階段的學習成效，因此若能藉由提升知識編碼效能來增加陳述性知識的獲

取量，或是藉由提升知識編譯效能來增加問題解決技能的產生，將能有效地促進學習者在問題解決方面的學習成效。

以下即詳述促進問題解決學習的相關教學設計。第一至四個教學設計在於促進陳述性知識的取得，也就是著重在知識的獲取階段，第五至七個教學設計在於促進問題解決技能的取得，也就是著重在問題解決階段。

一、複習相關之先備知識

在問題解決的過程中，學習者必需確認，選擇，並且應用相關的先備知識來進行解題(Dochy, 1995)。因此可利用以下三種方式補強學習者的先備知識：

1. 在進行問題解決教學之前, 提供補救課程讓學習者有機會複習與問題相關的概念，原則，及解題流程。
2. 在問題解決教學前進行前測，針對先備知識不足的學習者提供補救教學。
3. 將相關的補救課程與問題解決課程進行整合, 當學習者因為先備知識不足而產生學習的障礙時即可進行先備知識的輔助學習。

二、提供相關的概念模型

良好的概念模型可以藉由視覺化的方式，將教材所要呈現的知識本質、概念要素、原則間的關係、以及範例問題的狀態，以學習者熟悉的方式加以詳細呈現，讓教材和範例問題變的清晰易懂(Mayer, 1989)。概念模型的功用在於促進心智模型(mental model)的建立，進而降低學習者在將外在資訊轉換成內部心

智表徵時的認知負載；另一個解釋是因為概念模型將相關的知識明確地以結構化的方式加以呈現，減少初學者與專家在問題解決認知結構上的差距，進而促進了問題解決的成效。

三、提供演練範例

演練範例(worked example)包含了問題的解題過程以及專家在解題時的思考流程，其主要目的在於幫助學習者從範例學習中建立相關的問題解決技能。許多的研究均指出：演練範例的學習較實際解題更能提升問題解決的學習成效(Copper & Sweller, 1987; Sweller & Cooper, 1985)，這是因為演練範例可以幫助學習者建立合適的問題基模(problem schema)，並根據基模的本質或解法的相似性將不同的演練範例進行分類，當學習者遇到陌生的問題時就可以尋找合適的問題基模，並以類推的方式建構出適當之解法(Sweller & Cooper, 1985)。演練範例藉由提供類比問題(analogical problems)的方式，補足了學習者所缺乏的解題經驗，替問題解決提供了一個相當有用的學習鷹架。

四、進行自我解釋策略的訓練

雖然演練範例為問題解決提供了許多的類比問題，但是並不是所有的學習者都能掌握演練範例的精，並建構出完整的問題基模基模不足以掌握到問題的本質時，就只能解決和範例相同的問題，在面對新的問題時便一籌莫展。因此若能讓學習者主動的解釋範例，建立新舊知識的連結，就能建立出較完善的問題基模，並掌握其解題的程序。過去的研究發現學習表

現較好的學生，對演練範例會有較多的自我解釋，由於自我解釋的功用在於幫助學習者將新的知識與先備知識進行聯結，因而能提高演練範例學習的效果，進而將範例問題成功的類比到新的問題上(Reder, Charney & Morgan, 1986; Chi, et . al., 1989)。

五、提供練習問題

由於不同領域的問題需要不同的解題程序及技能，因此單單透過演練範例，只能建立學習者的問題基模，在缺乏練習的情況下，學習者無法有效且快速的利用問題基模進行解題。解題程序的自動化是一個相當緩慢的過程，並且需要龐大的練習，因此對於初學者而言，不只要提供演練範例，還要給予充份的練習，如此才有可能幫助問題基模的建立並促進問題基模對新問題的學習遷移(Cooper & Sweller, 1987)。

六、將問題分割成子問題尋找問題解決方案

輔助學習者將原本的問題分割成子問題可以增加解題時的效率，由於子問題的難度較低也容易解決，因此可以針對各個子問題提供相關的解題提示或是解題樣板，以促進問題解決成效(Fay,1990)。

七、建立對一般性問題解決策略的認知

King(1991)讓小學五年級的學生進行問題解決的學習，結果顯示，有提供提示卡進行問題解決策略學習的學生，其學習成效較沒有提供的學習成效來的優異。Delclos 和 Harrington(1991)將五年級和六年級學生分成三組，一組接受

問題解決及後設認知的訓練，一組接受問題解決的訓練，第三組沒有接受任何訓練；結果顯示，接受問題解決及後設認知綜合訓練的學生表現的比其它兩組好，而接受問題解決訓練的學生則表現的比控制組好。這些研究顯示出教導初學者一般性問題解決策略可以促進對問題解決的學習成效，這是因為學習者可藉由對問題解決歷程的認知，了解問題解決大致上的順序及相關步驟，使其有機會成為較佳的問題解決者。

第二節　自我解釋策略的相關研究

壹、 自我解釋的定義

過去的研究認為將教材編碼為陳述性知識的過程只是簡單的知識儲存動作，而將陳述性知識編譯成問題解決技能的過程則較為困難，因此大部分學者著重於問題解決的歷程研究(Newell & Simon, 1972; Ross, 1980; Pirolli & Anderson, 1985)。然而從 Anderson 的學習歷程模型可以看出陳述性知識的品質與數量對於問題解決的成效有相當程度的影響，由於過去有許多演練範例的相關研究，因此有我們可以針對演練範例的編碼歷程做進一步探討。

由於教材中的文章對學習者而言可能太過深奧，因此透過演練範例可以將教材中抽象的概念以具體的範例加以呈現，當學習者能夠充份的了解範例並從中歸納出不同問題的共通原則時，即代表學習者在初步的知識獲取階段建構了較為完整的問題基模，亦產生了較高層次的陳述性知識。為了進一步了解個別差異對知識編碼的影響，必須針對個別學生的自我解釋進行深入分析。

Chi, Bassok, Lewis, Reimann 和 Glaser(1989)認為自我解釋是指學習者在閱讀文章或演練範例的過程中，為澄清或補充句子的敘述所提出的推論。不論這些推論的正確或完整與否，都算是自我解釋的一部分。在進行自我解釋的同時，學習者必須監控自我的理解程度，然後根據先備知識以及外在知識進行自我推論以克服理解上的問題。

為了探討自我解釋對於問題解決的學習影響，Chi(1989)以物理學為學科領

域，要求大學生在進行力學範例學習的同時，以放聲思考(think aloud)的方式讓研究人員了解學習者內部的自我解釋歷程，並根據問題解決的成效，將受試者區分成好的解題者與差的解題者。其研究結果發現：(1)在研讀範例的過程中，好的解題者比差的解題者產生更多的自我解釋。(2)好的解題者比差的解題者花更多時間於研讀範例上(3)好的解題者能從範例中歸納出原則，因此其自我解釋較能涉及範例的深層意義，而差的解題者之自我解釋通常只涉及文字的表面語義。(4)好的解題者能進行學習監控並察覺出學習過程中的誤解，然而差的解題者卻不然。(5)好的解題者因為有明確的目標，因此只引用範例中與問題相關的程序，但是差的解題者卻引用到範例中的絕大部分。

自我解釋有助於範例學習的原因在於自我解釋是一種自發性的自我產出活動，這與外在所提供的解釋或說明大不相同。此種學習效果稱為學習的產出作用(generation effect)，也就是說學習者自我產生的學習內容(例如上課所作的筆記或是為了完成作業所進行的討論)較外在所提供的學習內容來的更容易吸收及記憶(Hirshman & Bjork, 1988)。

然而若從廣義的觀點來看待自我解釋，「解釋」一詞可以是指為了理解某些事物所進行的所有活動，因此自我解釋不僅包含學習者在學習新知識所進行的推論，亦包含在問題解決過程中學習者對問題的澄清以及在進行解題活動時所持之理由(Neuman, Leibowitz & Schwarz, 2000)。從以上的定義可以得知自我解釋可視為一個知識建構的學習活動：新知識的學習必需建構在原有的知識基

礎上。如果把學習視為一種建構的歷程，那麼自我解釋則提供了學習者自我建構的機會。

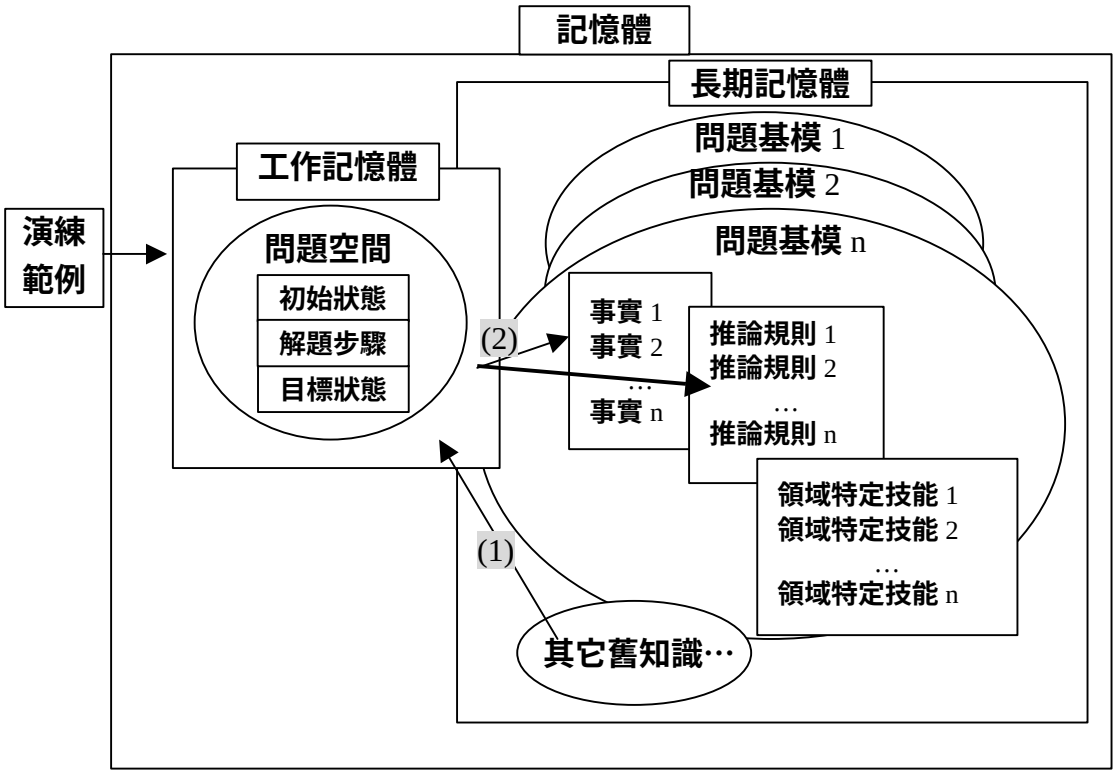
貳、 自我解釋對學習的影響

自我解釋對學習的影響可以分成二部分來探討，第一部分為自我解釋在知識獲取階段的作用，第二部分為自我解釋在問題解決階段的作用，首先就自我解釋在知識獲取階段的作用進行探討。

一、自我解釋在知識獲取階段的作用

在知識獲取階段，為了克服演練範例的學習障礙(例如：範例說明不夠清楚)，學習者必需從現有的資訊來推斷出範例所要傳達的概念及原則，為了達到這個目的，學習者會整合外在資訊與自身的先備知識，來對範例中的各個動作進行解釋，藉以了解這些動作的程序及目的，並推斷出其適用的情況為何(Chi et . al., 1989)。

這些由學習者所產生的解釋，在結構上與範例的本質相互呼應，也就是說自我解釋可視為問題空間的建構過程，其目的在於找出範例問題中的初始狀態，目標狀態、並從中範例解題的解題步驟中建立出此類問題的問題基模(Recker & Pirolli, 1990)。因此如圖 3 所示，在學習者進行知識連結與重組的過程中，自我解釋可以幫助學習者建立完善的問題空間，進而促進新舊知識間的連結。



(1) 知識連結與重組
(2) 知識編碼

圖 3 自我解釋在知識獲取階段的作用

學習者在建構問題空間的過程可依資訊處理程度(process level)的不同，分為表面處理(surface processing)及深度處理(deep processing)兩種等級(Marton, 1986; Ferguson-Hessler, 1990; Chi et . al., 1994)。表面處理指的是：學生處理資訊的主要目的僅在於資訊重製及記憶，將範例中的概念說明及程序視為理所當然，例如：“既然範例這樣寫，應該就是用這個公式了“、“既然範例提到這個概念，我就把它背下來吧“…這類的自我解釋都屬於表面處理。深度處理指的是：學習者將範例中的資訊與先備知識相互連結，並依照屬性及類別將外在資訊加以重組，藉以找出隱藏於範例中的原則。例如：“這個解法不只適用於

這個範例問題，還可以用在幾何學中“、“根據我的經驗以及範例的說明，這個動作的目的在於找出資料中的平均數“…這類的自我解釋都屬於深度處理。

從過去的研究可以發現差的問題解決者傾向對外在資訊進行表面處理，而好的問題解決者則會將資訊進行重組以建立良好的問題基模，而問題基模所內涵的知識內容將有助於學習者解決類似的問題(Ferguson-Hessler, 1990)。

問題基模包含了事實性知識、推論規則兩種陳述性知識以及與解題相關的特定技能。學習者若能對演練範例進行深度處理，新知識就較容易和舊知識整合，有利於問題空間的建構；而問題空間建構的愈完善，就愈能幫助學習者推導出問題基模中的推論規則。反之，若學習者只對演練範例進行表面處理，就只能獲取事實性知識，不利推論規則的建立。由於自我解釋可以幫助學習者建立完善的問題空間，所以能夠促進學習者對外在資訊進行深度處理，增加推論規則的產生。因此如圖 3 所示，在知識的編碼過程中，自我解釋可以幫助學習者產生較多的推斷規則。

所謂的推論規則是學習者從範例中的概念及原則所推斷出來的特定規則，以 IF…THEN 的形態存於問題基模之中(Chi et . al., 1989)，例如：“IF 程式問題若有兩種可能的情況，而且我必須去判斷目前的狀況是那一種，THEN 我可以用 if 敘述來進行判斷“就屬於程式問題基模中的一個推論規則。推論規則一旦建立之後，就能藉由問題解決的過程將原屬陳述性知識的推論規則轉換成相關的解題程序或特定技能，因此自我解釋在知識獲取階段最重要的作用在

於促進推論規則的形成。從自我解釋的內容分析，可了解學習者建構出多少推論規則，推論規則愈多代表學習者對演練範例的了解程度愈高。

學習者在整個自我解釋的過程中不免會產生許多錯誤的自我解釋，然而這些錯誤的自我解釋亦提供了一個學習的機會。這是因為即使產生了錯誤的自我解釋，學習者依然會依序閱讀教材的各個部份；由於教材所呈現都是正確的資訊，使得後來的自我解釋推論與之前所產生的推論相互矛盾，因此提供學習者一個自我補救的機會(Chi et . al., 1989)。

二、自我解釋在問題解決階段的作用

自我解釋在知識獲取階段的作用一再地在許多學科領域上得到證實，例如物理學(Chi et . al., 1989; Ferguson-Hessler, 1990)、生物學(Chi et . al., 1994)、電腦科學(Recker & Pirolli, 1990; Pirolli & Brown, 1995; Reimann & Neubert, 2000)及數學(Renkl, 1997)，然而直到近年來，才有學者開始探討自我解釋在問題解決階段的作用(Neuman & Schwarz, 2000; Neuman, Leibowitz, & Schwarz, 2000)。

Neuman 和 Schwarz(2000)在其研究中，以代數為學科領域，要求大學生在進行代數解題的同時，以自我解釋的方式讓研究人員了解學習者在解題時的心智歷程。為了探討不同種類的自我解釋在解題時的影響，因而將自我解釋分成以下三種類型：

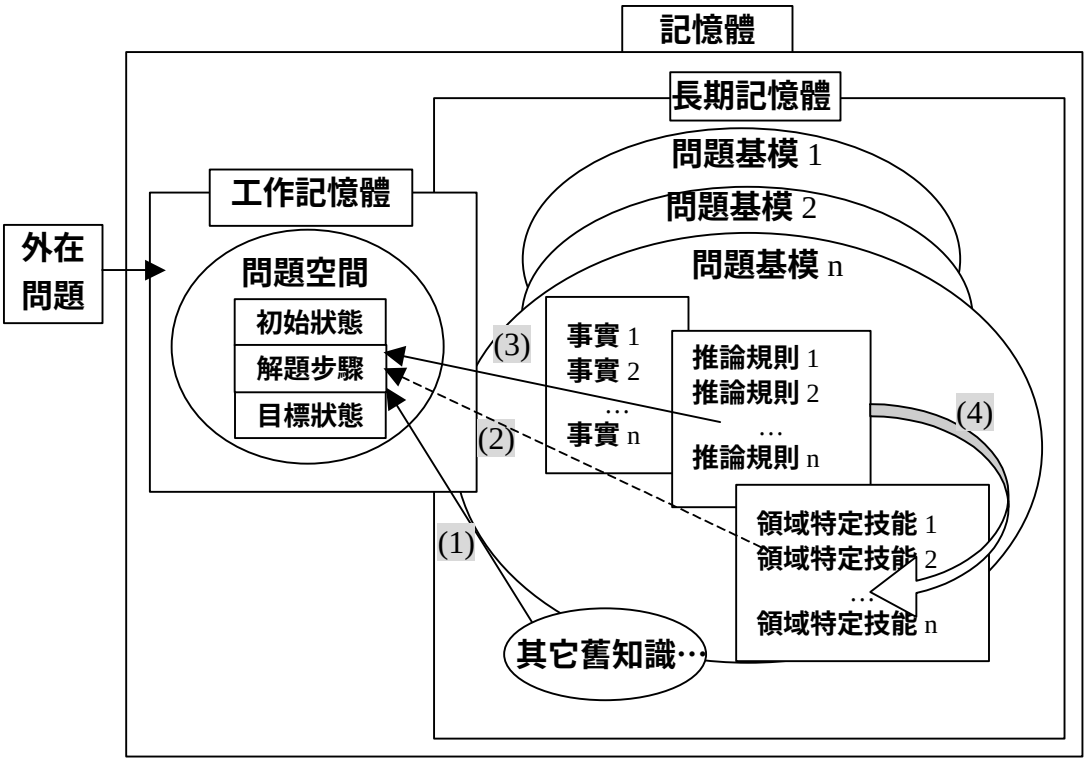
1. 分類：對句子進行語義上的分類，藉以將問題說明逐句的對映到問題空間中。

2. 推論：從現存的語句推論出另一各新的語句。

3. 定義：定義新的變數。

其結果發現：(1)雖然學習者對一般的問題解決程序相當熟悉，但是在解題程序的實行上仍有相當的困難(2)學習者在代數問題的了解階段(將問題說明轉換成問題空間)所產生的困難較數學解題階段來的大(3)「分類」型態的自我解釋可以幫助學習者將問題說明轉換成問題空間。因此自我解釋在問題解決階段可以幫助學習者對不同型態的問題外在表徵進行轉換，進行提升問題解決的成效。

所以如圖 4 所示，在問題解決的過程中，學習者首先要能了解問題，而自我解釋可幫助學習者建立初步的問題空間(問題初始狀態、目標狀態)，促進新舊知識間的連結。接著學習者會先從問題基模中尋找特定的解題程序或技能，學習者若缺乏特定解題技能，則需尋找相關的推論規則來解決問題。當推論規則被使用於問題解決時，就會逐漸的轉換成特定的程序或技能，由於自我解釋有利於推論規則的產生，進而促進特定技能的生成。



(1) 了解問題
(2) 尋找特定解題程序或技能
(3) 尋找與解題相關的推論規則
(4) 知識編譯

圖 4 自我解釋在問題解決階段的作用

參、 影響自我解釋成效的因素

在發現自我解釋與問題解決成效有顯著的關聯後，研究者開始探討那些因素會影響到自我解釋的成效。Chi 與 VanLenhn(1991)將 Chi 等人於 1989 年的實驗數據重新分析，發現解題優異的學習者與較差的學習者在先備知識測驗的成績(GPA 測驗)上並無顯著差異。Chi 等人(1994)在生物科的教學研究發現自我解釋提示組的學生在自我解釋的表現上和先備知識並與顯著關聯。

Renkl(1997)進一步探討先備知識、自我解釋與問題解決成效的關係，其研

究發現：(1)先備知識與問題解決成效產生關聯，但先備知識卻未與自我解釋產生關聯(2)低先備知識的學習者中，有一群人的問題解決成效相當不錯，這種學生傾向找出範例中各個動作的意義，並解釋其中的原則及相關目標。從以上的研究成果可以知道學習成效不完全取決於先備知識，藉由自我解釋活動的實行，亦能提升學習成效。

雖然自我解釋的效益與先備知識沒有明顯的關係，不過由於至少有 30%的自我解釋來自於新舊知識間的整合，因此一般性的知識可能在自我解釋中扮演著相當關鍵的角色(Chi et . al.,1994)，此外自我解釋的類型及模式亦可能影響自我解釋的學習成效。

一、自我解釋的類型

從過去的程式語言教學研究中發現下列四種類型的自我解釋可以促進學習成效(Pirolli & Recker, 1994; Bielaczyc, Pirolli, & Brown, 1995)：

1. 解釋文章中的主要概念：這種自我解釋用於闡明教材中個別概念間的相互關係，其功能在於幫助學習者回想教材中的相關術語、程式類別、及程式架構，因此可以啟發學習者設計出正確的程式架構。
2. 解釋程式型態及程式碼的意義：這種自我解釋用於判斷程式碼的執行目的及執行過程，進而幫助學習者在程式設計時將其想法轉換為對應的程式碼。
3. 解釋程式範例用到了將那些概念：這種自我解釋用於將教材中所提到

的抽象概念連結至具體的範例之中，進而增進學習者在問題解決時成功獲取相關資訊的機率。藉由這種知識連結，學習者較能理解特定程式碼的使用時機及重要性，因此對於程式設計時的程式碼撰寫有相當的幫助。

4. 解釋目前所採取的自我調節策略：這種自我解釋用於進行自我調節(self-regulation)，例如：監控自我對教材的理解度、說明閱讀時所欲使用的學習策略、說明如何採取適當步驟來達到學習目標。進行自我調節可以讓學習者察覺到在教材的理解上有那些缺失，並幫助學習者對各種策略進行判斷，以決定那一種學習策略較為適當。

Renkl(1997)則在數學機率教學中發現下列三種類型的自我解釋可以促進學習成效：(1)解釋範例中的機率數學運算用到了何種數學原理。(2)解釋某個動作的意圖及目的為何。(3)在範例尚為提及的情況下，預先進行機率的運算，並解釋其原因及目的。

二、自我解釋的模式

Neuman, Leibowitz 和 Schwarz(2000)研究學習者在進行代數解題時之自我解釋模式，其研究結果發現優秀的問題解決者傾向於先說明目前想要進行的計畫步驟(自我調節)，然後對自已的計畫進行進一步解釋(辯解)；由於「先自我調節後辯解」的自我解釋模式，代表學習者正在建構問題空間中的解題步驟，因此這種學習者的問題解決成效較高。差的問題解決者則傾向先對問題推論(推

論)，然後對自已下的推論進行進一步的說明(澄清)；由於「先推論後澄清」的自我解釋模式，代表學習者還在建構初步的問題空間(了解問題)，尚未真正進入解題階段，因此這種學習者的問題解決成效較前者來的差。

雖然研究發現自我解釋的模式與問題解決成效有關聯，然而是否是因為自我解釋的模式影響到問題解決成效，還是其它個別因素的影響，仍有待進一步的證實。

肆、 自我解釋的教學設計

在學習的過程中，不論是自發性的行為，或是由外界所促發，自我解釋促進了學習者對學習的反省，對教材的深入思考及推論，因此自我解釋可視為一種反思的歷程(Chi et . al., 1994 ; Davis, 2000)。

為了促使學習者進行自我解釋，可以在教學中加入了自我解釋提示，或是在教學前實施自我解釋策略訓練以降低學習者進行自我解釋的困難度。此兩種方式詳述如下：

一、自我解釋提示

為了幫助學生進行自我解釋活動，Chi 等人(1994)在生物循環系統的教學中，以三種不同的提示要求學生在閱讀的過程中進行自我解釋。第一種提示為一般性的自我解釋教學，於閱讀教材前實施。此提示以文字的方式，要求學生將閱讀教材時的心得、想法、疑問、學習策略、及文章所隱含的意義大聲講出來。第二種提示為教材所提供的引導問題，為生物循環系統的功能性問題提示；

這些問題置於每頁教材的最下方，學習者在進行完自發性的自我解釋活動後，可以根據這些引導問題，詳細回答循環系統中各個構成要素之主要功能。第三種提示為實驗人員在學習者進行自我解釋的過程中所給予的問題提示；由於學習者的自我解釋可能不夠詳盡或在自我解釋活動的實施上遇到了阻礙，因此實驗人員可以要進學習者做進一步的解釋或澄清。例如：“你剛剛講的那句話是什麼意思？你可不可以再解釋一下“、“你已經停頓很久了，請把你遇到的問題講出來“。

其研究結果顯示，藉由這三種提示，學習者在記憶人體循環系統的概念名詞上有不錯的學習成效，此外對於各器官的運作原理及功能亦建構了相當完善的心智模型。

二、自我解釋策略訓練

為了幫助學生進行自我解釋活動，Bielaczyc、Pirolli 和 Brown (1995)在進行自我解釋活動前，將教材的呈現方式分成(1)文章(2)範例(3)文章及範例，等三種主要類別，然後再根據各個類別進行自我解釋策略的訓練。

當教材純粹以文章的形式呈現時，訓練學習者找出文章中的主要概念並闡明概念間的相互關係；此訓練的目的在於促進學習者對文章的理解程度。當教材以範例的形式呈現時，訓練學習者確定範例的架構形式並找出範例的用意為何；此訓練的目的在於幫助學習者熟悉範例特徵及潛在目的。當教材以文章及範例的形式呈現時，訓練學習者將文章中的概念連結至範例之中；此訓練的目

的在於提昇學習者對於範例的理解程度。

其研究結果顯示：(1)接受自我解釋策略訓練的實驗組在學習成就上顯著優於沒有接受訓練的控制組。(2)由於實驗組有實施自我解釋策略訓練，因此在自我解釋策略的應用上較控制組來的優異。

第三節　程式語言學習的相關研究

壹、 程式設計的本質

在程式設計的過程中需要考慮到許多的細節，因此就如同數學與物理一樣，程式設計是一門相當精密的學科。此外學習者在程式設計中必須聚精會神地以各種方式解決程式設計問題，因此程式設計在本質上屬於另一種型式的問題解決學習，而數學，物理，以及其它科學相關學科亦有相同的性質(Schwartz & Steven, 1988)。

根據一般性的問題解決模型，程式設計可分成五個子步驟：(1)了解問題；(2)找出一般性的問題解決方法；(3)將一般性的問題解決方法轉換成與程式相容之演算法；(4)將演算法轉換成相關之程式語言：(5)測試並進行程式除錯(Winslow, 1996)。從步驟二與步驟三的銜接上，可看出程式設計與一般問題解決的不同處在於程式設計不只涉及到問題解決還需進行額外的設計流程，也就是說學習者不僅要推導出如：“ 時速=70 公里/每分鐘“ 、“ X=80“ ，等特定答案，還要進行額外的程序以建構出複雜的學習作品；其它如美術(美術作業)、幾何學(幾何證明)亦有相同的性質，因此程式設計在本質上亦可視為另一種形式的建構式學習。

貳、 程式設計的知識階層

有許多研究探討程式初學者的學習歷程，在這些研究中都可以發現一個相同的問題:從程式設計中, 學生到底學了那些知識？這些知識之間的關係為何？

由於人的知識系統是屬於階層狀的結構(Gagne, 1980)，因此我們可從知識階層的觀點來描述程式初學者在進行程式設計時，須具備那些知識。

Linn(1985)認為程式設計初學習必需經歷一連串的認知階段才能由初學者成為專家，這三個階段由低階到高階分別是(1)程式語言特徵階段(language feature)、(2)程式設計技巧階段(design skills)及(3)問題解決技巧階段(problem solving skills)。學習者在具備前一階段知識之後才能繼續下一階段知識的學習。

Mayer 和 Fay (1987) 以 Logo 程式設計為例，將程式設計的知識分成三類：第一類為程式設計的語法知識，例如了解 Logo 語言中的 RT，LT，FD，BK 等關鍵字分別代表往右，往左，往前，往後等繪圖指令。第二類為程式設計的語義知識，例如學習者可以意識到 RT 指的是 Logo 中小海龜的右手邊而非螢幕的右邊。等三類則為問題解決的學習遷移知識，例如在學習完 Logo 程式設計後，可能促進學習者空間的認知能力；因此若在 Logo 的程式設計課程後，提供學習者陌生的地圖並給予指示，學習者將有足夠能力對此地圖進行瀏覽。由於程式語言特徵可視為程式設計的語法知識，程式設計技巧因屬於程式設計之特定領域，可視為程式設計的語義知識，問題解決技巧則可視為問題解決的學習遷移知識，因此 Mayer 對程式設計知識的分類與 Linn 的區分頗為一致。

Bayman 和 Mayer (1988) 進一步將程式設計的語義知識分為概念性知識(conceptual knowledge)和策略性知識(strategic knowledge)，概念性知識指的是程式架構及執行流程等相關知識，當學習者描述程式的詳述執行過程時就是在展

示其自身的概念性知識；而策略性知識則是指融合程式語法及概念性知識藉以進行問題剖析、詳述問題、撰寫演算法及程式除錯的能力，當學習者針對不同程式設計問題擬定演算法時就是在展示其自身的策略性知識。

由於概念性知識所包含的範圍相當廣範，因此有必要從知識階層的角度對概念性知識進行細部分類。從認知心理學的角度，任何的學科知識皆可分成陳述性知識(declarative knowledge)、程序性知識(procedural knowledge)及條件性知識(conditional knowledge)。陳述性知識指的是學科中的事實性知識，學習者若能將所學的陳述性知識提升成具備實用功能的程序單元，這些程序單元即是程序性知識，條件性知識則是指了解何時何地可採用那一種程序性知識之能力。

在這樣的觀點下，Oliver(1993)將程式設計的語義知識依知識階層分成三種不同的類型。第一類知識指的是程式架構的知識，學習者若具備第一類知識就可以了解程式的架構並能描述程式執行後的結果。第二類知識指的是演算法的知識，學習者若具備演算法知識也就代表著能夠將程式中的各個程序加以組合和運用。第三類知識指的是設計演算法的能力，學習者若具備設計演算法的能力就能夠根據不同問題的需求設計出不同的演算法。Oliver 亦依此分類發展了一套測驗：演算法處理測驗(Algorithmic Process Test, APT)，用來評量程式初學者的知識層級，並證明此三種知識具有階層性。從廣義角度來看，第一類知識可視為程序性知識、第二類知識可視為條件性知識，第三類知識則可視為策略性知識。

綜合過去學者的研究成果（表 1），程式設計的知識階層可分成程式語法知識、程式語義知識、及問題解決學習遷移知識，其中的程式語法知識和程式語義知識屬於程式設計的專屬領域，而程式語義知識又可細分為陳述性知識、程序性知識、條件性知識及策略性知識。

表 1

程式設計知識階層表

<table>
<tr><td rowspan="2">知識領域</td><td>Linn (1986)</td><td>Mayer&Fay (1987)</td><td>Bayman&Mayer (1988)</td><td>Oliver (1993)</td></tr>
<tr><td colspan="4">知識分類</td></tr>
<tr><td rowspan="6">程式設計</td><td>程式語言特徵</td><td colspan="3">語法知識</td></tr>
<tr><td rowspan="5">程式設計技巧</td><td rowspan="5">語義知識</td><td rowspan="3">概念性知識</td><td>陳述性知識</td></tr>
<tr><td>程序性知識</td></tr>
<tr><td>條件性知識</td></tr>
<tr><td colspan="2">策略性知識</td></tr>
<tr><td>其它</td><td>問題解決技巧</td><td colspan="3">問題解決學習遷移知識</td></tr>
</table>

参、 程式初學者與專家的差異

從過去的相關的程式語言教學研究可以發現：初學者雖然在程式語言的學習保留上有不錯的學習成效，然而在程式設計問題的學習成效上卻不盡理想 (Mayer, 1981)。為了要瞭解學習者在程式語言的學習中有那些學習問題，許多學者開始探討程式初學者與專家的差異。

根據程式設計經驗的多寡，可將程式設計人員分成：新手、初學者、中等程度、及進階等類型。為了解不同類型程式設計者之學習模式，Adelson(1981)

要求程式設計專家與初學者回想教材中的程式片斷，其結果發現：程式初學者傾向根據程式表面特徵之不同對程式碼進行重組；而程式設計專家則傾向根據程式功能以及程序類型之不同對程式碼進行重組。在 Mayer(1988)的研究亦發現專家對程式程序所進行的分類較初學者來的精確，因此專家在解題時有較多可用的程序類型。其發現與 Adelson 之研究結果頗為一致。

由於程式初學者只將注意力放在程式碼的語法結構，因此造成初學者產生程式概念偏差及程式概念不足(Cooke & Schvanevedt, 1988)等學習問題，這使得初學者對程式執行時的電腦內部運作缺乏正確的概念，降低初學者心智模型的正確性(Schwartz & Steven, 1988)。反之，由於專家原先已具備建全的程式語法知識，對於程式碼在電腦內的運作流程有較正確的認知，因此將注意力專注於程序類型及功能，進而建構出較為完整的心智模型(Mayer, 1988)。

除了學習模式的不同，在解題策略方面，專家習慣對問題進行模組化的程式設計，因此有較高的解題策略，而初學者則缺乏良好的問題解決策略。這些因素使得初學者在學習自信以及自我控制上產生問題，並對程式語言學習造成負面影響(Mayer, 1988 ; Schwartz & Steven, 1988)。

肆、 概念模型在程式語言教學中之應用

由以上的研究結果可以得知：專家不只較初學生有較多的程式設計知識，在知識的組織及架構上也有明顯的差別。因此 Bayman, Piraye 和 Mayer(1982)開始研究學習者在程式語言學習中所建構的心智模型。Bayman 等人將程式碼

視為一連串的執行指令，而電腦的內部元件則負責執行這些指令。這些指令通常涉及到電腦運作時之相關概念，例如：電腦內部不同部門(輸入部門，記憶體，算術邏輯部門，輸出部門)如何相互作用來完成一個特定動作。一個好的心智模型必需包合相關的程式指令內部運作概念，然而指令的運作涉及到電腦內部元件的運作，學習者往往看不到其詳細過程，因而造成心智模型建構上的困難。

為了幫助學習者建構出完善的心智模型，Mayer(1988)以圖像的方式呈現出電腦的內部構造；這些繪製於教材中的圖像詳細說明不同的程式指令如何利用電腦的運作規則對內部元件進行操作，藉以提供程式初學者概念模型的學習輔助，其結果顯示：(1)提供概念模型的教學成功地促進了學習者的心智模型，然而低能力的學習者在心智模型的改進程度較高能力者來的高，其原因可能是因為高能力的學習者已經擁有較完整的心智模型，因此其進步空間有限。(2)概念模型教學促進了學習者在問題解決的學習成效。(3)在概念知識方面得分高的學習者，其問題解決的學習成效較高；而概念知識方面得分低的學習者，其問題解決的學習成效較低。

Schwartz 和 Steven(1988)則進一步提出後繼課程(meta-course)的概念，後繼課程的設計目的在於提供視覺化模形來幫助學習者習得程式執行時電腦內部運作的相關概念。後繼課程將程式所用到的變數、變數中的儲存數值、程式的控制流程、及輸出至螢幕之字串以圖像的方式加以呈現，幫助學習者建立對程式指令的心智模型。此外，課程中亦提供簡單的策略工具輔助學習者進行程式的

除錯，其研究結果指出：學習者在接受了後繼課程的教學之後，不論在程式設計或是程式除錯方面，其學習成就都較控制組來的好。

由於概念模型有助於程式語言的學習，近年來有許多學者致力於創造出視覺化的程式學習環境，將電腦內部的執行過程予以圖像化。這些環境通常動態地呈現出程式中程式碼的執行流程，或是在學習者進行程式設計時提供不同的程式執行狀態，例如有些工具可以將程式設為除錯狀態，將各行程式碼的執行動作以圖形方式具體呈現(Shih & Alessi, 1994; Karsten & Kaparthi, 1996; Byrne, 1999; Smith & Webb, 2000)。

第四節　歸納與結論

學習歷程可分為知識獲取階段以及問題解決階段，在知識獲取階段，學習者根據先備知識將教材中的文章及範例轉化成心智表徵，直接從教學中獲取與技能相關的初步知識。在問題解決的過程中，學習者藉由了解問題、找尋解決方案等問題解決策略將原本習得的陳述性知識轉變成高層次的問題解題技能。而自我解釋在知識的獲取階段可以幫助學習者建立完善的問題空間，因此可以促進資訊的深度處理，增加推斷規則的產生；在問題解決階段則可以幫助學習者對不同型態的問題外在表徵進行轉換，進而提升問題解決的成效。

程式設計具有問題解決及建構式學習的本質。從過去文獻得知，視覺化的概念模型在程式語言的教學中扮演著相當重要的角色，研究結果顯示，概念模型可以幫助學習者建立完善的心智模型，並促進問題解決方面的學習成效(Bayman, Piraye, & Mayer, 1982; Mayer, 1988)。除了可利用概念模型提升程式語言的學習成效外，亦可藉由自我解釋建構出完善的問題基模，促進解題時問題空間的建立，進而提供了建構式學習及反思等學習機會，然而卻少有研究將自我解釋學習策略實際應用在程式語言的教學設計之中。

由於學習者的個別差異亦會影響其學習成效，例如學習者的先備知識便是一個主要的主要個別因素，所以先備知識也是一個值得探討的課題。但是過去研究卻發現先備知識與自我解釋的品質與數量並無明顯相關，這代表問題解決

的成效並不完全取決於先備知識多寡，自我解釋活動的實施亦可以提升問題解決的成效。

因此本研究提供「自我解釋原則提示」和「自我解釋問題引導」兩種自我解釋學習活動，並探究此兩種學習活動是否能夠促進學習者在陳述性知識(事實性知識)及問題解決技能(程序性技能、條件性技能、策略性技能)上的學習成效。其研究目的在於(1)藉由不同的自我解釋學習活動促進自我解釋的進行，進而提升程式語言學習成效，(2)藉由不同的自我解釋學習活動促進自我解釋的進行，進而增進自我解釋的產生數量，(3)探討自我解釋學習活動與先備知識對於學習成效與自我解釋數量的交互作用，(4)探討學習者對於自我解釋學習活動的態度與看法。