# 使用搭檔編程來探索程式設計課程之認知能力—以技術學院學生為例

董炳信*,陳志遠,邱守男,羅裕群

桃園創新技術學院資訊工程系

## 摘要

學習程式設計是個困難的過程，尤其是對修電腦程式設計課程的大一新生。搭檔編程是另一種合作學習的技術，已經被發現對學習程式設計有很大的益處。其中一個益處是搭檔的學生能比未搭檔的學生可獲得較高的測驗分數。然而，這個測驗分數並不能精確地反映出學生的認知能力。因此，本研究利用Bloom分類去探索搭檔的技術學院學生或未搭檔的技術學院學生在程式設計上的認知能力。根據實驗的結果，搭檔編程能提升有搭檔的技術學院學生的認知層次，此外教師因此可了解學生的認知層次，然後調整教學策略、課程教材以及教室活動，以提升學生的學習成效。

關鍵字: Bloom分類、合作學習、認知能力、搭檔編程

*Email: philip@tiit.edu.tw Tel:03-4361070#6938 Fax:03-431070#6999

# A STUDY OF USING PAIR PROGRAMMING FOR EXPLORING COGNITIVE ABILITY IN PROGRAMMING COURSES —INSTITUTE OF TECHNOLOGY AS AN EXAMPLE

Ping-Hsing Don*,Chih-Yuan Chen, Shounan Chiou, Y.C. Luo

Department of Computer Science and Engineering

Taoyuan Innovation Institute of Technology

## Abstract

The learning of programming is a difficult process especially for first-year students in the computer programming course. Pair programming is an alternative collaboration learning technique and has been found significantly learning benefits. One of the benefits is that paired students can achieve higher exam grades than students who program alone. However, the exam grade cannot accurately reflect the students' cognitive abilities. Therefore, this study adopts Bloom's taxonomy as a framework to explore the cognitive abilities students acquired after students have performed (or not performed) pair programming activities in an programming course. According to the experiment results, pair programming can promote students' cognitive levels when comparing with students who program alone. In addition, instructors then can realize the students' cognitive levels and then adjust their pedagogy, course materials and classroom activities to promote learning performance of students.

Keywords: Bloom's taxonomy, Collaboration learning, Cognitive ability, Pair programming

*Email: philip@tiit.edu.tw Tel:03-4361070#6938 Fax:03-431070#6999

# 壹、簡介

在程式設計課程中，學生應掌握基本概念(或構念)以及培養有關開發簡單的程序編寫技巧。由於學生無法直接從課程的授課中獲得的抽象程式構念和技能，他們必須學習如何程式設計[1]。程式設計是通過嘗試和錯誤的一個實際的學習方式，學生可以在程式設計過程中接受建設性的反饋和糾正，以提高他們的認知能力[2]。因此，教師通常會在課程中指派給學生一些程式設計的作業，並要求學生他們獨立編寫程式。雖然老師的良好的辛勤指導，大部分學生仍對程式設計艱困，然後表現不佳[3][4]。

新手同學於程式設計課程中遭遇了很多的困難，除了新手學生的語法和語義基礎不穩固的問題外，也常有如遺失、插入、錯位、使用不當的諸多問題。此外，新手學生在程式設計過程中通常無法撰寫出完整的程式碼[5]，最近很多程式設計的環境或工具的開發，可減少新手同學於程式設計課程中的學習困難。這些程式設計開發環境或工具，主要目的在尋求視覺化整個程式設計執行的完整程序[6][7]。雖然在可視覺化(非傳統)的環境中，新手同學可以減少語法和概念的負擔並加快程式設計產品的開發，但會導致新手同學忽略程式設計語法和語義的重要性，以及如何撰寫程代碼的真正工作。

合作學習的活動，可以取代程式設計教學[8][9]。搭檔編程，是教育環境中一個合作學習的實例[10]，它提供了一個框架，搭檔的學員一起在一台電腦前實現一個共同的目標。雖然有些教師對搭檔編程有一些誤解[11]，但大多數研究已證明，搭檔編程是一種有效的程式設計課程的教學法[8][12]。

越來越多的研究人員已經明確記載學生在程式設計課程中使用搭檔編程方式的教育利益[11][13]。舉例來說，搭檔的學生不僅能獲得更好的程式設計作業成績，爾後在獨立程式設計的考試中，學生亦能有較高的成績[14]。然而，以往搭檔編程的研究中，獨立的考試成績不能準確反映個人事實的認知能力。因此，本研究應用Bloom的分類[15]作為一個框架，來探索學生執行（或不執行）搭檔編程的程式設計的認知能力。

# 貳、背景

## 2.1 搭檔編程

搭檔編程是軟件開發業Beck（2000年）的方法創建的極限編程（eXtreme Programming）的一個關鍵組成部分[10]。搭檔編程提供了一個框架，使他們的任務搭檔一位程式設計員。程式設計搭檔員在程式發展過程中需要與另一段程式碼關聯時，他們可以用手或指尖指向電腦螢幕。因此，程式設計搭檔員可以立即分享論證知識與意見，此種方法它可以鼓勵搭檔員完善或修改自己的知識和技能，進一步獲取新的認知[16]，也可以在更短的時間內產生更高質量的程式碼，因為會更早發現許多程式碼的錯誤及缺損[8][17]。

最近，研究人員已經證明，採用搭檔編程的程式設計課程，對學生和教師可以產生很多的好處。這些好處包括提高程式的質量、更少的時間完成程式、考試成績較高、及格率較高、更多的樂趣和信心、正面的合作學習態度等[4][13][14]。雖然所有這些研究顯示，採用搭檔編程對學習表現沒有負面影響，但也有一些研究擔心，包括安排時程的衝突，不對稱的合作夥伴和不平等的參與[18]。然而，一些研究發現即使搭檔到不對稱的合作夥伴，搭檔編程亦可有效的提升學生學習程式設計 [4][8][13]。

## 2.2 Bloom's 分類

Bloom和他的同事（1956年）分類認知為六個層次，即"知識"、"理解"、"應用"、"分析"、"綜合"和"評鑑"的認知方面。認知層次是從簡單到複雜，具體到抽象。Bloom的分類，可以提供一個有用的結構被用在計算機科學的課程設計或成就評估[19]。根據 Bloom 分類法的基礎上的認知能力的分級可以用來準確地評估學生的學習成果[14][20]。Lister和Leaney （2003年）將Bloom的分類的六個類別分為三對，並通過標準參照分級的方法進行分類，了解每個學生的程式設計認知能力[21]。Bloom 的分類可以用來校準一個比較式的學習成果，Bloom的認知可以應用到學生的認知能力進行評估的準確和簡潔的模型[2]。學生的認知能力在程式設計課程的定義如下。

- 等級1：知識：記得以前學過內容的能力。例如，學生可以記住的程式設計的語法。
- 等級2：理解：了解以前學過內容的意義。例如，學生可以理解的程式設計語言的語義，或預測程式設計的輸出。
- 等級3：應用：能夠運用以前學過的內容到新的或特定的情況。例如，根據給定的方法、資料和變數，學生可以追踪或完成特定的程式設計作業。
- 等級4：分析：分解和理解程式的結構。例如，學生可以識別一個給定的程式的

正確性，或根據問題的定義可修改一個給定的程式。

- 等級5：綜合：能夠運用以前學過的內容，創造一些新的程序。例如，學生可以寫一個完整的程式，以解決給定的問題。許多程式設計的教科書稱之為“解決問題”。
- 等級6：評鑑：在給定目的程式中有能力判斷有關構念或演算法。例如，學生可以查看一個給定的程式，然後設計一個良好和正確的程式。

# 參、研究方法

在搭檔編程後，個人責任發生於所有搭檔的學生採取單獨的考試，並接受個人成績後[10]。先前搭檔編程的研究證明了搭檔的學生在個人的考試中的表現通常與他們的搭檔編程作業有正相關[12][20][21]。然而，個人的考試成績不能準確地反映個人的認知能力。Bloom的分類是一個有用的認知過程分類工具。因此，本研究採用基於Bloom的分類的考題來探討學生已執行（或不執行）搭檔編程課程的學生的認知能力。

## 3.1 課堂中的程式設計作業

Radermacher等學者(2011年)指出新手同學無法在規定的實驗時間內獨力完成程式設計作業[9]，即使使用搭檔編程的方法，Braught等學者(2011年）指出，只有20%左右搭檔的學生能全面完成程式設計作業[8]。為了減少學生的認知負荷，並系統地建立基本的程式設計構念和技能，在本次研究中的程式設計作業採用“骨架”的方式。搭檔或獨自的學生期望延續或修改所指定的程式設計作業。

## 3.2 基於Bloom 分類的考題

一個好的考題應該讓教師了解學生的問題並提高教學能力。Scott（2003年）建議，在考試中的問題應來自所有Bloom的分類[23]，因此標準參照的分級方法可以給教師測學生的認知成就，而不是只看學生的成績分數。因此，本研究應用Bloom分類法的考題，以評估學生的認知能力。此外，考題結合程式設計作業，可以讓學生在搭檔中確保學生掌握程式設計知識和技能並對自己負責[10][14]。因此在這項研究中，考題類似之前在實驗室中指定的程式設計作業。

在本研究中的考題中並沒有測試“評鑑”的能力，即使有些學生的認知能力，在雙

人程式設計作業中達到“綜合”的能力。在每次考試的考題類型包括：（1）要求學生回答定義或語法問題以測試“知識”的能力，（2）要求學生預測程式碼片段的輸出以測試“理解”的能力，(3)要求學生完成一個特定任務的程式碼所缺少的問題以測試“應用”的能力， （4）要求學生在找錯或修改程式碼片段以測試“分析”的能力，（5）指定特定任務的問題，要求學生寫一個完整的程式以測試“綜合”的能力。在每次考試中有五個級別，每個級別都有20分。

Whalley等學者[24]應用Bloom的分類開發選擇題來評估新手在程式設計上的技能。然而，他們認為“綜合”的多選題不僅設計難度大，但也不能直接測試學生在創造一個完整的程式來解決問題的能力。因此，在每次撰寫程式碼的考試中只要求每個學生寫一個簡短而完整的程式碼，來評估“綜合”的能力。“綜合”能力的評分標準是基於撰寫出的程式碼的正確性和完整性的程度。選擇題或填充題的問題是以填空的方式來評估其他的認知能力。除了“綜合”能力外，在每次考試中每一等級有四個問題。兩個教師分別審查了每次考試問題的所有內容和分類，並分為五個等級。因此，每次考試中的每個問題可以是認知能力的很好指標。

## 3.3 搭檔學生的問卷

問卷可用於評估實驗的有效性，並提供學生的學習過程和教學效果的見解。因此，每個搭檔的學生的問卷可探索搭檔編程對搭檔學生的感受是否有效。此問卷是基於科技接受度模型和學習活動中搭檔編程的效度。在每一個問卷敘述，學生可指出自己的同意程度，我們使用以下五點李克特(Likert)量表：1 - 堅決不同意，2 - 不同意，3 - 既不同意也不反對，4 - 同意，5 - 強烈同意。在問卷中的問題如下：

- 問題1：我很喜歡搭檔編程。
- 問題2：我從合作夥伴了解基本概念。
- 問題3：我從合作夥伴獲得了更多的對概念的理解與解釋。
- 問題4：透過搭檔編程作業提昇我的應用能力。
- 問題5：透過搭檔編程作業使我更有效地分析程式碼。
- 問題6：我有更多的信心面對一個新的問題，能獨立撰寫一個程式。
- 問題7：我喜歡與不同的合作夥伴完成每個搭檔編程作業。

# 肆、實驗

## 4.1參與者與課程

在2011年秋季學期，在桃園創新技術學院資訊工程系一年級學生的物件導向程式設計課（Computer Programming I）進行了一個實驗。一班有54名學生(A班)和另一班48名學生(B班)，學生大部分對程式設計都不熟悉，這兩班由同一個教師（作者）任教，學生每星期參加3個小時的授課和1小時的上機實習，此課程共有18週。

由於封閉的實驗室比開放實驗室更使學生投入[10][14]，因此本研究中是使用一個強制性的封閉實驗室，讓學生每週在電腦教室中完成教師所指定的程式設計作業。兩個班的學生在第九週的期中考試後，教師隨機選擇有54名學生的班級當實驗組(A)，有48名學生的班級當控制組(B)。在控制組，學生在上機實習中單獨完成程式設計作業。在實驗組，教師首先介紹搭檔編程的描述和它的好處給學生，然後與助教展示完成搭檔編程作業的處理過程。爾後起，54名學生將與搭檔夥伴一起做他們的程式設計作業，如圖1，展示了搭檔同學間的互動解題情境。此54名學生被授權由自己選擇自己的合作夥伴，如果學生找不到合夥人，教師或助教將隨機指派合作伴侶，並且每次作業需搭配不同的夥伴。在最後一周，兩個班同時舉辦期末考試。而這兩個班級的程式設計作業與考試是相同的，考試是在課堂上使用筆和紙進行。

圖 1 搭檔學生共同解題之畫面

## 4.2 資料分析

資料是從兩個班的學生所收集的數據，這些數據包括期中考分數當成前測成績，

期末考分數當成後測成績。刪除取消修課與沒有參加考試的學生。因此，實驗組(A班)剩50名學生，控制組(B班)剩41名學生。本研究對獲得的數據進行了分析，來確定學生執行（或不執行）搭檔編程後學生的認知能力的變化。

# 伍、結果與討論

## 5.1 前測分析

獨立樣本的T檢定(T-test)是可以檢定兩群體特性的期望值是否相等之一種常用的統計方法，由於兩班是獨立性的，為了比較兩班的程式設計程度間是否存在差異，因此我們使用獨立T檢定。因此在作T檢定時，必須考慮此兩班的標準差是否相等的問題。而標準差(standard deviation)，是指以全部期中考成績的算術平均分數為中心，計算全部期中考成績與算術平均分數的平均距離，用來表明期中考成績的離散情況。如果要檢驗是否顯著差異，其T值計算公式為：

$T = \frac{\overline{X}_n - \mu}{S_n/\sqrt{n}}$，其中：

$X_n$是檢驗樣本的平均數；

μ是已知總體的平均數；

$S_n$是樣本的標準差；

$n$是樣本容量；

表 1 前測分析

| 班級 | 人數 | 平均分數 | 標準差 |
|---|---|---|---|
| 實驗組 | 50 | 33.8 | 12.06 |
| 控制組 | 41 | 33.54 | 11.79 |

$p > 0.05$

如表1數據，經T檢定之後算出$p$ ($p$值代表的為假設兩班差異的檢定值)，因$p >$

0.05，因此實驗組與控制組兩班的期中考試成績沒有顯著的差異，根據此結果，我們可以得出結論，這兩個班的學生對修物件導向程式設計課之前的程度類似。

## 5.2 後測分析

如表2數據，經T檢定之後算出$p < 0.05$，因此兩個班在期末考考試後，他們執行（或不執行）搭檔編程有顯著區別。根據結果，我們可以得出結論，搭檔編程幫助搭檔的學生在期末考試上有更好的表現，此與以前類似的研究結果是一致的[9][14]。為了了解學生的認知變化，本研究進一步分析了在期末考試後的認知能力。如表3數據所示，“搭檔”是指實驗組，“單獨”是指控制組。搭檔學生的表現，在“知識”能力上顯著高於單獨的學生（顯著性 = 0.036）;搭檔學生的“理解”能力的表現是明顯高於單獨學生的表現（顯著性 =0.001）;搭檔學生表現“應用”能力明顯高於單獨學生的表現（顯著性 =0.013）;搭檔的學生在“分析”的能力明顯高於單獨學生的表現（顯著性 =0.047）。然而，在“綜合”能力（顯著性 = 0.059）則無顯著性差異。總之，使用搭檔編程的認知變化的影響是明顯的。

表 2 後測分析

| 班級 | 人數 | 平均分數 | 標準差 |
|---|---|---|---|
| 實驗組 | 50 | 55.6 | 14.42 |
| 控制組 | 41 | 47.56 | 12.61 |

$p < 0.05$

搭檔編程能有效地幫助對學生完成“知識”、“理解”和“應用”能力的目標。但對“知識”能力的顯著性稍低於“理解”和“應用”的能力。研究人員認為，合理的解釋是在期末考試中“知識”的考題，通常是簡單的語法問題，此基本題目在修完此課都具有基本的知識能力。這也證實了Daigle 和 Doran（1998年），在實驗室的合作能促進個體的認知活動中的“理解”和“應用”的能力[22]。除了“綜合”，“分析”的顯著性比其他的認知能力低一點。從標準差的“分析”能力明顯高（標準差 = 5.30），研究人員認為，合理的解釋是一些搭檔的學生仍在掙扎程式除錯。搭檔編程顯然不能提升“綜合”的能力。研

究人員認為，“綜合”能力的成長是從不斷地練習與花費時間。

表 3 在後測獨立T-檢定與五個認知能力實驗結果

| 認知能力 | 組別 | 平均分數 | 標準差 | T | 顯著性($p$) |
|---|---|---|---|---|---|
| 知識 | 搭檔 | 18.6 | 2.27 | 1.80 | 0.036 |
| | 單獨 | 17.7 | 2.52 | | |
| 理解 | 搭檔 | 16.6 | 3.26 | 3.05 | 0.001 |
| | 單獨 | 14.4 | 3.57 | | |
| 應用 | 搭檔 | 12.1 | 4.05 | 2.20 | 0.013 |
| | 單獨 | 10.0 | 4.87 | | |
| 分析 | 搭檔 | 6.2 | 5.30 | 1.68 | 0.047 |
| | 單獨 | 4.51 | 4.30 | | |
| 綜合 | 搭檔 | 2.1 | 4.41 | 1.56 | 0.059 |
| | 單獨 | 0.98 | 2.30 | | |

初學的同學會花更多的時間在“知識”和“理解”能力，當學生積累更多的知識，他們更容易轉移到更高的認知能力，如“分析”和“綜合”。從實驗結果來看，在期末考試成績好的搭檔或單獨學生通常有較高的“綜合”能力。雖然大多數搭檔的學生不能達到傳統的“解決問題”的第五級“綜合”能力目標，但實驗結果仍然顯示搭檔的學生比單獨的學生具有更高層次的思維技巧。

## 5.3 搭檔學生的感知

由表4分析搭檔學生的問卷調查結果可得知，問題1的調查結果顯示，大部分的搭檔學生對搭檔編程有正面的態度。問題2的調查結果顯示，搭檔學生會自然地相互指

導並促進學習基本概念。問題3的調查結果顯示，搭檔學生強烈認為他們向合作夥伴解釋課程的教材內容時學到了程式設計的概念。問題4的調查結果顯示，搭檔的學生當他們能夠方便地問夥伴問題可使他們對實際的應用技巧更有信心。問題5的調查結果顯示，搭檔的學生認為他們學會了更有效的方式了解問題，減少錯誤，並且能夠填補自己與其他學生的知識差距。然而，從問題6的調查結果，學生能獨立應用概念解決新的程式設計問題與搭檔與否無關。研究人員認為合理的解釋是，理解概念和規則不保證能夠解決新新的程式設計問題，因為新手程式設計員常在概念應用上有共同的問題[6]。此外，從問題7的調查結果顯示，搭檔的學生喜歡交換不同的合作夥伴。研究人員認為，不同的合作夥伴能填補知識缺口並提供技能的互補。綜上所述，本調查結果顯示搭檔編程是一個優越的學習環境。

表 4 搭檔學生的問卷結果

| 問題 | 平均數 | 中位數 | $p$ ($\mu$=4) |
|---|---|---|---|
| 問題1 | 4.26 | 5 | < .001 |
| 問題2 | 4.33 | 4 | < .001 |
| 問題3 | 4.07 | 4 | < .001 |
| 問題4 | 4.55 | 4 | < .001 |
| 問題5 | 4.31 | 3 | < .001 |
| 問題6 | 3.68 | 3 | < .001 |
| 問題7 | 4.67 | 5 | < .001 |

## 陸、結論與未來工作

本研究應用Bloom的分類作為一個框架，有系統的評估技術學院學生在搭檔編程後他們的認知能力。本實驗結果顯示，實行搭檔編程之技術，在封閉的實驗室中

搭檔的學生可以產生更好的個人考試成績。此結果進一步提出了對個人的認知能力的見解。本實驗的結果可以反饋給教師和學生進行必要的改善教學和學習。例如，教師可以對教學策略進行評估、修改和制定課程和課堂活動，以提高技術學院學生的認知水平能力，從而提高學生的學習成效，此外學生可以看到他們對特定主題的認知困難，然後修改自己的學習流程或策略。雖然在此次實驗中的搭檔編程作業時間比Nosek[17]的實驗還長，但研究人員認為，還需要進一步的實驗是面向更大的搭檔編程作業和多次的考試。未來的研究工作方向是探討搭檔編程隨後的相關搭檔編程課程式設計的長期影響以及對技術學院學生的未來學習成就的表現。

## 柒、參考文獻

[1] Ben-Ari, M., (2001), Constructivism in computer science education, Journal of Computers in Mathe-matics and Science Teaching, 20 (1), pp. 45–73.

[2] Buck, D., Stucki, D., and JKarelRobot (2001), A Case Study in Supporting Levels of Cognitive Development in the Computer Science Curriculum, Proceedings of SIGCSE Technical Symposium on Computer Science Education, ACM Press, pp. 16-20.

[3] McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y., Laxer, C., Thomas, L., Utting, I., and Wilusz, T. (2001), A Multi-national, Multi-institutional Study of Assessment of Programming Skills of First-year CS Students, ACM SIGCSE Bulletin, 33(4), pp. 125 – 140.

[4] Nagappan, N., Williams, L., Ferzli, M., Yang, K., Wiebe, E., Miller,C. and Balik, S. (2003), Improving the CS1 Experience with Pair Programming, In Proceedings of SIGCSE 2003, pp. 359 – 362.

[5] Lister, R., Adams E.S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J.E., Sanders, K., Seppällä, O., Simon, B. and Thomas, L. (2004), A Multi-National Study of Reading and Tracing Skills in Novice Programmers, SIGSCE Bulletin, 36(4), pp. 119-150.

[6] Goldweber, M., Bergin, J., Lister, R., and McNally, M. (2006), A Comparison of Different Approaches to the Programming Course, ACE2006, 52.

[7]Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., and Rusk, N. (2008), Programming by choice: urban youth learning programming with scratch, ACM SIGCSE Bulletin, 40(1).

[8] Braught, G., Wahls, T., Eby, L. M. (2011), The Case for Pair Programming in the Computer Science Classroom. ACM Transactions on Computing Education, 11, 1 (February 2011), Article 2.

[9]Radermacher, A., and Walia, G. S. (2011), Investigating student-instructor interactions when using pair programming: an empirical study. To be published in 24th Conference on Software Engineering Education & Training (CSEET ’11). IEEE Computer Society, Washington, DC, USA.

[10]Beck, K. (2000), Extreme Programming Explained: Embrace Change, Addison-Wesley, Boston.

[11] Williams, L. and Kessler, R. (2003), Pair Programming Illuminated. Reading, Mass.: Addison Wesley.

[12] McDowell, C., Werner, L., Bullock, H., and Fernald, J. (2002), The Effect of Pair Programming on Performance in an Programming Course, Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education, ACM SIGCSE Bulletin, 34(1), pp. 38-42.

[13] McDowell, C., Werner, L., Bullock, H., and Fernald, J. (2006), Pair Programming Improves Student Retention, Confidence, and Program Quality, Communications of the ACM, 49(8).

[14] McDowell, C., Hanks B., and Werner L. (2003), Experimenting with Pair programming in the Classroom, ITiCSE'03, Thessaloniki, Greece.

[15] Bloom, B., Englehart M, Furst E, Hill W, and Krathwohl D. (1956), The Taxonomy of Educational Objectives: The Classification of the Educational Goals. Handbook I: The Cognitive Domain. McKay Press, New York.

[16] Cao, L. and Xu, P. (2005), Activity Patterns of Pair Programming, Proceedings of the 38th Hawaii International Conference on System Sciences.

[17] Nosek, J. T.(1998), "The case for collaborative programming," Communications of the

ACM, 41, pp. 105-108.

[18] Srikanth, H., Williams, L., Wiebe, E., Miller, C., and Balik, S., On Pair Rotation in the Computer Science Course, Conference on Software Engineering Education and Training 2004, pp. 144-149.

[19] Johnson, C.G. and Fuller, U. (2006), Is Bloom's taxonomy appropriate for computer science? Proceedings of the Sixth Baltic Sea Conference on Computing Education Research (Koli Calling 2006), pp. 120-123.

[20] He, L. and Brandt, P. (2007), WEAS: A Web-based Educational Assessment System, ACMSE 2007, Winston-Salem, N. Carolina, USA, pp. 126-131.

[21] Lister, R. and Leaney, J. (2003), Programming, Criterion-Referencing, and Bloom, SIGCSE'03, February 19-23, Reno Nevada USA, pp. 143-147.

[22] Daigle, R. J. and Doran, M.V. (1998), Facilitating Bloom's Level One through Active Learning and Collaboration, Journal of Information Systems Education, 9(3), pp.3-6.

[23] Scott, T. (2003), Bloom's taxonomy applied to testing in computer science classes, Journal of Computing Sciences in Colleges, 19(1), pp. 267-274.

[24] Whalley, J.L., Lister, R. , Thompson, E., Clear, T., Robbins, P., Ajith Kumar, P. K., and Prasad, C. (2006), An Australasian study of reading and comprehension skills in novice programmers, using the bloom and SOLO taxonomies, Proceedings of the 8th Austalian conference on Computing education, pp. 243-252.