

# 美國高中的先修大學電腦科學課程(APCS)簡介

高慧君 吳正己

國立臺灣師範大學 資訊教育學系

## 摘要

APCS 是美國高中的先修大學電腦科學課程，每年由美國教育測驗服務社(ETS)定期舉辦全國性的測驗，藉由選修 APCS 課程，學生除了可以試探自己的性向以供未來就讀相關科系的參考外，也可以做為抵免大學學分之用。本文介紹 APCS 的課程內容、程式語言選用、測驗方式與學分抵免等，以做為我國高中電腦課程實施的參考。

## 壹、前言

1950 年代初期，美國的大學認為高中教育缺乏具挑戰性的課程，使得許多資優學生無法得到適當的發展。當時，由俄亥俄州的 Kenyon College 校長發起了 The Kenyon Plan，召集 12 所大學與 12 所高中成立聯合工作小組，規劃一套相當於大一階段的先修課程，並選擇 11 個科目在 18 所高中試辦。由於實施成效不錯，1954 年 The Kenyon Plan 正式由大學委員會(the College Board)接辦，定名為先修大學課程(The Advanced Placement Program，簡稱 AP)；為使 AP 課程具有客觀一致的評量標準，大學委員會委託教育測驗服務社(Education Testing Service，簡稱 ETS)舉辦全國性的 AP 測驗。經過四十餘年的發展，目前 AP 課程共設有 19 種學門、35 個科目(見表一)，每年報考 AP 測驗

總人數高達百萬人次。以 1998-99 學年為例，約 70 萬名高中生報考，測驗人次總計約 115 萬個(The College Board, 2001)。隨著報考 AP 測驗的人數逐年增加，美國大學新生中具有 AP 測驗合格成績的人數也逐年增加。以美國麻省理工學院為例，1998 年入學的新生中，約有七成新生獲得抵免 6 學分以上，其中很多均以 AP 成績抵免學分(MIT, 2000)。以下若無特別註明，有關 APCS 之內容均參考 The College Board 的 AP 網站(<http://cbweb1.collegeboard.org/ap/>)。

美國前任教育部長 Riley (1999)指出：「兩年後，每一所高中都應該提供大學先修課程或其他高級課程，而且在三至五年內提供更多大學先修課程選修科目，目前雖有 49% 的高中提供這類課程，但只有 10% 的學生選讀。」為使更多學生能受惠於 AP 的課程，美國聯邦教育部於 1999 年度提撥三百萬美元，做為低收入戶學生報考 AP 測驗的費用補助，2000 年提高為四百萬美元，AP 課程自實施以來漸漸地成為引領高中生認識大學課程的有效策略。

AP 課程於 1984 年開始提供先修大學電腦科學課程(Advanced Placement Computer Science，以下簡稱 APCS)，1988 年起 APCS 分為 A 級和 AB 級兩種(以下簡稱 CS-A 和 CS-AB)。CS-A 主要著重問題解決方法的學

## 美國高中的先修大學電腦科學課程(APCS)簡介

習，內容包括程式設計、基本的資料結構及演算法等；CS-AB 除了包含 CS-A 的所有內容外，還有較深入的演算法、資料結構、資料抽象化等內容。

表一 AP 的學門領域及科目名稱

學門領域	科目名稱
Art	Art History, Studio Art: Drawing Portfolio, Studio Art: 2-D Portfolio, Studio Art: 3-D Portfolio
Biology	Biology
Calculus	AB, BC
Chemistry	Chemistry
Computer Science	A*, AB
Economics	Macroeconomics*, Microeconomics*
English	Language and Composition, Literature and Composition, International English Language(APIEL)
Environmental Science	Environmental Science*
French	Language, Literature
German	Language
Geography	Human Geography*
Government and Politics	Comparative*, United States*
History	European, United States, World
Latin	Literature, Vergil
Music	Music Theory
Physics	B; C: Electricity and Magnetism*, C: Mechanics*
Psychology	Psychology*
Spanish	Language, Literature
Statistics	Statistics*

\*為一學期課程，其餘為一學年課程

### 貳、APCS 課程內容

APCS 的課程是由大學教授、高中教師、大學委員會代表、測驗服務社代表等共同組成的委員會決定，課程內容相當於大學資訊科系的入門課程，其教學目標是希望學

生：

1. 能以電腦解決一些應用領域的問題。
2. 能了解基本的演算法與資料結構。
3. 能選擇適當的演算法與資料結構來解題。
4. 能熟練地運用 C++ 程式語言。

5. 能閱讀和理解大型程式及其說明文件。
6. 能了解電腦系統的主要硬體單元及其工作原理。
7. 能了解電腦倫理與社會議題。

CS-A 及 CS-AB 的課程內涵係根據上述的目標訂定，二種課程均包括程式規劃、程式實作、程式分析、資訊結構、演算法、電腦系統等六個主題。各主題內涵如下：

#### 1. 程式規劃 (program design)

能夠正確的規劃程式以解決問題。內容包括：定義問題、定義物件(object)及類別(class)、分析資料結構與演算法、設計使用者介面等。

#### 2. 程式實作 (program implementation)

能實際撰寫程式以解決問題。內容包括物件導向及由上而下的程式發展、抽象化的應用、及程式設計的基本結構(如：變數宣告、流程控制、輸出入等)。

#### 3. 程式分析 (program analysis)

能分析及測試程式執行的結果。內容包括程式碼的測試與除錯、演算法的執行效率分析、數值資料極限的了解等。

#### 4. 基本資料結構 (standard data structures)

能使用基本的資料型態，如 C++ 語言中的 integer char bool double、string、structs、arrays、class 等。CS-AB 還比 CS-A 多包括了鏈結串列(linked list)、堆疊(stack)、佇列(queue)、樹(tree)、錐疊(heap)、優先佇列(priority queue)等基本資料結構。

#### 5. 基本演算法 (standard algorithms)

能以基本的演算法設計程式解決問題。內容包括資料結構的巡行(traversal)、插入、刪除等操作，能使用循序搜尋法(sequential

search)及二分搜尋法(binary search)、選擇排序法(selection sort)、插入排序法(insertion sort)、合併排序法(merge sort)、快速排序法(quick sort)等。CS-AB 課程還包括了動態資料結構、記憶體配置操作、及雜湊搜尋法(hashing search)等。

#### 6. 電腦系統 (computer systems)

能了解電腦系統的軟硬體單元及其工作原理，並認識電腦倫理。硬體方面，包括主、輔記憶體、處理器、週邊設備等單元。軟體方面，包括單使用者系統與網路系統、編譯器、作業系統等。電腦倫理方面，包括系統的可信賴度、個人隱私權、智慧財產權、及社會與倫理議題等。

一般的程式語言課程，學生的作業練習的程式都非常的小，僅能用以為語法之說明，而難以解決實際的問題(吳正己、何榮桂，1998)。為了讓學生能夠由解決實際的問題中學習程式設計，APCS 自 1994-1995 學年起提供案例(case study)的教材供教師採用。案例係以實際生活的問題為例，呈現出專家解決問題與構思程式設計的過程；學生由案例教材的學習中瞭解專家解題的過程，並由閱讀及修改專家所寫的程式來學習良好的程式設計風格。而案例所提供的大型程式設計經驗，亦有助於學生發展不同的程式設計策略。1999-2000 學年提供的案例教材為「海洋生物案例研究 (Marine Biology Case Study)」，可以從 APCS 網站免費下載，並附有教師手冊及程式原始碼供參考。

美國高中的先修大學電腦科學課程(APCS)簡介

表二 APCS 課程主題 (ETS, 2001)

CS- A 及 CS-AB 共有的主題	CS-AB 特有的主題
1. Program Design	
1-1 Problem definition	
1-2 Program design	
2. Program Implementation	
2-1 Implementation techniques	
2-2 Program constructs	
2-3 Generic data types and functions (AP classes)	2-3 Generic data types and functions (Templates)
3. Program Analysis	
3-1 Testing	
3-2 Debugging	
3-3 Understanding and modifying existing code	
3-4 Handling errors	
3-5 Reasoning about programs (Pre/post conditions, Assertions)	3-5 Reasoning about programs (Invariants)
3-6 Analysis of algorithms (running times comparison, statement execution count)	3-6 Analysis of algorithms (Big-Oh notation, time and space analysis)
3-7 Numerical limits	
4. Standard Data Structures	
4-1 Simple data types	
4-2 Aggregate data types	
4-3 Classes	
	4-4 Linked lists
	4-5 Stacks
	4-6 Trees
	4-7 Heaps
	4-8 Priority queues
5. Standard Algorithms	
5-1 Operations on data structure	
	5-2 Operations on dynamic
5-3 Searching (Sequential, Binary)	5-3 Searching (Hashing)
5-4 Sorting (Selection, Insertion, Mergesort, Quicksort)	5-4 Sorting (Heapsort)
6. Computer Systems	
6-1 Major hardware components	
6-2 System software	
6-3 Types of systems	
6-4 Responsible use of computer systems	

## 參、APCS 程式語言的選用

APCS 既為先修大學電腦課程，且又可於進入大學後申請抵免大學學分，然而不同的大學選用的程式語言又不盡相同，因此有人建議 APCS 應該配合學生申請大學的需求，提供不同程式語言版本的課程與測驗。APCS 的課程委員會認為 APCS 是全國性實施的課程，應該指定選用單一程式語言，其考量的因素如下(The AP Computer Science Ad Hoc Committee, 2000)：

### 1. 避免教師授課及學生學習的困擾

在大學中同一門課通常只會選用單一程式語言，極少大學會同時選用兩種以上，尤其是在入門課程中，不同程式語言的語法容易造成初學者的混淆。而且如果不指定單一程式語言，那麼教師所選用的程式語言很容易受到當地大學的影響；尤有近者，多數學生計劃申請多所大學，而每所大學選用的程式語言不盡相同，APCS 教師在選用程式語言時會面臨極大的壓力，甚至於被迫於在同一門課中選用兩種以上的程式語言。

### 2. 維持測驗評分的公平性

不同的程式語言有不同的語意邏輯表達方式，有些程式語言專為解決某些特定問題所設計，而且每種程式語言提供的函式庫也不盡相同，同樣的問題以不同的程式語言設計，在評分上很不容易達到一致標準。

### 3. 減輕大學認可 APCS 測驗成績的複雜度

單一的程式語言可以使大學對 APCS 測驗成績的認可單純化，也可以使他們能依學生的 APCS 成績來推定學生的能力背景，以便於規劃學生須再補足或加強的學習主題或

修習科目。

至於 APCS 應該選用哪一種程式語言呢？APCS 課程委員會認為除了該語言必須能夠支援課程中的所有主題外，最重要的，它必須是多數大學資訊科系入門課程中所選用的程式語言。早期各大學多採用 Pascal 語言授課，故而 APCS 課程也是選用 Pascal 語言；後來由於愈來愈多的大學資訊科系於入門課程中選用 C++ 語言，1995 年的課程委員會在廣徵各界意見後決定自 1999 年起 APCS 改選用 C++ 語言。由於 C++ 語言所包含的範圍過於龐大，委員會特別針對課程內容訂定一個 C++ 語言的子集，減低程式語言學習的複雜度，以使學生的學習能更專注於學習主題 (Astrachan, Chapman, Rodger, & Weiss, 1997)。

近幾年來由於愈來愈多的大學於入門課程中選用 Java 語言，根據 APCS 課程委員會在 1999 年調查美國各大學的程式語言選用情形，發現 Java 語言是未來五年內許多大學資訊科系入門的第一門(CS1)及第二門(CS2)課程所將普遍採用的語言(見表三, The AP Computer Science Ad Hoc Committee, 2000)，故而委員會決定自 2003 年起 APCS 課程將改選用 Java 語言。

## 肆、APCS 的測驗

APCS 測驗每年定期於五月舉行，學生們只能自 CS-A 和 CS-AB 擇一報考，而各大學對於測驗成績是否可以抵修大學學分也有不同的規定。

一、題目型式與評分

APCS 測驗的總時間為三小時，測驗題型分為選擇題與程式設計題兩種。選擇題部份共有 40 個題目，作答時間為 1 小時 15 分鐘，

主要評量學生對於各學習主題的精熟程度，表四是選擇題的一個範例，係評量學生對副程式呼叫時參數傳遞觀念的了解。

表三 APCS 課程委員會 1999 年調查美國大學的程式語言選用情形

	C++	C++或 Java	Java	其它	未知
<b>CS1 課程</b>					
目前選用的語言	49		34	17	
3 年後選用的語言	18	14	47	15	6
5 年後選用的語言	8	11	38	8	35
<b>CS2 課程</b>					
目前選用的語言	57		37	6	
3 年後選用的語言	24	16	51	3	6
5 年後選用的語言	13	11	40	2	34

註：寄發問卷約 280 份，回收 100 份

表四 APCS 測驗的選擇題範例

```

以下兩個函數：
Void Myst (int a, int &b)
{
    a *= b;
    b = 2 + a;
}

void Test()
{
    int u = 2;
    int v = 3;
    Myst(u, v);
    Cout << u << " " << v << endl;
}
    
```

- 請問呼叫 Test() 的輸出結果是？
- (A) 2 3
  - (B) 2 8
  - (C) 6 3
  - (D) 6 8
  - (E) 8 3

程式設計題部份有 4 個題目，作答時間為 1 小時 45 分，著重於評量學生推理方面的解決問題能力，包括程式的規劃、撰寫、分

析及文作製作等。程式設計題並非以上機測驗方式進行，而是以徒手寫出程式碼的方式作答，試卷評量也採用人工批閱。閱卷委員

會由大學教授與高中教師共同組成，在每年六月舉行 AP 閱卷會議(AP Reading)，經由反覆的討論訂出一致的評分標準，委員們才開始個別地進行閱卷的工作。表五為 2000 年 APCS 測驗的程式設計題中的一個小題與其

評分標準。本題是要求學生設計一個函式以檢查陣列中的元素值是否符合某個要求的特性，此小題分數為 2 分；如果學生在答案中能進行陣列元素值的比較，即可得 1 分；如果函式的執行結果正確，可再得 1 分。

表五 APCS 測驗的問答題範例與評分標準

問題

1. 當一個陣列元素之值同時大於其上一個及下一個陣列元素值時，則我們稱作這個陣列元素值為 mode。換句話說，mode 發生在以下情況：k 為陣列 A 的索引值，當  $A[k] > A[k-1]$  且  $A[k] > A[k+1]$  時，A[k] 之值為 mode。

當一個陣列的元素值從第一個起開始遞增，並且遇到一個 mode 之後即開始遞減，則我們稱作這個陣列為 unimodal 陣列。unimodal 陣列中只會出現一個 mode。例如，下列的陣列 A 是 unimodal 陣列，而其 mode 出現在索引值為 4 之處。我們並且假定 mode 不能是陣列中第一個或最後一個元素。

Index k	A [k]
0	3
1	5
2	9
3	10
4	12 <--- mode
5	11
6	9
7	4

- (a) 寫一個函式 IsMode，如下列所宣告。如果 data[k] 大於 data[k-1] 且 data[k] 大於 data[k+1] 時，IsMode 將傳回 true，否則傳回 false。在上例的情況中，呼叫函式 IsMode(A,4) 將傳回 true，而呼叫函式 IsMode(A,5) 將傳回 false。

請完成下列的 IsMode 函式。

```
bool IsMode (const apvector<int> & data, int k)
// 傳入參數值之限制：0 < k < data.length() - 1
(本題共有 a, b, c 三個小題，以下省略後兩個小題)
```

評分標準

- (a) 本題 2 分

- +1 至少答案中有進行下列嘗試：對陣列之元素值進行比較，用 >= 運算子是可被接受的，且必須使用 || 運算子 (AND 運算子) 而不能使用 && 運算子 (OR 運算子)。
- +1 答案是正確的 (用 1 和 0 來分別代表 true 和 false 是可被接受的)

註：如果迴圈中的程式碼沒有使用到 k，或者改變了 k 之值，則以零分計。  
(以下省略後兩個小題的評分標準)

表六 2000 年 APCS 測驗的成績統計

測驗成績		百分比	
		CS-A	CS-AB
特優	5	14.4	34.1
優	4	25.7	12.0
合格	3	19.4	26.8
接近合格	2	9.3	10.5
不合格	1	31.3	16.6
平均		2.83	3.36
標準差		1.46	1.46
應試人數		13,646	6,876
參與學校數		2,033	1,271

表七 史丹佛大學的學分抵免規定

測驗科目	測驗分數	抵免課程代號 (學分數)
Computer Science A	4,5	CS106X (5)
Computer Science AB	4,5	CS106X (5)

表八 美國德州奧斯汀大學 APCS 抵免學分規定

測驗科目	測驗分數	抵免課程代號 (成績登錄*)
Computer Science A	5	CS303E (A)
	4	CS303E (B)
Computer Science AB	5	CS303E (A) 和 CS315 (CR)
	4	CS303E (B) 和 CS315 (CR)
	3	CS303E (C)

\* A, B, C 為抵免後登錄的成績等級，CR 為給予學分但無成績。

## 二、測驗成績

學生的 AP 測驗成績會被轉換為 AP 的 5 分制，5 分表示「特優 (extremely well qualified)」、4 分表示「優 (well qualified)」、3 分表示「合格 (qualified)」、2 分表示「接近合格 (possibly qualified)」、1 分表示「不合格 (no recommendation)」。表六是 2000 年 APCS 測驗的統計資料，共有來自 2,033 所高中的 13,646 個學生報考 CS-A 測驗，平均得分為 2.83 分；來自 1,271 所高中的 6,876 個學生報考 CS-AB 測驗，平均得分為 3.36 分。

## 三、抵免大學學分

測驗成績是否可以抵免大學學分，各大學間有不同的規定。大部分的大學接受 3 分以上 AP 測驗成績的抵免，例如柏克萊大學、加州洛杉磯大學，德州奧斯汀大學等；有些大學採取較嚴格的規定，只接受 4 分以上的 AP 測驗成績，例如麻省理工學院、史丹佛大學、普林斯頓大學、哥倫比亞大學等。有些大學則不願意接受 AP 測驗成績，例如哈佛大學和加州理工學院是採用學校自辦的測驗來認可學分的抵免 (MIT, 2000)。表七是史丹佛大學接受以 APCS 測驗成績抵免課程的規定舉隅，獲得 APCS 測驗 4 分以上的學生可以抵免 CS106X 等基礎課程，直接選修較進階的課程 (Stanford, 2000)。表八是德州奧斯汀大學接受以 APCS 測驗成績抵免程式設計基礎課程中的前二門 CS303E, CS315 課程，並且可以依測驗分數轉換登錄為該科成績，如 APCS 的 5 分可登錄為 CS303E 的 A，或 CS315 的 CR (表示僅給學分，但無成績) (UT Austin, 2001)。

## 伍、結語

我國高中電腦課程列為選修科目，雖設有部頒的課程標準，但由於各校課程實施存在著許多差異及缺乏全國性的測驗，對於有興趣鑽研電腦科學領域的學生往往無法獲得適當的引導。目前美國高中雖然也沒有將電腦科學列入必修科目，但是高中生可以藉由選修 APCS 課程了解電腦科學領域的知識，其測驗成績不僅可以做為申請大學的參考資料，也可以抵免入學後的基礎課程學分。APCS 歷經十五年的發展，其課程大綱、教材資源、程式語言選用、測驗方式等，頗值得做為我國高中電腦課程規劃與實施的參考。

## 參考文獻

1. 吳正己 (1993)：美國 ACM 1993 年中學電腦課程簡介 兼談我國中等學校電腦課程之規劃。中等教育，44 卷，6 期，65-72 頁。
2. 吳正己、何榮桂 (1998)：高級中學新訂電腦課程的內涵與特色。科學教育月刊，208 期，26-32 頁。
3. 教育部 (1996)：高級中學電腦課程標準。台北：作者。
4. Astrachan, O. L., Chapman, G., Rodger, S. H., & Weiss, M. A. (1997). The reasoning for the Advanced Placement C++ subset, *ACM SIGCSE Bulletin*, 29(4), 62-65.
5. Ben-Ari, M. & Henney, K. (1997). A critique of the Advanced Placement C++ subset, *ACM SIGCSE Bulletin*, 29(2), 7-10.

## 美國高中的先修大學電腦科學課程 (APCS)簡介

6. Education Testing Service (ETS) (2001). Advanced Placement course description: Computer science. Retrieved November 20, 2001, from [http://cbweb2s.collegeboard.org/ap/pdf/ap\\_compsci01.pdf](http://cbweb2s.collegeboard.org/ap/pdf/ap_compsci01.pdf)
7. Riley, R.W. (1999). Changing the American high school to fit modern times. Retrieved November 20, 2001, from <http://www.ed.gov/Speeches/09-1999/990915.html>
8. Stanford (2000). Degree requirements. Retrieved November 20, 2001, from <http://www.stanford.edu/dept/registrar/bulletin/bulletin00-01/pdf/DegreeRequirements.pdf>
9. The AP Computer Science Ad Hoc Committee (2000). Recommendations of the AP Computer Science Ad Hoc Committee. Retrieved November 20, 2001, from [http://cbweb2s.collegeboard.org/ap/pdf/ad\\_hoc\\_report\\_surveys.pdf](http://cbweb2s.collegeboard.org/ap/pdf/ad_hoc_report_surveys.pdf)
10. The College Board (2001). Advanced Placement Program. Retrieved November 20, 2001, from <http://cbweb1.collegeboard.org/ap/>
11. The Massachusetts Institute of Technology (MIT) (2000). Freshman pass/no record grading and Advanced Placement policy. Retrieved November 20, 2001, from <http://web.mit.edu/faculty/reports/pnrap/subcommittee.html>
12. The University of Texas at Austin (UT Austin) (2001). College Board Advanced Placement examinations: Satisfactory scores and course credit 2001-2002. Retrieved November 20, 2001, from <http://www.utexas.edu/academic/mec/cbe/cutape.html>