

APCS 大學程式設計先修檢測

在這個 APCS 測驗中，我取得了 6 分（觀念題 3 分、實作題 3 分）。這對於不會演算法與對 C 語言不熟悉的我而言，其實已經是不錯的成績了，但在這次考試後，我才意識到演算法究竟有多重要，也讓我認知道了還有許多東西是我需要學習的。



大學程式設計先修檢測成績證明

蔡淳宇

臺北市市立陽明高中

准考證號：114019652

身分證號：[REDACTED]

檢測日期：2025年01月05日

科目	原始總分	級別
程式設計觀念題	68	第三級
程式設計實作題	205	第三級

檢測成績級別說明

程式設計觀念題 檢測人數2729人			程式設計實作題 檢測人數2739人			
級別	原始總分範圍	百分比*	級別	原始總分範圍	百分比*	說明
五	90~100	5.6	五	350~400	0.1	具備常見資料結構與基礎演算程序運用能力
四	70~89	34.3	四	250~349	1.9	具備程式設計與基礎資料結構運用能力
三	50~69	32.6	三	150~249	43.6	具備基礎程式設計與基礎資料結構運用能力
二	30~49	19.4	二	50~149	27.6	具備基礎程式設計能力
一	0~29	8.1	一	0~49	26.9	尚未具備基礎程式設計能力

* 該次檢測人數百分比（四捨五入取概數到小數第一位）

APCS 大學程式設計先修檢測

參加動機和訓練過程

我自高中自學程式後，就被學校同學與老師推薦去參加了 APCS，而我之所以會參加，也是想藉著這樣一個免費的測驗，看看自己的程式能力到底如何。我第一次參加時，程式能力並不是很好，因此實作題一題都沒解出來，只拿到了 1 級分。經過了這一次的經驗，我開始到 [ZeroJudge](#) 等平台上練習實作題的部分，並去網路上找一些 [觀念題的教學](#)。經過一段時間的學習，我再度參加了 APCS 測驗，並取得了總分 6 級分的成績。雖然已取得明顯進步，但我仍然認為自己還有許多進步空間，我希望未來我可以繼續學習如 [演算法](#) 和 [資料結構](#) 等程式技術，努力將自己的實力提升到能達到實作 5 級分的程度。

ZERO JUDGE		分類題庫	解題動態	排行榜	討論區	競賽區	搜尋題目關鍵字、題號...	Q	淳 0
	✓	a005. Eva 的回家作業 – POJ	95%	/62472 人	迴圈	2007-04-22	2023-09-04		
6	✓	a006. 一元二次方程式	94%	/55333 人	數學	2007-04-22	2020-01-02		
7	✓	a009. 解碼器 – ACM 458	95%	/41232 人	字元處理	2007-04-25	2020-05-21		
8	✓	a010. 因數分解	92%	/29953 人	數論 質因數分解	2007-04-26	2021-03-22		
9		a013. 羅馬數字 – NPSC 模擬試題	87%	/12645 人	進位制	2007-10-05	2015-12-31		
10	✓	a015. 矩陣的翻轉	94%	/30303 人	陣列	2007-10-05	2015-12-31		
11		a017. 五則運算	80%	/8220 人	parser stringstream	2007-10-05	2020-01-03		
12	✓	a020. 身分證檢驗	93%	/23920 人	流程控制	2007-10-17	2020-05-25		
13	✓	a021. 大數運算	80%	/8842 人	大數 陣列	2007-10-05	2021-02-25		
14	✓	a022. 迴文	96%	/29300 人	字串 迴文	2007-10-11	2021-03-24		
15	✓	a024. 最大公因數(GCD)	93%	/37994 人	GCD 數學 最大公因數 迴圈	2007-10-18	2020-04-22		
16	✓	a034. 二進位制轉換	95%	/27089 人	迴圈	2007-10-18	2020-01-03		
17	✓	a038. 數字翻轉	92%	/29213 人	while 迴圈	2007-11-01	2020-05-25		
18	✓	a040. 阿姆斯壯數	95%	/17381 人	數學 窮舉	2008-02-13	2021-04-16		
19	✓	a042. 平面圖形切割 – 許介彥	97%	/20250 人	數學	2008-03-13	2020-01-03		
20	✓	a044. 空間切割 – 許介彥	90%	/17621 人	數學 遞迴	2008-03-13	2020-01-03		

第一頁 | 《- 上一頁 | 第 1 頁 | 下一頁 -》 | 最末頁

APCS 大學程式設計先修檢測 解題語言與實作例題

由於我從高中開始就專注在學習 Python 了，因此我選擇使用 Python 作為我寫實作題時用的語言，Python 作為目前全球最紅的程式語言，雖然執行效率不如傳統編譯型語言來得高，但它以其簡單、美觀與強大的功能而聞名，它也是目前多數人在開發和研究 AI 時會用的主要語言。我之所以選擇它來解題，也是因為它對於字串控制語法與部分內建函式庫都較 C/C++ 等語言方便。

以下是一個程式的題目範例與解答，我先將輸入的算式全部翻轉過來之後，用一個迴圈迭代所有算式，由於 Python 對遞迴效率不佳，我使用 `stack` 來存放與處理未計算的數值，以此處理不同函數的運算處理及計算優先順序。

內容

給一個運算式，運算式的內容由數字、+、* 和 某個函式 $f()$ 所組成，除了函式 $f()$ 以外不會有額外的括號。請將此運算式依照先加後乘的方式運算。
函式 $f(x_1, x_2, x_3, x_4, \dots)$ 定義為從這個不定長度的參數 $x_1, x_2, x_3, x_4, \dots$ 中的最大值扣掉最小值。例如 $f(3, 6, 2) = 6 - 2 = 4$ 、 $f(3) = 0$ 。

▲ ZeroJudge j607. (取自 2023 年 1 月 APCS 實作第三題)

```

1 def high(i): #最高層的運算：(*) > (+)。“處理乘法鍵”
2     v, i = mid(i)
3     while i < len(inp) and inp[i] == '*':
4         n, i = mid(i + 1)
5         v *= n
6     return (v, i)
7
8 def mid(i): #加法層級運算：(+) 比 (*) 晚處理。“處理加法鍵”
9     v, i = low(i)
10    while i < len(inp) and inp[i] == '+':
11        n, i = low(i + 1)
12        v += n
13    return (v, i)
14
15 def low(i): #最基礎的運算：數字直接回傳、f(...) 進括號並 call high()。“處理最小單位”
16    if inp[i].isdigit():
17        n = 0
18        while i < len(inp) and inp[i].isdigit():
19            n = n * 10 + int(inp[i])
20            i += 1
21        return (n, i)
22    elif inp[i] == 'f':
23        i += 2 #跳過 "f("
24        args = []
25        while True:
26            v, i = high(i)
27            args.append(v)
28            if inp[i] == ',':
29                i += 1
30            else:
31                break
32        i += 1 #跳過 ")"
33        return (max(args)-min(args), i)
34    else:
35        raise ValueError("Unexpected character: " + inp[i])
36
37 inp = input().strip()
38 ans, _ = high(0)
39 print(ans)

```

▲ 我的 Python 解法 (含註釋)