

壹、 競賽成果、特殊表現證明



大學程式設計先修檢測成績證明(歷次)

游展勳

身分證號：A130892482

臺北市市立大同高中

檢測日期：2019年10月26日			
科目	原始總分	級別	備註
程式設計觀念題	48	第二級	該等級(含)以上占該次檢測人數81.5%
程式設計實作題	100	第二級	該等級(含)以上占該次檢測人數50.7%

檢測日期：2020年01月05日			
科目	原始總分	級別	備註
程式設計觀念題	64	第三級	該等級(含)以上占該次檢測人數42.1%
程式設計實作題	100	第二級	該等級(含)以上占該次檢測人數41.4%

檢測日期：2021年01月09日			
科目	原始總分	級別	備註
程式設計觀念題	88	第四級	該等級(含)以上占該次檢測人數15.7%
程式設計實作題	90	第二級	該等級(含)以上占該次檢測人數58.7%

檢測成績級別說明

程式設計觀念題		程式設計實作題		
級別	原始總分範圍	級別	原始總分範圍	說明
五	90~100	五	350~400	具備常見資料結構與基礎演算法程序運用能力
四	70~89	四	250~349	具備程式設計與基礎資料結構運用能力
三	50~69	三	150~249	具備基礎程式設計與基礎資料結構運用能力
二	30~49	二	50~149	具備基礎程式設計能力
一	0~29	一	0~49	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)

Page: 1/1

申請日期：2021年04月06日

圖一、歷次 APCS 檢測成績證明

錯誤! 找不到參照來源。為歷次參加 APCS 檢測的成績。第一次報考 APCS 時，僅有高資訊課所擁有的實力，後來經過選修校內課程 C++進階程式設計後，於第二次報考 APCS 時，觀念題進步 16 分，上升一級，但仍然只算是及格而已。為了真正學好 C++這個程式語言，我參加國立臺北科技大學招開的程式設計推廣教育課程，原因是學校所教的內容，不足以在 APCS 檢測中取得很高的成績，所有關於指標的題目都非常陌生。在上課數周之後，在課堂上學到的新觀念、新語法，遠遠超出預期，對於整個程式的運作方式有更清晰的理解，且對理論基礎有很大的幫助，因此得以在觀念題中取得相當滿意的分數。實作部分由於缺乏練習，成績較不出色，因此在考完之後，規劃實作題的練習與加強。其一透過選擇每天花些時間至 ZeroJudge 網站自主練習實作，維持實作的邏輯思維；其二則是校外課程的輔助，並於這學期修讀 C++進階課程資料結構與演算法，學習更多的演算法以利實作題運用，希望能藉此提升自己的實作能力。



國際運算思維挑戰賽

International Challenge on Informatics and Computational Thinking

挑戰證明

臺北市 市立大同高中 游旻勛 同學 (學號：10735145)
參加 2019 年國際運算思維挑戰賽 十一、十二年級 組
獲得 285/300 分 (全國 PR 98)


李忠謀教授
國立臺灣師範大學資訊工程學系

圖二、國際運算思維挑戰證明

錯誤！找不到參照來源。為第二次參加國際運算思維挑戰的成績，高一時考的成績為全國 PR68，並不出色，在經過一整年的學習之後，成功於高二時得到全國 PR98 的好成績，相信擁有好的邏輯思維能力能夠在學習程式邏輯上，或是學習更多更深入的理論時，加快理解的速度。



圖三、國立臺北科技大學推廣教育修讀證明書

圖三是我修讀程式先修課程的證明書，起初選擇報名這個課程的目的是為了學習更多學校沒有教完的內容，例如函式與指標，到了真正開始上課之後，才發現學校教的內容也不夠詳細，大部分都只是教語法的使用而已，對於程式內部的運作理論都沒有提及，在經過前面幾堂課的複習，改變了之前對於整個 C++ 以及程式運作概念的理解，在後續幾堂課也陸續學習到物件導向程式設計的概念，因此修讀完這門課之後，我擁有很大的收穫。

```

C:\Users\andre\Desktop\Git資料庫\PublicLibr...
Microsoft Windows [版本 10.0.19042.867]
(c) 2019 Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\andre>C:\Users\andre\Desktop\Git資料庫\PublicLibrary\
MyProjects\排座位\排座位.exe
===== 207-隨機座位 =====

按Enter以啟動程式...

===== 設定視角 =====

請選擇視角：(輸入數字1或2)
(1)看向講台
(2)看向桌椅

回答：1

===== 設定模式 =====

請選擇顯示方式：(輸入數字1或2或3)
(1)同時顯示號碼
(2)逐一顯示號碼
(3)按Enter鍵顯示號碼

回答：1

===== 設定完成 =====

===== 開始執行 =====

                                [講台]

3          31          17          20          4          19          24
23         25         42         39         16         18         14
38          2          35         28          9          34         26
30         22          8          7          6          33         15
40         27         21         11         13         36          1
          29         12         37         32         43

===== 執行完畢 =====

輸入"again"可再次產生新座位
輸入"reset"可重新設定
輸入"end"可結束程式

回答：

```

圖四、隨機換座位程式(執行結果)

隨機換座位是我寫出來的第一個小工具，靈感來自於班導師利用iGeogebra製作的換座位工具，想嘗試利用當時有學過的基本C++語法，自己編寫一個能同樣達到隨機換座位效果的程式。圖四為程式的執行結果，由圖可見裡面有編寫一些特殊功能，如可以改變視角，控制講台在上方或是下方，也可以改變顯示模式，增加隨機換座位的趣味。

```

206 void fn_SetSeatNum()
207 {
208     for(int i = 1; i <= 39; i++)
209     {
210         int j;
211
212         do
213         {
214             do
215             {
216                 int_SeatNum[i-1] = rand()%43 + 1;
217             }
218             while(int_SeatNum[i-1] == 41 ||
219                 int_SeatNum[i-1] == 10 ||
220                 int_SeatNum[i-1] == 20 ||
221                 int_SeatNum[i-1] == 5);
222
223             for(j = 1; j < i; j++)
224             {
225                 if(int_SeatNum[i-1] == int_SeatNum[j-1]) break;
226             }
227         }
228         while(j != i);
229     }
230
231     return;
232 }

```

圖五、隨機換座位程式(程式碼)

圖五為隨機換座位程式中用來設定隨機座號的函式，當時遇到的第一個問題是我們班的座號並不完全連續，因此需要編寫一個 do-while 迴圈來確保產生的隨機座號存在，並且在外層同樣使用 do-while 迴圈，確保產生的座號沒有重複，最後存入陣列。

```

234 void fn_FillNum(int int_Num)
235 {
236     int int_Count = 0;
237
238     switch(int_Num)
239     {
240         case 1:
241             for(int i = 0; i < 6; i++)
242             {
243                 for(int j = 0; j < 7; j++)
244                 {
245                     if(i == 0 && j == 3)
246                     {
247                         intarr_v1[0][3] = 20;
248                         continue;
249                     }
250
251                     if((i == 5 && j == 0) || (i == 5 && j == 6))
252                     {
253                         std::cout<< '\t';
254
255                         continue;
256                     }
257
258                     intarr_v1[i][j] = int_SeatNum[int_Count];
259                     int_Count++;
260                 }
261             }
262
263             break;
264

```

圖六、隨機換座位程式(程式碼)

編寫這個程式遇到的第二個問題是 20 號學生有弱視的問題，因此座位被固定在講桌前面那一格，因此當我建立二維陣列來模擬班上桌椅的橫排與直排時，需要跳過講台前那一格，自行設定座號，如圖六中的 Line 247，若座位是空的則直接跳過那一格，如圖六中的 Line 253。


```

C:\Users\andre\AppData\Local\Programs\Python\Python38-32\python.exe
*Choose the number of files to open it

(1) Unit5.txt
(2) WordBox.txt
2
*File import successfully!

*Input a number of the options
(1)Add new words
(2)Remove words
(3)Inspect the file
(4)Clean file
(5)Save contents
(6)Return to lated archive
(7)Vocabulary test
(8)Change file
(9)End the program
7

=====

*Test Start!
1. a_____e n. 青少年時間
adolescence
*Correct!    Score:1

-----

2. m_____e v. 使現代化
modernize
*Correct!    Score:2

-----

3. b_____e a. 離奇的
bizzare
*Wrong answer!

Correct answer:bizarre
Life:9    Score:2

```

圖七、背單字程式(執行結果)

背單字程式是於高三上學期的時候編寫的程式，同時也是目前唯一一個利用 Python 編寫且具有完整功能的程式，靈感來自各類英文單字軟體，建立專案放入各個自選的英文單字 txt 檔，並利用 Python 的 Read file 與 Write file 功能，以及字串、字元的比對，達成能夠自選單字庫來考試的功能，不過因為沒有特別學習 Read file 和 Write file 的語法，呈現這些單字功能的方式有些簡陋，因此希望在未來學習完整學習 Python 後，能夠將這份專案以更有效率的方式重新編寫。

```

28 #Pack all contents in lines into a list
29 def ScanFile(fname, lstname):
30
31     with open(fname) as f:
32         for line in f:
33             lstname.append(line.strip())
34
35     return lstname
36
37 #Renew the file
38 def RenewFile(fname, lstname):
39
40     with open(fname, 'w') as f:
41         for lst in lstname:
42             f.write(lst + '\n')
43
44 #Pack all contents in words into a list
45 def ScanWord(fname, lstname):
46
47     with open(fname, 'r') as f:
48         for line in f:
49             item = [i for i in line.split()]
50             lstname.append(item)
51
52     return lstname

```

圖八、背單字程式(程式碼)

利用 Read file 功能，逐行讀取單字並存入陣列裡。如果有新增單字或刪除單字，則利用 Write file 功能將檔案重新寫入陣列裡的所有單字。


```
while True:

    if life == 0:

        print('\n*Test Over!\n Please practice more!\n\n' + 30 * '=' + '\n')

        break

    elif count == len(numlst):

        print('\n*Congratulations!\n You have answered all the questions!'
              + '\n Test Over!\n\n' + 30 * '=' + '\n')

        break

    else:

        rnum = numlst[count]-1

        rword = words2[rnum][0]
        lst = list(rword)

        for i in range(1, len(lst) - 1):
            lst[i] = '_'

        sub = str()

        for i in range(0, len(lst)):
            sub += str(lst[i])

        print(str(count+1) + '. ' + sub + ' ' + words2[rnum][1] + ' ' + words2[rnum][2] + '\n')

        ans = str(input())

        if ans == rword:

            count += 1
            count2 += 1

            print('\n*Correct!   Score:' + str(count2) + '\n\n' + 30 * '-' + '\n')

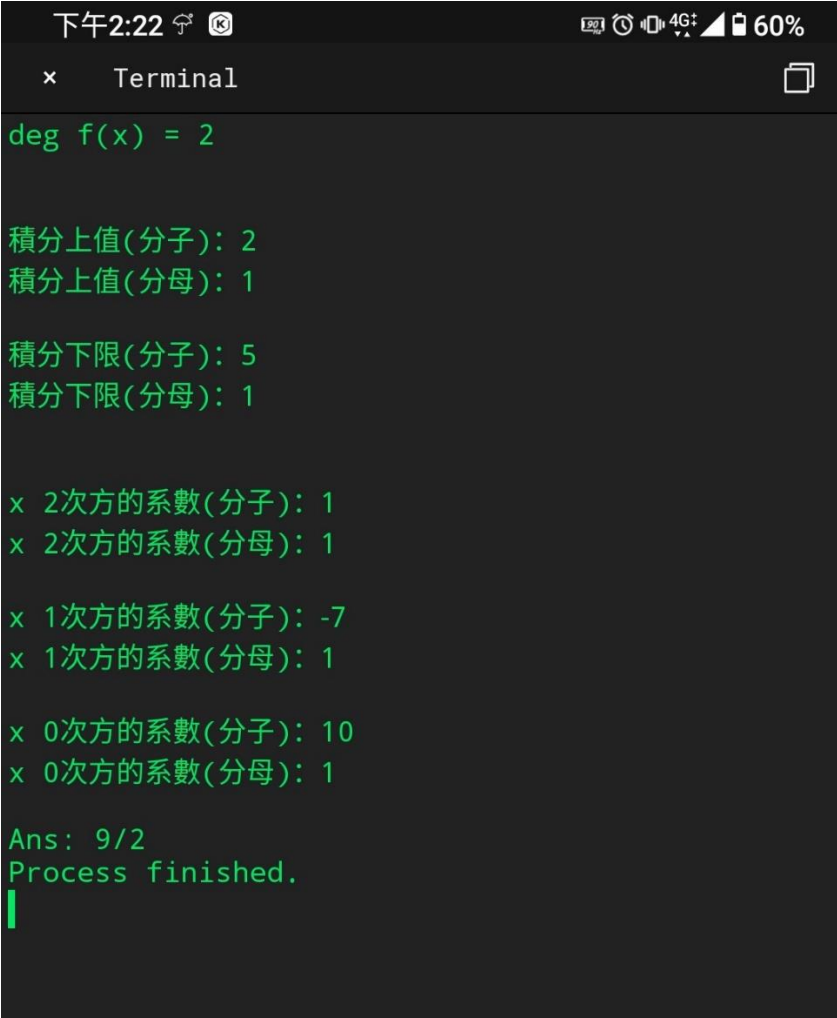
        else:

            life -= 1
            count += 1

            print('\n*Wrong answer!\n\n Correct answer:' + rword + '\n Life:' + str(life)
                  + '   Score:' + str(count2) + '\n\n' + 30 * '-' + '\n')
```

圖九、背單字程式(程式碼)

圖九為單字測驗模式的程式碼，將檔案裡的單字、詞性以及字義分開儲存於二維陣列裡，再將使用者輸入的字串跟陣列裡的單字進行比對，即可達到考單字的效果。



```

下午2:22
deg f(x) = 2

積分上值(分子): 2
積分上值(分母): 1

積分下限(分子): 5
積分下限(分母): 1

x 2次方的系數(分子): 1
x 2次方的系數(分母): 1

x 1次方的系數(分子): -7
x 1次方的系數(分母): 1

x 0次方的系數(分子): 10
x 0次方的系數(分母): 1

Ans: 9/2
Process finished.

```

圖十、定積分計算機(執行結果)

圖十為定積分計算機的輸入與輸出，輸入分為三個部分，第一部分是積函數的最高次數；第二部分是積分的上、下限；第三部分是積函數的各項係數，為了增加實用性，因而設計使用者可以輸入分數係數，得到的答案也會以分數來表示。

```

8  struct Coefficient // 係數結構
9  {
10     int int_Numer = 0;    //分子
11     int int_Denom = 1;    //分母
12
13     void operator-= (Coefficient &coef_Num) //定義此結構的複合指定減號運算子
14     {
15         int int_TempNumer = coef_Num.int_Numer;
16         int int_TempDenom = coef_Num.int_Denom;
17
18         int int_LCM = fn_LCM(this->int_Denom, coef_Num.int_Denom);
19
20         this->int_Numer *= int_LCM / this->int_Denom;    // 通分過程
21         this->int_Denom = int_LCM;
22         coef_Num.int_Numer *= int_LCM / coef_Num.int_Denom;
23         coef_Num.int_Denom = int_LCM;
24
25         this->int_Numer -= coef_Num.int_Numer; // 分數相減
26
27         int int_GCD = fn_GCD(this->int_Numer, this->int_Denom);
28
29         this->int_Numer /= int_GCD;    // 約分過程
30         this->int_Denom /= int_GCD;
31
32         coef_Num.int_Numer = int_TempNumer;
33         coef_Num.int_Denom = int_TempDenom;
34     }
35 };

```

圖十一、定積分計算機(程式碼)

圖十一的內容為這個定積分計算程式中最重要的部分，目的是為了可以輸入分數的係數，畢竟在求反導函數的時候，常常出現係數是分數的狀況，因此編寫了這個係數結構，內含分子及分母兩個整數欄位，在計算時則將分子以及分母分開運算，通分時運用 `fn_LCM` 求最小公倍數；約分時運用 `fn_GCD` 求最大公因數，則可以保證每次進行運算後，各個係數都會化簡至最簡分數。

```

45  int main()
46  {
47      std::cout<< "deg f(x) = ";
48      std::cin>> int_Deg;
49
50      Coefficient* coefptr_Max = new Coefficient;    // 上值
51      Coefficient* coefptr_Min = new Coefficient;    // 下值
52      Coefficient* coefptr_Arr = new Coefficient[int_Deg + 2];    // 各係數
53
54      fn_Preset(coefptr_Max, coefptr_Min, coefptr_Arr);
55      fn_Antiderivative(coefptr_Arr);
56      fn_Calculate(coefptr_Max, coefptr_Arr, &coef_MaxRes);
57      fn_Calculate(coefptr_Min, coefptr_Arr, &coef_MinRes);
58
59      delete coefptr_Max;
60      delete coefptr_Min;
61      delete [] coefptr_Arr;
62
63      coef_MaxRes -= coef_MinRes; // 上函數值 - 下函數值
64
65      if(coef_MaxRes.int_Denom == 1) std::cout<< "Ans: " << coef_MaxRes.int_Numer;
66      else std::cout<< "Ans: " << coef_MaxRes.int_Numer << '/' << coef_MaxRes.int_Denom;
67
68      return 0;
69  }

```

圖十二、定積分計算機(程式碼)

圖十二為定積分計算機的主函式程式碼，其中我練習利用動態記憶體來實作這個定積分計算機，並利用函式進行輸入積分各項目數值、將積函數轉為反導函數，最後將上、下限的值代入反導函數，得出計算結果，若是分數則以分數形式輸出。

class 類別

是一種自定義資料型別

主要著重在方法設計與延伸

包含資料成員與成員函式

類別定義必須利用存取標籤 (access label) `public` 或 `private` 將成員的權限歸類

- | | | |
|--------------------------------------|---|---|
| • <code>Public</code>
可隨意在類別外部做存取 | • <code>Private (C++ 預設)</code>
只能在類別內部做存取 | • <code>Protected</code>
允許在子類別內部做存取 |
|--------------------------------------|---|---|

this 指標

- `this->`
- 用來識別參數名稱與資料成員名稱

💡 參數名稱與資料成員名稱相同時，要明確使用 `this` 指標 ->

Scope Resolution Operator 範疇解析運算子

- 範疇解析運算子 `::`
- 用來表示 `bmi()` 屬於 `class person` 的方法

Encapsulation 封裝

將資料成員與成員函式依功能劃分為公有與私有

包裝在一個類別內

保護私有成員

使它不會直接受到外部的存取

圖十三、Notion 筆記內容(一)

圖十三、錯誤! 找不到參照來源。圖十四是我利用ⁱⁱNotion 這個筆記軟體整理的筆記，選擇 Notion 來整理筆記的原因是它的操作便利性，Notion 的文字方塊可以任意拖曳移動，能夠快速進行排版，第二個原因是 Notion 擁有特殊的文字方塊，可以寫下多種語言的程式碼，方便直接寫下語法的範例。而這些筆記的內容主要為 C++ 的語法、易錯觀念以及ⁱⁱⁱZeroJudge 實作題目的解題思維，這些語法多數是在國立臺北科技大學的程式先修課程學到的重點，例如指標與動態記憶體配置、物件導向程式設計以及泛型設計等。其中又以指標與動態記憶體配置最為複雜，在筆記中有很大的篇幅是著重於這個部分，因為只有 C/C++ 允許使用者直接對記憶體進行操控，是這個程式語言的特色，同時也是 C/C++ 編譯速度最快的原因之一。

函式的多載

- 解決執行內容相同，但引數不同的問題

💡 引數的不同可以是**型態不同**或**個數不同**

- 有**相容匹配問題**(請避免發生)

參數預設值

- 不能使用分離式宣告**
- 有順序問題
P(0 , 4)這樣呼叫**會報錯**

💡 必須將要初始化的參數都**擺後面**

inline 內行展開函式

- 用於加速**簡短函式**的**執行速度**
- 將簡短函式**展開**至 main函式，取代**頻繁的換手過程**

```
inline int fn_Add(int int_Num1, int int_Num2)
{
    return int_Num1 + int_Num2;
}
```

💡 編譯器會**自動判斷**是否執行內行展開的動作，若不執行則視為普通函式，**不影響輸出結果**

Generic Programming 泛型設計

- 開發一堆相同的多載程式

圖十四、Notion 筆記內容(二)

ⁱ GeoGebra -- 自由的數學工具--全世界超過一億人使用 Url: <https://www.geogebra.org/?lang=zh-TW>

ⁱⁱ Notion — The all-in-one workspace for your notes, tasks, wikis, and databases. Url: <https://www.notion.so/>

ⁱⁱⁱ ZeroJudge — 高中生程式解題系統 Url: <https://zerojudge.tw/>