電腦音樂
以程式語言體現「準則作曲」的概念
Computer Music
The Realization of the concept of Algorithmic Composition by Applying Computer Programming

邱明慶
Ming-Ching CHIU
美國伊利諾大學音樂學院作曲系碩士班研究生

準則作曲的源流

「準則作曲（Algorithmic Music Composition）」是一種音樂創作手法。其基本概念是由作曲家設定一套規則，再根據這套規則來生產音樂。雖然這樣的概念看似新穎，但其實它的起始可溯源至古典時期。莫札特曾經創作一個音樂遊戲，以擲骰子的方式決定音樂的走向，此作品可謂準則作曲之濫觴。到了二十世紀，作曲家漸漸將注意力移向系統化的作曲手法，如十二音列作曲法、機遇音樂等。靠著電腦的幫助，系統化的音樂創作在二十世紀後半益發熱絡，並與另一音樂潮流 —「電子音樂」相接軌。相關的研究、創作在這段期間日漸增加，可利用之軟硬體設備也慢慢的普及化。本文將簡略敘述電腦音樂的相關基礎概念，並引入三層次的音樂創作觀點來闡述準則作曲的基本理論，再以簡單的電腦程式碼來實現此理論。

電腦音樂之基本概要

見到「電腦音樂」一詞，有些讀者會直覺的認爲這是「MIDI」的中文翻譯，有些人則會聯想到使用隨身聽播放的MP3檔案，也可能有些人在閱讀此文章之前從未聽過MIDI，也不知道MP3是什麼。這兩種檔案格式各代表了電腦音樂的兩大基本型態。而在開始創作電腦音樂之前，對它們的的初步認識是必要的。故此，本文在此對這兩種基本型態稍作說明。

MIDI（Musical Instrument Digital Interface，數位樂器介面，見圖1）是一種標準化的工業通訊協定，經由這種介面而流通的訊號爲一連串代表音高、音長、力度等資訊的數字。這種介面能讓所有懂得MIDI通訊協定的電子樂器（包括電腦）互相溝通。簡單來說，它是一種通用於電子樂器之間的語言。不論是哪家公司生產的電子音樂設備，只要它懂MIDI，即可控制其他也懂MIDI的設備，或反之由對方控制以演奏音樂或是將訊號記錄下來。

值得一提的是，MIDI介面中傳輸的資料並非一般音響音源線內直接由物理聲波轉化而成的電流波形訊號，而是一連串的數字。MIDI設備運作時，這一連串的數字告訴接受訊號的設備何時該用哪種音色發出多長的音、音量多大、音高多高。此外，主控方也可以利用MIDI介面傳送諸如滑音（Pitch bend）或是左右相位（Pan）等其他相關的資訊。因此，MIDI訊號可以說是給MIDI樂器看的樂譜（圖2）。這樣的通訊方式或多或少可以從MIDI的英文全名中看出端倪來。

圖1　MIDI訊號的傳輸埠

Trk	HMSF	MBT	Ch	Kind	Data		
1	00:00:00:00	1:01:000	1	Control	7-Volume	90	
1	00:00:00:00	1:01:003	1	RPN	Pitch Bend Sensitivity	1536	
1	00:00:13:08	8:04:939	1	Control	7-Volume	79	
1	00:00:13:08	9:01:003	1	Note	Bb5	77	408
1	00:00:13:18	9:02:028	1	Note	Bb5	73	375
1	00:00:14:04	9:03:019	1	Note	Bb5	75	375
1	00:00:14:14	9:04:011	1	Note	Bb5	73	372
1	00:00:15:00	10:01:003	1	Note	Bb5	77	408
1	00:00:15:10	10:02:028	1	Note	Bb5	73	375
1	00:00:15:20	10:03:019	1	Note	Bb5	75	375

圖2　MIDI訊號所傳送的資料

至於我們所聽到的MIDI音樂，則是由受控的MIDI設備在接收訊號之後驅動機器內的合成器（Synthesizer）製造不同波形的電流訊號，再傳至揚聲器（喇叭）振動空氣後產生音波而成。合成器產生的電流訊號即是一般音響音源線中流通之音訊（圖3）。至於合成器如何製造波形，端看樂器製造者的設計。早期的MIDI設備只能簡單的模擬樂器的聲音（圖4），而隨著科技的進步，現在的MIDI設備可以發出的聲音複雜得多，模擬眞實樂器音色的眞實度亦日趨精緻。由於人所聽到的聲音品質取決於最後發出聲音的設備，因此若不滿意家中電腦播放MIDI檔案的聲色，換一張品質較高的音效卡即能改善，而音色不佳的電子琴若另外接上較專業的音源器，亦能發出悅耳的聲音。

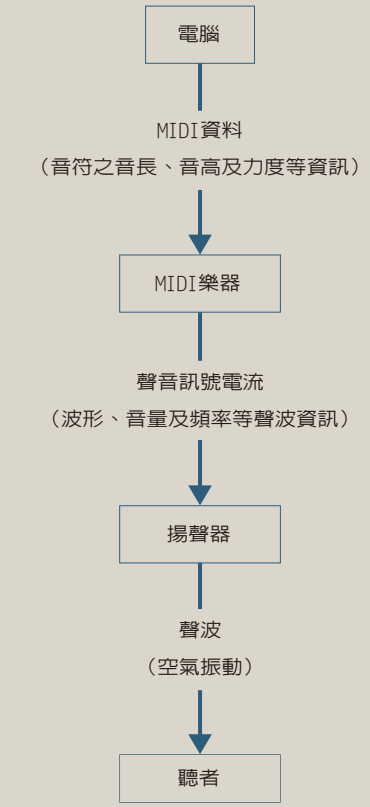


圖3　MIDI訊號在經過兩次轉換後，成為人能夠聽見的音波

圖4A　豎笛的波形

圖4B　早期電子樂器模擬的豎笛波形（方波）

圖4C　小提琴的波形

圖4D　早期電子樂器模擬的小提琴的波形（鋸齒波）

除了MIDI之外，電腦上另一種用來記錄音樂的格式是將音波直接記錄下來，稱爲數位音訊（Digital Audio），MP3就是這種類型的代表。這種格式所儲存的是經過採樣的聲音波形。播放此類檔案時，播放軟體將檔案裡的資料還原成接近原本波形的電流訊號，然後由揚聲器照著此音訊發出聲音。因此，姑且不論揚聲器之性能，理論上同一個MP3檔案在任何電腦或隨身聽上播放時聲音品質應無二致。在電腦的世界中，MIDI資料以相當於樂譜的型態存在，而數位音訊則以錄音帶的方式存在。演出MIDI訊號需要電子樂器的現場演奏，而播放數位音訊則較近似於播放錄音帶。

上文介紹之MIDI以及數位音訊兩種資料格式實際上僅是電腦音樂領域中的一小部分而已。所謂電腦音樂在廣義的定義上還包括所有由電腦直接產生的音樂和以電腦爲工具所創作出來的音樂。Roads（1995）的著作中，電腦音樂可粗分爲兩種：（１）數位音訊處理（２）音樂程式設計。前文所提之MP3屬於數位訊處理的範疇，而MIDI訊號的創作則是音樂程式設計的一環。不過這樣的界定並不是絕對的，現在的趨勢之下，一種電腦音樂可能包含了多種媒介的互動，在這樣的情形之下，刻意分類並沒有太大的意義。

以數位音訊處理為主體的電腦音樂

數位音訊可說是電腦音樂的中心，與數位音訊相關的知識技術大多承襲於二十世紀前期及中期的電子音樂，所以它有接近百年的歷史，相關的理論及作品也非常豐富。

在自然界中，聲音訊號是以「類比」（Analog）的型態在介質中傳遞（例如空氣、水等）。類比型態的資訊是一系列連續不斷的值。舉例來說，空氣振動時在某個固定點所測定的氣壓就是一連串連續而且不斷變動的值。這一連串的值可以直接寫入錄音帶中以類比的方式儲存，要播放時再由音響轉換爲音訊（電壓不斷變動之電流）最後由揚聲器再轉換成音波。

可惜的是，電腦的運算和資料儲存是以一連串的零與一組成的，它能儲存的資料沒有辦法無限制的連續。因此，以處理數位音訊爲目的的理論及技術便應運而生。這些技術包括了數位與類比訊號的互相轉換、數位音訊的儲存、壓縮及其他處理（圖5）。時下不少以數位音訊處理爲主要功能之商業軟體均備有強大功能，作曲家只要能夠善用這些功能便能創作出具有個人風格之音樂作品。

除了處理已存在的音訊之外，音樂家也會利用電腦來創造個人化的音色。利用軟體來控制合成器製造聲音對音樂家來說並不困難，

圖5A　原本空氣中之波形

圖5B　以類比方式記錄之波形

圖5C　以數位方式記錄之波形

只要對音波的物理特性有一定程度的了解，人們即可利用這類軟體來製造自然界聽不到的聲音。傳統作曲法中，作曲家利用不同的樂器組合和演奏技巧以製造出不同的音色，而現代的電腦音樂家們則是憑著他們對電腦及物理學的知識來滿足他們對音色的想像。有些具有程式設計能力的音樂家也會自行設計能夠處理數位音訊的程式以滿足他們個人對於音樂創作上的需求。

以音樂程式設計為主體的電腦音樂

有些設計程式者爲了處理數位音訊而動手設計程式，但另一群音樂家著迷的卻是程式設計本身所架構出的美感。

程式設計的基本概念是命令電腦執行一系列預先安排好的的指令以完成某項工作，這是一個尋求解答的過程。每個程式的誕生都是爲了解決一個或大或小的問題。而程式設計讓作曲家著迷的原因主要是由於程式設計和作曲的關聯性。不論在設計、過程或結果上，作曲和程式設計上都有極大的相似之處（圖6）。

首先，這兩者都與數學有某種程度的關係。雖然作曲不像程式設計一樣直接處理數字，但是音高、音階、音程、和弦、和聲、節奏、音量、速度等音樂元素，都與數字有關而且都可以數值化。其次，兩者都是依照簡單的規則，寫出合邏輯的一連串符號，再由他人解讀這些符號並產出成品。圖6中兩系統的並置，讓我們可以了解兩者之間的相似性。再細看下去，程式設計中的基本概念如迴圈（Loop）、函式呼叫（Function Call）等，在樂譜中亦不難發現（見譜例1）。電腦誕生之後，作曲家們很快就發現作曲與程式設計兩者相通之處，並開始以編寫程式碼的方式來創作新型態的音樂。但是，這樣的作曲方法和準則作曲又有什麼關係呢？

音樂創作的三個層次

美國電腦音樂作曲家Taube（2004）曾經指出，音樂創作可分爲三個不同的層次，分別爲「音響層次（Acoustic Level）」、「樂譜層次（Score Level）」以及「後設層次（Metalevel）」。音響層次的音樂創作直接產生人們可以聽到的音樂，例如爵士樂的即興演奏和以數位音訊爲主體的音樂創作。作曲家將音符寫在紙上由演奏者演出以及將音樂寫成MIDI檔案由MIDI設備演出則是屬於樂譜層次的音樂創作。在此層次中，作曲家並不直接製造音樂，而是生產某種符號，這些符號必需經過第三者進行轉化變形的過程，才會成爲一般人聽到的音樂。跟實際可聽到的音樂相

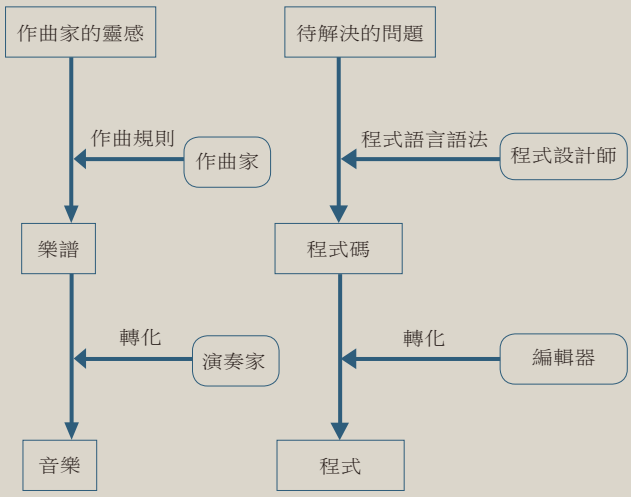
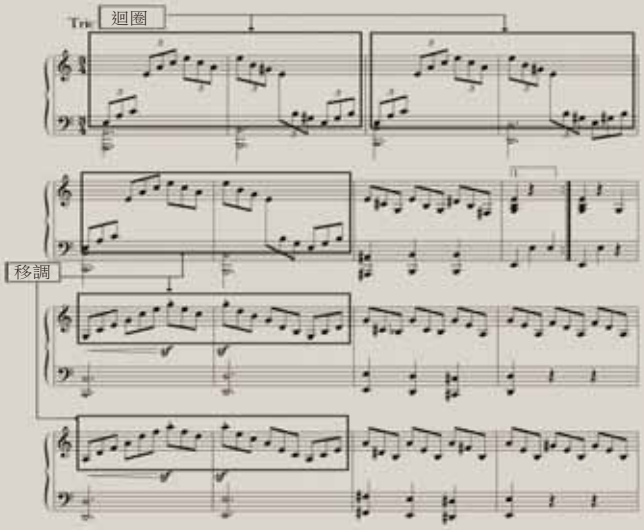


圖6　作曲與程式設計的比較



譜例1　貝多芬鋼琴奏鳴曲 Opus 2 Nr.3 Trio 片段。第一及第二小節是執行兩次的迴圈，移調則可視為一種函式的呼叫。

比，樂譜顯然抽象得多，因爲光見到樂譜並不能確切知道它演奏出來會是什麼樣子，而不同的演奏家演出相同的曲目得到的結果也不盡相同。

後設層次的音樂創作，則比樂譜層次的音樂創作來得更爲抽象。介紹後設層次的音樂創作之前，得先了解「後設」（Meta-）的意義。「後設」指的是「關於○○的○○」，舉例來說，一則與新聞有關的新聞，我們可以稱之爲一則「後設新聞」，關於認知的認知，即爲大名鼎鼎的「後設認知」，在路上看到一塊招牌上面寫了「招牌」兩個大字，則它是一塊「後設招牌」。最後一個例子雖然很奇怪，平常也不會有人這樣說，不過它確實是一個將「後設」這個概念實體化的例子。現在，我們可以大膽的說：後設層次的音樂創作指的就是「音樂創作的創作」。

後設音樂創作的成品既非音樂本身，亦非抽象的樂譜，而是更爲抽象的資訊。這些資訊必需經過轉化變形程序，才會變成我們認識的樂譜，因此比一般的音樂創作多了一道轉化手續。下例即是一個後設層次的音樂創作成品：

這首曲子有四小節，由鋼琴演奏，C大調， 每小節一個和弦，依序是C、F、G、C，左手是分解和弦伴奏，右手旋律是連續的四分音符。

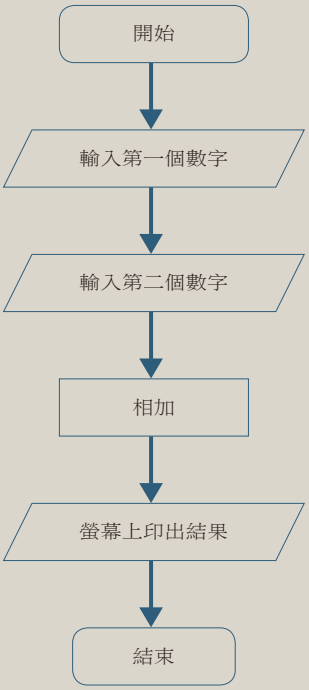
以上這段文字是一個貨眞價實的音樂創作，不過在這樣的狀態下我們沒有辦法欣賞它，必須要有人照著這些資訊將音符具體化（第一次轉化），再依照具體化的音符演奏出來（第二次轉化），我們才有辦法聽到。有趣的是，不同的人在轉化同一個後設音樂作品時並不會生產出相同的樂譜，而且作品裡面的資訊越少，具體化後的差異就越大，反之亦然。舉例來說，若在這個作品中加入「3/4拍，速度72」，那麼具體化時的限制就變大了。這樣的概念是不是很有趣呢？

程式碼運作之基本原理

後設作曲的成品不一定是一段文字，它可以是任何一種能被解讀轉化的資訊，其中包括了這次的主題 — 電腦程式的程式碼。在簡單的程式語言中，電腦按照程式碼內每一個命令依序從頭執行到尾。舉例來說，如果想要讓電腦作一個相加的運算，程式碼會是這樣寫的：

程式碼例一：

1. 請電腦使用者輸入第一個數字
2. 請使用者輸入第二個數字
3. 把第一個數字加上第二個數字
4. 把結果印在螢幕上



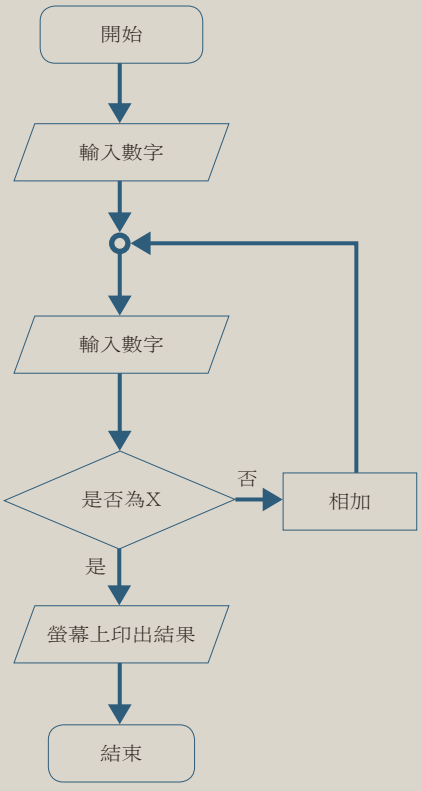
流程圖1

這短短的四行命令讓電腦能夠算出兩個數字的和。一組像這樣的命令組合，稱爲一個「函式（Function）」，每一個函式都可解決一個或大或小的問題。由於不同的程式語言語法均不相同，因此本文中只使用概念性的流程圖（Flowchart）以及易讀的虛擬碼（Pseudo code）。筆者實際上使用的程式語言是LISP Common Music之下的SAL語言。

如果電腦對於命令的處理只能照順序依次執行，有些問題會變得難以解決，此時電腦語言中另外兩個控制流程的功能就顯得非常重要了，它們分別是「條件執行（Condition）」和「迴圈（Loop）」。條件執行即是「如果…就…」，而迴圈即爲重覆執行。以下是此二功能之簡單範例：

程式碼例二：

1. 請使用者輸入第一個數字
2. 請使用者輸入另一個數字
3. 檢查此數字是否為X
   如果不是
      把兩數相加，再回到步驟2
   如果是
      把結果印在螢幕上



流程圖2

此函式的功能爲計算多個數字的和，使用者可以連續輸入數字，最後按下X得到結果。其中回到前面的步驟再做一次的作法，就是「迴圈」的概念，而判斷輸入的

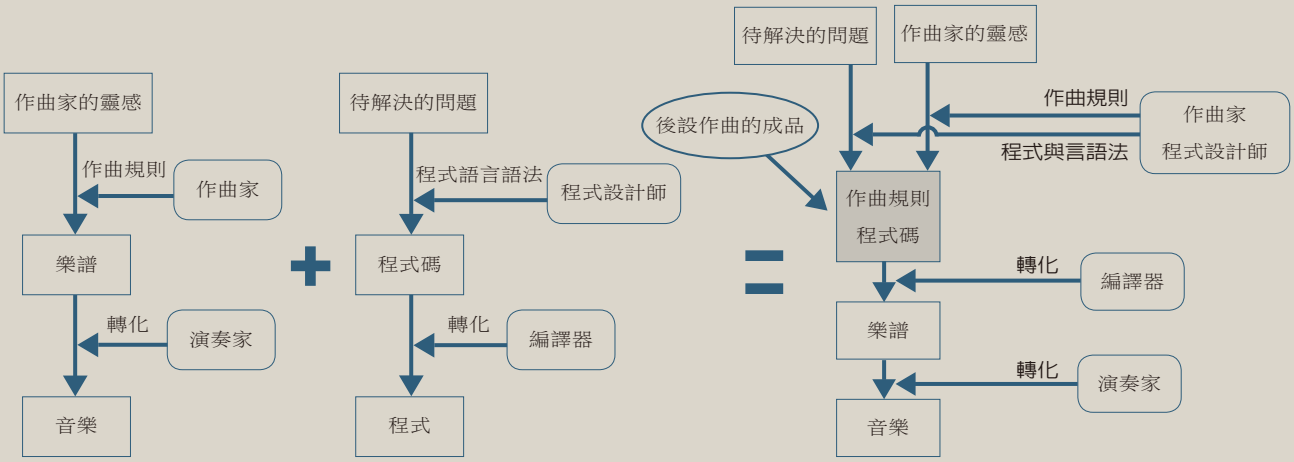


圖7　傳統作曲、程式設計和以程式設計為媒介的準則作曲。

數字是否爲X並進而依照指示做出不同的動作則屬於「條件判斷」。有了這樣的概念，即可開始用編寫程式碼的方式嘗試簡單的準則作曲。

以撰寫程式碼的方式體現「後設作曲」的概念

雖然現在有不少非常前衛的作曲法幾乎奇怪得令人無法想像，但基本上絕大多數的樂曲還是從作曲家的靈感開始發展。作曲家安排好段落、設計出樂曲的動機或主題、大致上決定好高潮、轉折等事件之時間點等等，再將音符依照這些想法記錄在五線譜上。音樂由概念具體化爲白紙黑字的過程中，作曲家運用不同的作曲技巧來決定音樂作品的最後的模樣。後設作曲的任務即是創作出能夠控制這個過程的資訊。如圖7所示，運用程式設計所創作之後設作曲即是在於設計出樂曲之準則並將此準則轉化爲能夠讓電腦執行之程式碼。

在這樣的創作過程中，創作者身兼作曲家及程式設計師的角色，利用其在作曲上以及程式語言上的知識，將靈感以及創作時需要解決的問題轉化成程式碼。電腦將程式碼轉化成樂譜層次的作品之後，再交給演奏者轉化成眞正的音樂。圖7中所標示之演奏家可以是人，也可以電腦或其他設備。

圖7中所標示之樂譜並不一定是指五線譜。作曲家在設定作品最後的演出方式時就決定了樂譜的型態，若作品最終將交給人演出，五線譜是不錯的選擇；若作品將直接交給電子樂器演出，則樂譜可以是MIDI檔案或其他電子樂器可以了解的型態。交由電腦演出的音樂作品中可以使用眞實樂器無法演奏之技巧，例如每拍一百連音或是複雜的微分音等等。這樣的特點讓作曲家可以極度延伸音樂語言的可能性而不再受制於眞人演奏者的技巧障礙。

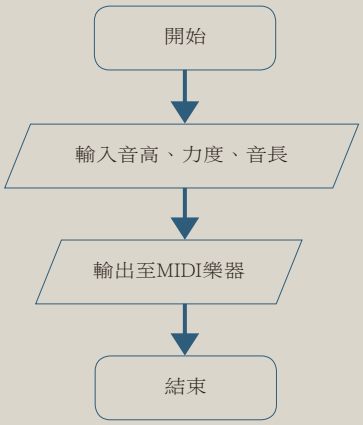
和傳統的作曲方式比起來，準則作曲多了一道轉化的手續，因此在轉化成實際的音樂時，其多樣性會更高。這是什麼意思呢？先來看看以下之簡單範例。

下面這一個小小函式的目的，在於能夠讓MIDI樂器發出一個指定的音。要演奏一個音，最起碼必須知道其音高、力度（大小聲）以及音長。在標準的MIDI介面中，力度由最小到最大共分爲一百二十八等分，音長可以「拍」作單位，音高也是以稱爲「音高號碼（Key Number）」的數字來代表。中央C的音高號碼是60，每高一個半音時，音高號碼會就加一，低一個半音就減一，因此比中央C高一個八度的C音高號碼爲72，低半音之B則爲59。

接下來，我們即可以簡單之程式碼讓電腦創作出演奏一個音的樂譜。

程式碼例三：發出〇音

1. 請使用者輸入音高
2. 請使用者輸入力度
3. 請使用者輸入音長
4. 把收集到的音高、力度、音長資料送至MIDI樂器奏出

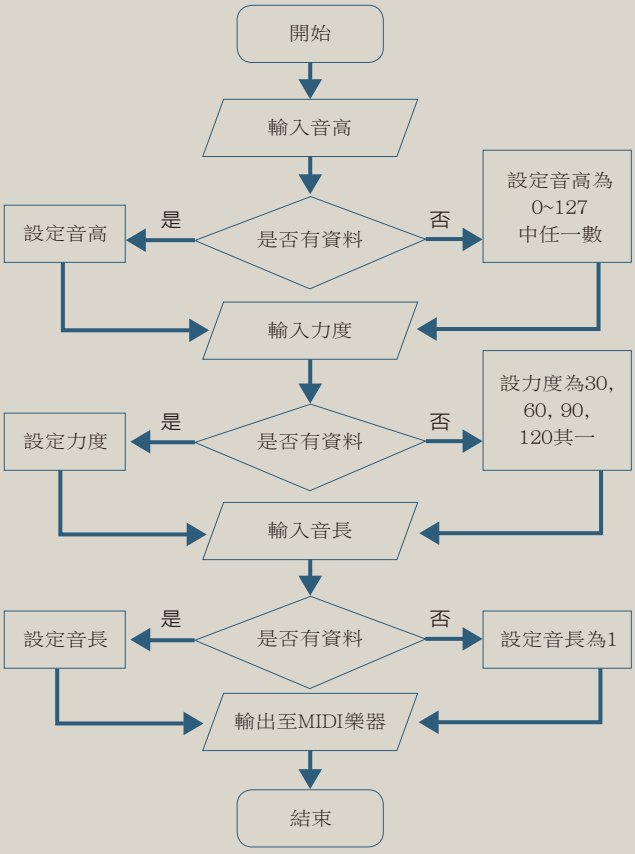


流程圖3

將以上的程式碼經過電腦的編譯再送至MIDI樂器演奏，即可聽到指定音高、音長以及力度的音了。不過，如果編輯程式碼只能寫出這樣的音樂，又何必如此麻煩呢？直接寫出樂譜或使用編輯MIDI訊號用的軟體豈非更爲方便快速？下面的程式碼提供了一個很好的例子，以展示這種創作方式的有趣之處。

程式碼例四：

1. 請使用者輸入音高
2. 如果沒輸入音高，就從0到127裡選一個整數當音高
3. 請使用者輸入力度
4. 如果沒輸入力度，就從30, 60, 90, 120這四個數裡選一個當力度
5. 請使用者輸入音長
   如果沒輸入音長，就把音長設定為1
6. 把收集到的音高、力度、音長資料送至MIDI樂器奏出



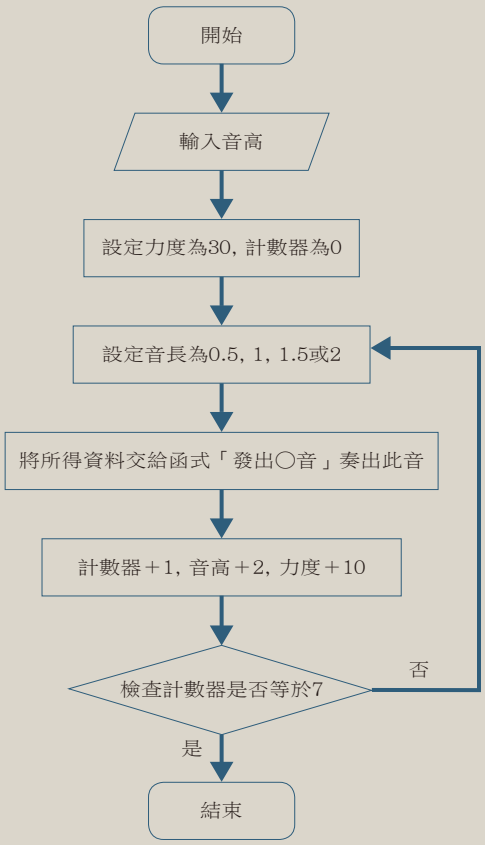
流程圖4

我們可以發現，由於加入了「條件執行」的元素，若在執行程式時不指定某個發音必要之條件，則電腦將會依照規則來找出適當的做法。像這樣只訂定規則而不直接寫下樂譜的方法就是準則作曲法的基本概念。執行這個程式時，隨著作曲者提供不同的輸入資料，所得到的結果也有極大的變化，而這些不同的結果都是同一個規則的體現。這個有趣的特性吸引了很多作曲家。同時，正如此例，這樣的作品也有常含有「機遇音樂（Aleatory）」的成分。在傳統的機遇音樂中，作曲家將作品中一部分必要的元素交給演出者自行安排，而在以電腦語言爲創作方法之的準則作曲中，隨機的部分則在第一次轉化時交給電腦來決定。當然，若最後輸出的樂譜將由眞人來演出，作曲家亦可在這一次的轉化上安排第二次的隨機因子。

下一個例子中，程式碼構成一個迴圈，並多次使用前文建立之函式「發出○音」，以奏出一個上行漸強之全音階。

程式碼例五：全音階上行

1. 輸入音高
2. 設定力度為30
3. 設定計數器為0
4. 設定音長為0.5, 1, 1.5, 2裡面其中任一數
5. 以所得資料輸出一個MIDI音符
6. 計數器+1，音高+2，力度+10
7. 檢查計數器，若計數器等於7，則停止，否則回到步驟4



流程圖5

在這個函式裡面，我們不再從頭告訴電腦如何發出一個音，而是直接執行另一個已經設定好的函式「發出○音」，這樣的概念稱爲「函式呼叫

（Function Call）」。利用函式呼叫的方式，我們可以建構出函式的階級表，階級較高的函式通常掌握作品中大範圍的結構諸如段落的長度、速度等資訊，它呼叫較低階的函式以控制作品中的細部資訊，諸如每個音的音長、音高、音量等，最後組成一個完整的作品。較高級的函式可以呼叫下方任何一級的函式。在大型的作品之中，這樣的層級可以有很多層，端看作曲者的需要而定（圖8）。

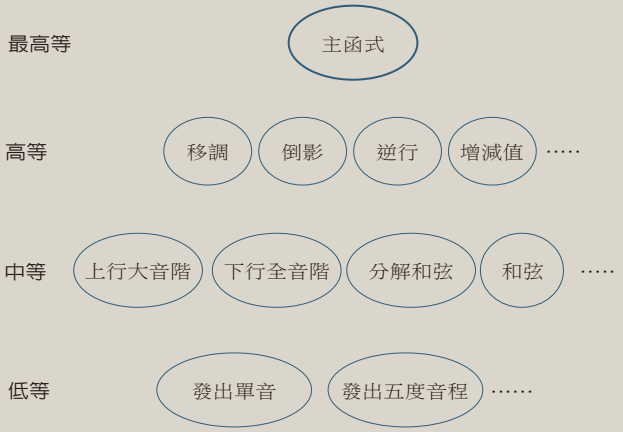


圖8　函式的階級關係一例

應用程式設計之電腦音樂其可能性

本文中以簡單的範例介紹以程式設計爲方法之準則作曲的初步概念，實際上，除了文中簡單的函式之外，作曲家亦使用數學、物理或其他領域之複雜理論例如渾沌理論、機率的運用、馬可夫分析（Markov Analysis）、基因演化、碎型或是由作曲家自創之演算法。

本文中程式碼經編譯後主要的輸出格式是MIDI，但是這並非意謂此作曲法只能輸出MIDI音樂而已。作曲家可以用更多元的輸出方式來設計最後的音樂成品，例如直接控制聲波合成器以操控音頻、音量等元素，或輸出樂譜交由演奏者演出，亦可一半輸出成樂譜交給演奏者演出，搭配另一半預先由電腦合成的聲音，或是輸出樂譜後再由作曲家加以修改等等，端看作曲者的巧思。

也許讀者已經約略對電腦音樂、後設層次的準則作曲等概念有了初步的認識。此種作曲方法讓作曲家可以將注意力從樂譜上的細節轉移到較高層次的結構上，因此對於創作者的技術要求，也與傳統作曲不太盡相同。某些在傳統作曲中注重的技巧在這種創作法中並不吃重，而某些傳統作曲不強調甚至不存在的技巧，例如對於數學與程式設計的相關知識，則顯得舉足輕重。在這樣的情況下，作曲家與其音樂作品的關係亦跟傳統音樂略有不同。其中最大的差異就是作曲不再是專業作曲家的專利，對電腦及數學略有涉獵的愛好音樂的人士也可以利用電腦來創作音樂。雖然有人質疑這種規則式、系統式的作曲方法沒有人性、不能稱爲藝術，而這樣的爭辯從荀白克的十二音列作曲法誕生之始就存在而且勢必將再持續下去，但是音樂家們仍然肩負著「有創新才有藝術」的信念，繼續將自己的意念透過音樂傳達給世上所有的聽眾。畢竟，音樂是人與人之間最純粹也最美妙的一種溝通方式！

■ 參考文獻

Roads, Curtis. (1995). *The computer music tutorial*. Cambridge, Mass: MIT Press.

Moore, F. R. (1990). *Elements of computer music*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc..

Chadabe, Joel. (1997). *Electric Sound: The Past And Promise of Electronic Music*. New Jersey: Prentice-Hall.

Cope, David. (2000). *The algorithmic composer*. Madison, Wis: A-R Editions.

Taube, Heinrich. (2004). *Notes from the metalevel: Introduction to algorithmic music composition*. New york: Taylor&Francis Group.

Taube, Heinrich. *Algorithmic design of music*. Internet on-line. Available from http://camilx2.music.uiuc.edu:16080/classes/404A/

Burns, Kristine H. (1994). "The history and development of algorithms in music composition, 1957-1993." Ph.D. diss., Ball State University.

Computer Music
The Realization of the concept of
Algorithmic Composition by Applying
Computer Programming