科學月刊【數・生活與學習】專欄・百年 8 月

# 算術的潛規則

單維彰・100 年 7 月 20 日

大約在今年四月，網路（特別是臉書）上流傳一道算術題目，熱鬧得上了平面媒體；其實，電視新聞和報紙都越來越直接引用不具名的網路開放資訊，所以臉書的流行被刊上了報紙，也不算什麼大事了。題目是：

**6÷2(1+2)**

我或許記錯了數字，但是無所謂，數字不是重點。

這一道人人會做的算術題目之所以引起大家的興趣，是因爲很多人做錯了。新聞最喜歡聳動的標題，我記得當時看到的標題是『連數學老師也不會做。』我想，數學老師或許會『算錯了』，但是說『不會做』就有點兒過份。就算不小心中了圈套，大概也只會聳肩微笑，體會自己在計算時受到習慣制約的影響而產生娛樂效果，卻不至於捶胸頓足。

受到代數算式的習慣制約，以及「先算刮號」的強烈暗示，很多人可能會落入圈套而先做÷號右側的乘法計算，也就是 $2(1+2)=2\times3=6$ ，然後做 $6\div6=1$ ，於是得到（錯誤的）答案 1。

何以見得它是錯的？我們若將整數的除改成分數的乘，可能看得更清楚：

$$6\div2(1+2)=6\times\frac{1}{2}(1+2)$$

當算式中全是相乘，則執行的次序不影響答案（這個性質稱爲乘法的結合律），這時候很肯定答案是 9。這是正確答案。

但是，學習並不止於正確答案。這一道其實更像是「心理測驗」的算術題，還可以觸發更多的學習。

首先，就一個算術的「潛規則」而言，這道題目的命題形式是錯的。小學教師都知道， $2(1+2)$ 是錯誤寫法，若小學生在作業簿裡寫出這樣的算術式，應該會被指正爲 $2\times(1+2)$。雖然數學課程裡似乎不曾明確地交代：只有在代數式中才能省略乘號，在算術式中不能省略乘號。這確實是書寫數學式的一條「潛規則」。例如， $x\div yz$ 或者 $6\div yz$ 或者 $6\div2z$ 都可以省略乘以 $z$ 的乘號，因爲相乘的兩數之中，至少有一個是以文字符號代表的數。所以，這一道題目的正確書寫形式，應該是 $6\div2\times(1+2)$ 。

一旦寫成了正確形式「 $6\div2\times(1+2)$ 」，會不會降低人們想要先算乘法部分的衝動呢？我認爲這是一個非常有趣的心理學問題，而我並不知道答案。

大家都知道，刮號裡的式子要先算，所以這一題應該化簡成「 $6\div2\times3$ 」。相信所有人都能達到這一步。所以，除了前述的「潛規則」以外，這道算術題目主要須探討的是：應該先算除、還是先算乘？顯然兩者的結果不一樣，如果先算乘，

則結果是$6\div(2\times3)=6\div6=1$，如果先算除，則結果是$(6\div2)\times3=3\times3=9$。由此可見一般中學生應該知道的事實：乘法和除法運算不滿足結合律。

算術有一條大家耳熟能詳的規則，稱『先乘除，後加減』。例如$2+3\times4=14$、$2\times3+4=10$，這些規則的運用應該是一般人都熟練的。而算術的另一條規則，就比較少被提起，它是說『連續的加減或連續的乘除，應從左至右依序執行』。例如$4-3+1=(4-3)+1=2$而不是$4-(3+1)=0$；依照這一條規則，$6\div2\times3=(6\div2)\times3=9$。

規定『從左到右』是必要的，而不是方便的。如此規定，才能與「更抽象」的情況相容。有了「負數」之後，我們不再須要減法，減去一數可以換成加上它的相反數，例如$4-3+1=4+(-3)+1$。有了「有理數」之後，我們不再須要除法，除以一數可以換成乘以它的倒數，例如$6\div2\times3=6\times\frac{1}{2}\times3$。做了這些形式上的更換之後，不再有連續加減或者連續乘除的算式，一律只有連續的加或者連續的乘，而這兩種運算都滿足結合律，也就是先算哪一個都不要緊。例如$4+(-3)+1$不論先算第一個加號還是先算第二個，或者$6\times\frac{1}{2}\times3$不論先算第一個乘號還是先算第二個，結果都一樣。

銜接 19 和 20 世紀的數學大師希爾伯特 (Hilbert, 1862—1943) 有一句名言：『如果你能對街上遇到的第一個人說明一個數學觀念，才是真正懂了它。』而幾乎被尊爲「神」的計算機科學大師高德納 (Knuth, 1938 年生) 有一個更妙的詮釋：『如果你能教會電腦如何計算一個數學觀念，才是真正懂了它。』以上介紹的那些算術的規則，是如何在電腦程式語言中實現的呢？電腦程式語言，如何處理『先算刮號』、『先乘除後加減』、『從左到右』這些規則呢？以下，我們談一談這個問題。

且慢，首先讓我們回顧前述的潛規則。電腦是死死板板的東西，它通常是沒有潛規則的，除非程式設計者將潛規則寫成了正式的規則，電腦才會執行。絕大多數的電腦語言不會自動補上乘號，所以它不能接受像「6/2(1+2)」這種算式，使用者必須明確地輸入「6/2*(1+2)」，它才會算。作者僅知的一個例外，是 Mathematica 數學代數系統；它可以接受數值或變數與左刮號之間省略乘號，而自動替使用者補上乘號。

電腦語言會先保留一些符號爲運算符號，例如 +、-、*、/ 經常被用來表示加、減、乘、除運算。當然一對小刮號也是運算符號。每一個運算符號，會被賦予兩個屬性， 分別稱爲優先序和方向性。優先序是 0 或正整數，方向是左或右。在一條式子裡面，電腦會先處理優先序比較大的運算，遇到兩個或更多同序的運算連續出現時，按照方向屬性決定是從左算到右、還是從右算到左。

以 C 語言爲例，它規定左、右小刮號的優先序是 14，乘、除的優先序是 12，加、減的優先序是 11。而它們的方向性都是左。C 語言一共定義了 45 種運算符號，有些符號看來相同但因前後文而意義不同，例如「3+ -1」是$3+(-1)$的意思，

式中的減其實是「負號」；有些運算由兩個符號組成，例如「>=」是≥的意思。相較而言，刮號、加、減、乘、除運算的優先序都是頗高的。

事實上，小刮號的優先序 14 是所有運算中最高的。所以，在算式中，一定會先處理一對小刮號裡面的計算；如果有兩層以上的小刮號，按照同樣規則，自然會發生內層先處理的效果。而因爲乘、除的優先序高於加、減，所以電腦就知道『先乘除後加減』。

當連續的加、減寫成一式，或者連續的乘、除寫成一式，因爲它們的優先序相同，就看方向性。而它們的方向性都是「左」，所以就由左向右計算。

讀者或許有興趣知道，哪些運算的方向性是「右」呢？比如說等號「=」的方向性就是右。例如，當我們寫「x=y=1」，電腦先執行右邊的「y=1」，所以變數 *y* 的值是 1 了，然後才執行左邊的「x=y」，於是變數 *x* 的值才會跟 *y* 的一樣，它們都是 1。

把人們習以爲常的潛規則和慣例寫進電腦程式裡，叫它表現得像我們感覺它本來就該會做的那樣，是一件很繁瑣的工作；但是我個人總是覺得很有趣。而電腦所知的只有規則，它嚴格按照規則和邏輯執行，所以它不會犯錯，但是也沒有想像力。相對地，我們會被習慣制約，也容易受到暗示而落入圈套，所以可能被上述的題目欺騙而犯錯。可是，如果犯這種小錯是具備「想像力」的副作用，那麼我認爲，還是讓我們犯些小錯吧！