間思考此表，根據 countWords() 函式驗證各變數的值。完成此動作後，您應該會對使用此函式，來計算字串中的字數之演算法，感到很愉快的。

表 9.1 countWords() 函式的執行

i	string[i]	wordCount	lookingForWord
		0	true
0	'W'	1	false
1	'e'	1	false
2	'l'	1	false
3	'l'	1	false
4	','	1	true
5	' '	1	true
6	'h'	2	false
7	'e'	2	false
8	'r'	2	false
9	'e'	2	false
10	' '	2	true
11	'g'	3	false
12	'o'	3	false
13	'e'	3	false
14	's'	3	false
15	'.'	3	true
16	'\0'	3	true

空字串

現在來探究使用 countWords() 函式的更實際的例子。此次使用的 readLine() 函式允許使用者輸入多行文字。之後，程式計算本文中的單字總數並顯示結果。

為了使程式更靈活，請不要限制或指定輸入本文的行數。因此，必須要有方法讓使用者在輸入完畢時 "通知" 程式。一種方法是讓使用者在輸入最後一行文字後再按一次 Enter（或 Return）鍵。當呼叫 readLine() 函式用以讀取這樣的行時，函式立即遇到換行字元，因此，將空字元儲存為緩衝區中的第一個（且唯一的）字元。程式可以檢查這種特殊情況，並在讀取不包含字元的行之後，獲知已經輸入了最後一行文字。

在 C 語言中，除了空字元之外，不包含字元的字串也有其特殊名稱；它被稱為空字串（null string）。空字串的使用仍然完全符合本章定義的所有函式。stringLength() 函式回傳 0 作為空字串的大小；concat() 函式也會將 "空白" 連接到另一個字串的末端；甚至 equalStrings() 函式也是，當任一或兩個字串為 null 的時候（在後一種情況下，函式會指示它們是相同）。

請記住，事實上空字串有一個字元，雖然是空字元。

有時需要將字串的值設置為空字串。在 C 中，空字串由相鄰的一對雙引號表示。所以，以下敘述：

```
char  buffer[100] = "";
```

定義一個名為 buffer 的字元陣列，並將其值設置為空字串。請注意，字串 "" 與字串 " " 不同，因為第二個字串包含一個空白字元。（如果您懷疑，發送這兩個字串到 equalStrings() 函式，看看它回傳什麼結果。）

範例程式 9.8 使用前面提及的 readLine()、alphabetic() 和 countWords() 函式。

範例程式 9.8 計算本文的字數

```
#include <stdio.h>
#include <stdbool.h>

bool alphabetic (const char  c)
{
    if  ( (c >= 'a'  &&  c <= 'z') || (c >= 'A'  &&  c <= 'Z') )
       return true;
    else
       return false;
}

void  readLine (char  buffer[])
{
    char  character;
    int   i = 0;

    do
    {
        character = getchar ();
        buffer[i] = character;
        ++i;
    }
    while ( character != '\n' );

    buffer[i - 1] = '\0';
}
```

```
int  countWords (const char  string[])
{
    int   i, wordCount = 0;
    bool  lookingForWord = true, alphabetic (const char  c);

    for ( i = 0;  string[i] != '\0';  ++i )
        if ( alphabetic(string[i]) )
        {
            if ( lookingForWord )
            {
                ++wordCount;
                lookingForWord = false;
            }
        }
        else
            lookingForWord = true;

    return wordCount;
}

int main (void)
{
    char  text[81];
    int   totalWords = 0;
    int   countWords (const char  string[]);
    void  readLine (char  buffer[]);
    bool  endOfText = false;

    printf ("Type in your text.\n");
    printf ("When you are done, press 'RETURN'.\n\n");

    while ( ! endOfText )
    {
        readLine (text);

        if ( text[0] == '\0' )
            endOfText = true;
        else
            totalWords += countWords (text);
    }

    printf ("\nThere are %i words in the above text.\n",  totalWords);

    return 0;
}
```

範例程式 9.8 輸出結果

```
Type in your text.
When you are done, press 'RETURN'.

Wendy glanced up at the ceiling where the mound of lasagna loomed
like a mottled mountain range. Within seconds, she was crowned with
ricotta ringlets and a tomato sauce tiara. Bits of beef formed meaty
moles on her forehead. After the second thud, her culinary coronation
was complete.
Return
There are 48 words in the above text.
```

標記為 *Return* 的那一行表示按下 Enter 或 Return 鍵。

endOfText 變數用作一個旗幟，指示何時到達輸入本文的結尾。只要該旗幟為 false，則執行 while 迴圈。在這個迴圈中，程式呼叫 readLine() 函式來讀取一行文字。if 敘述測試所輸入的行（其被儲存在 text 陣列中）以查看是否按了 Enter（或 Return）鍵。如果是，則緩衝區包含空字串，在這種情況下，endOfText 旗幟被設置為 true，表示已輸入所有文字。

如果緩衝區包含一些文字，則呼叫 countWords() 函式來計算 text 陣列中的字數。此函式回傳的值將被累加到 totalWords，其包含到目前為止輸入的所有文字的累積字數。

退出 while 迴圈後，程式將顯示 totalWords 值以及一些訊息。

看起來前面的程式並沒有幫助減少您的工作，因為您仍必須手動輸入所有的文字。我們將在第 15 章 "C 語言的輸入和輸出" 中看到，這個相同的程式也可以用於計算儲存在硬碟上的檔案中所包含的字數。例如，作者使用電腦來準備手稿可能會發現這個程式非常有價值，因為它可以用來快速計算手稿中包含的字數（假設檔案儲存為正常的文字檔，而不是一些類似像 Microsoft Word 的文字處理器格式）。