國立屏東大學 資訊工程學系 C++程式設計入門教材

附錄1. 如何編譯範例程式

本書的範例程式除特別說明以外，全部都可以在Linux、MacOS及Microsoft Windows等作業系統上編譯與執行，本附錄將分別針對這些作業系統上如何編譯與執行範例程式進行說明，其中在Microsoft Windows的部份，我們會示範如何使用Dev-C++進行範例程式的編譯與執行；至於在Linux與MacOS的部份，則是以Console模式(或稱為命令行介面、Command Line Interface)的終端機(Terminal)做為示範。如果要使用其它的開發工具或軟體進行程式開發，可參考本書附錄其它開發環境與工具的相關介紹。

要提醒讀者注意的是，本附錄僅就範例程式的編譯與執行進行說明，至於如何在作業系統準備好C++語言的開發環境，則請參考本書開發您的第一個<nowiki>C++</nowiki>程式節的說明，其中針對Windows作業系統裡的Dev-C++的下載與安裝，請參考在Windows上開發<nowiki>C++</nowiki>程式小節的說明、Linux作業系統方面請參考在Linux上開發<nowiki>C++</nowiki>程式、MacOS系統請參考在MacOS上開發<nowiki>C++</nowiki>程式。

為便利說明起見，本附錄的示範都將以第2章的Example 1、檔名為hello.cpp、程式為例。假設讀者將本書的範例程式下載並解壓縮於「C:\Examples」裡，如此一來你將可以在「C:\Examples\ch2」裡找到「hello.cpp」這個檔案：

```
C++</nowiki>程式 hello.cpp >
/* Hello, <nowiki>C++</nowiki>! */
// This is my first <nowiki>C++</nowiki> program

#include <iostream>
using namespace std;

int main()
{
    cout << "Hello, <nowiki>C++</nowiki>!" << endl;
    return 0;
}
```

附錄1.1 Windows作業系統

附錄1.1.1 使用Dev-<nowiki>C++</nowiki>在Windows上編譯及執行範例程式

本書所有的程式範例(除特別說明以外)，全部都可以在Windows作業系統裡使用Dev-C++進行編譯與執行，建議讀者可以將本

書範例程式下載並解壓縮到C:\Examples(或其它你偏好的目錄裡)，以便利未來的使用。

Fig. 1: Dev-<nowiki

- 步驟1：請以滑數雙擊在桌面的Dev-C++圖示(如figure 1)，或從選單裡找尋Dev-C++並加以啟動。
- 步驟2：在選單中選擇「檔案(File)>開啟舊檔(Open)在檔案對話窗裡選取你要編譯的範例程式。
- 步驟3：完成上述步驟後，你應該可以看到如figure 2的畫面，，請在選單中選取「執行>編譯」，或是使用F9快速鍵進行編譯，如figure 3所示。如此一來就完成了範例程式的編譯，其編譯結果會出現在視窗下方的「編譯記錄」區域，如figure 4所示。

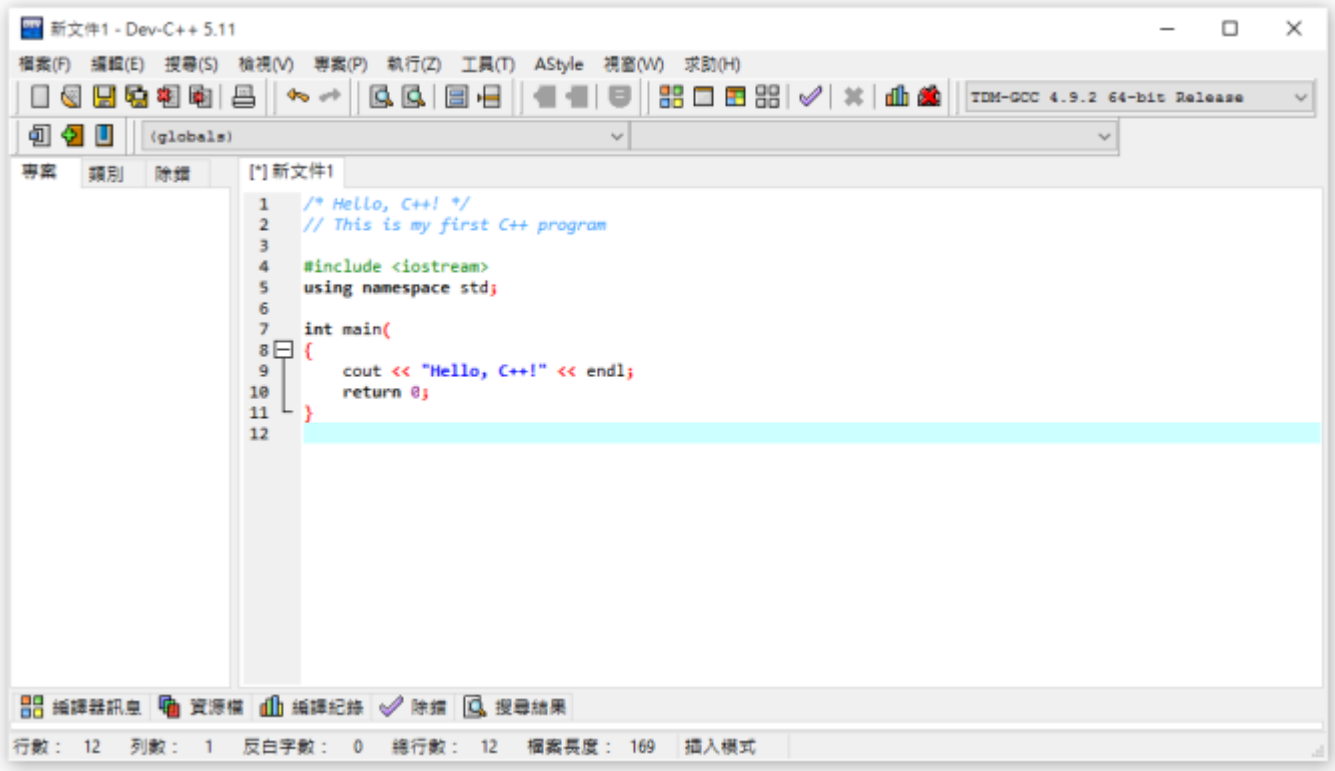


Fig. 2: 在Dev-<nowiki

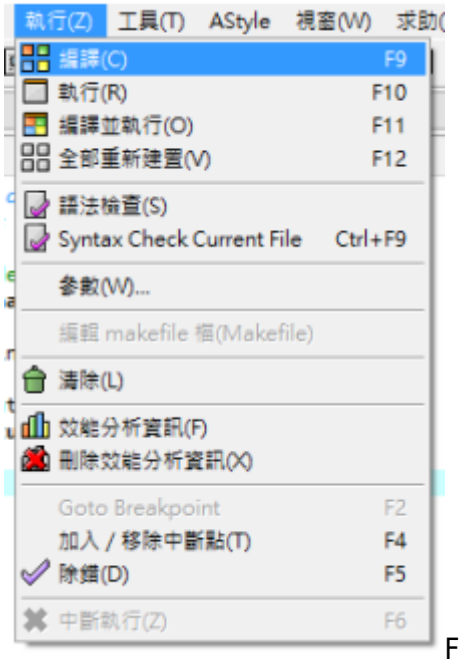


Fig. 3: 從選單中執行編譯。

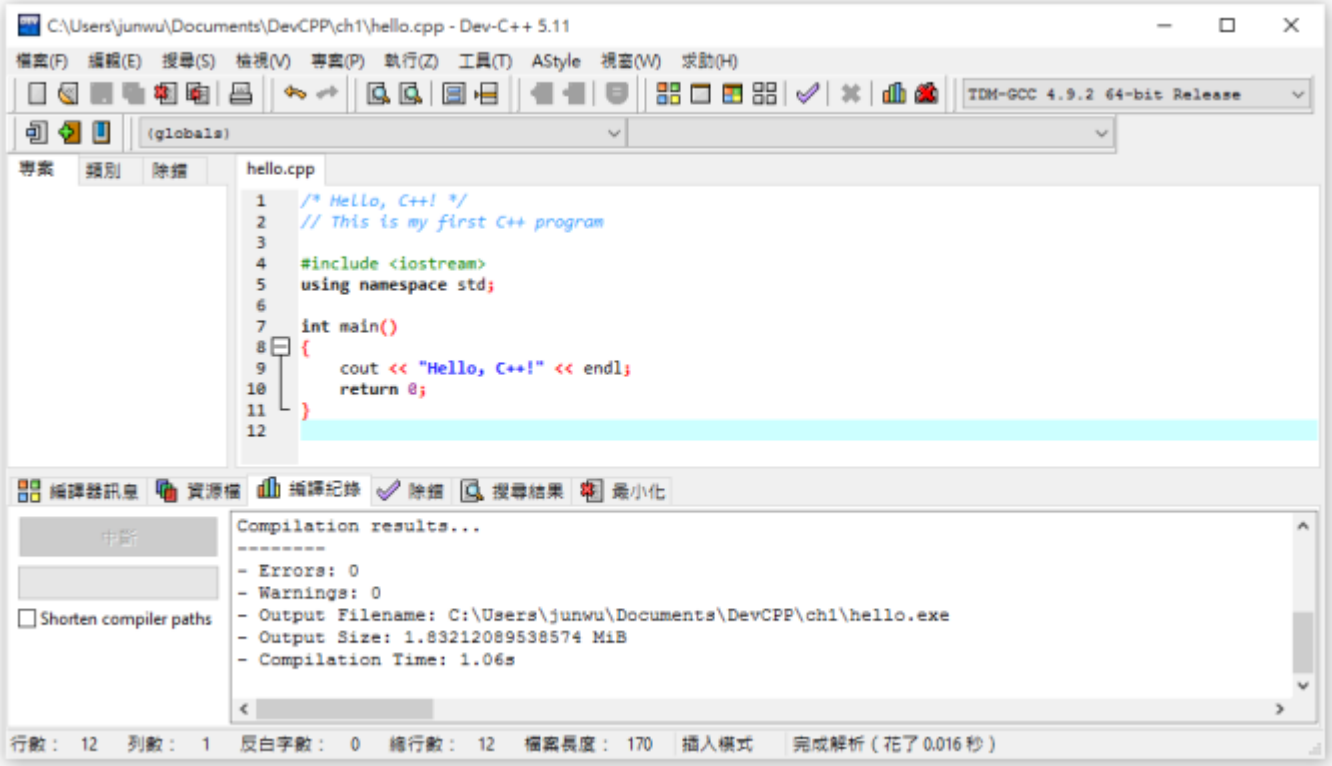


Fig. 4: 編譯結果顯示於程式碼編輯區下方的「編譯記錄」區。

- 步驟4：一旦確認程式碼的編譯沒有任何錯誤後，就會得到一個檔名與範例程式一致，但副檔名為□.exe□的可執行檔，例如□hello.exe□□請在選單中選取「執行>執行」或是使用□F10□快速鍵，就可以執行這個編譯完成的hello.exe可執行檔，如figure 5所示。一旦我們對Dev-C++下達了執行的命令後□Dev-C++會啟動一個「命令提示字元(Command Prompt)□的應用程式（也就是Microsoft Windows系統的Console環境），並在此程式中執行我們所編譯完成的可執行檔，其執行畫面如figure 6所示。以hello.exe為例，你可以看到此程式輸出了□Hello, C++!□字串。執行完成後，只要在該視窗裡按下任意一個鍵盤按鍵就可以結束該視窗。

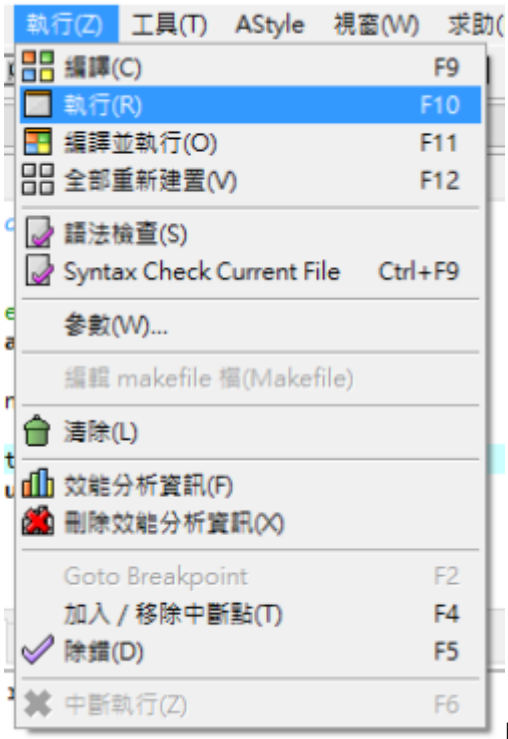


Fig. 5: 從選單選取「執行

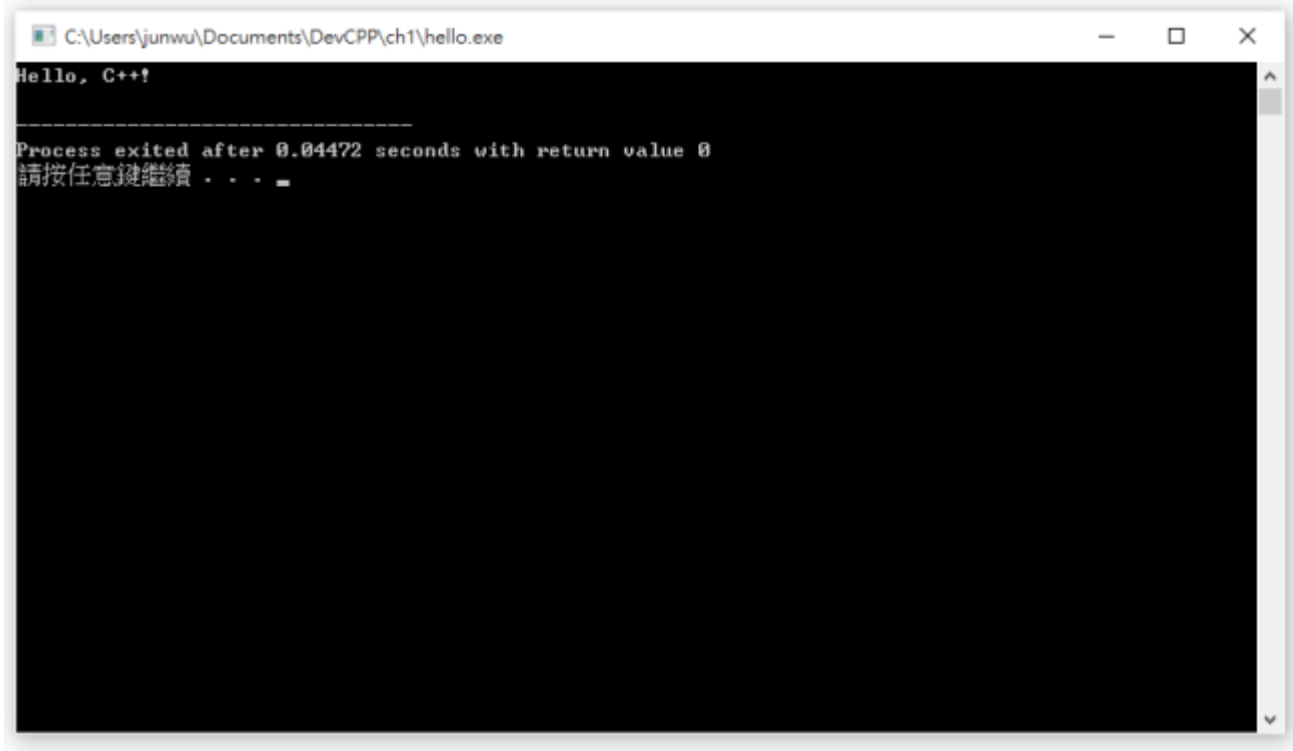


Fig. 6: Dev-<nowiki

要特別注意的是，儘管使用Dev-C++執行範例程式的結果會顯示在「命令提示字元」視窗裡，但為了便利起見，本書在說明範例程式的執行結果時，筆者將僅擷取在「命令提示字元」視窗裡的「文字執行結果」來代替整個視窗的截圖。例如figure 6的執行結果，將會改以下面的文字內容代替：

```
Hello, <nowiki>C++</nowiki>!
```

你應該會發現這種顯示的方式與原本figure 6所呈現的內容完全相同，但是卻更為簡潔，能夠讓我們更專注於範例程式的執行結果。當然，這種只擷取「文字執行結果」的方式與截圖還是有一點小差異 – 在視窗中的最後一行的「按任意鍵結束...(Press any key to continue...)」這句話不見了！可是這完全不會造成任何問題，那句話本來就不屬於程式的執行結果，它只是Dev-C++用來提示我們如何關閉命令提示字元視窗的訊息而已，和範例程式的執行結果完全無關。

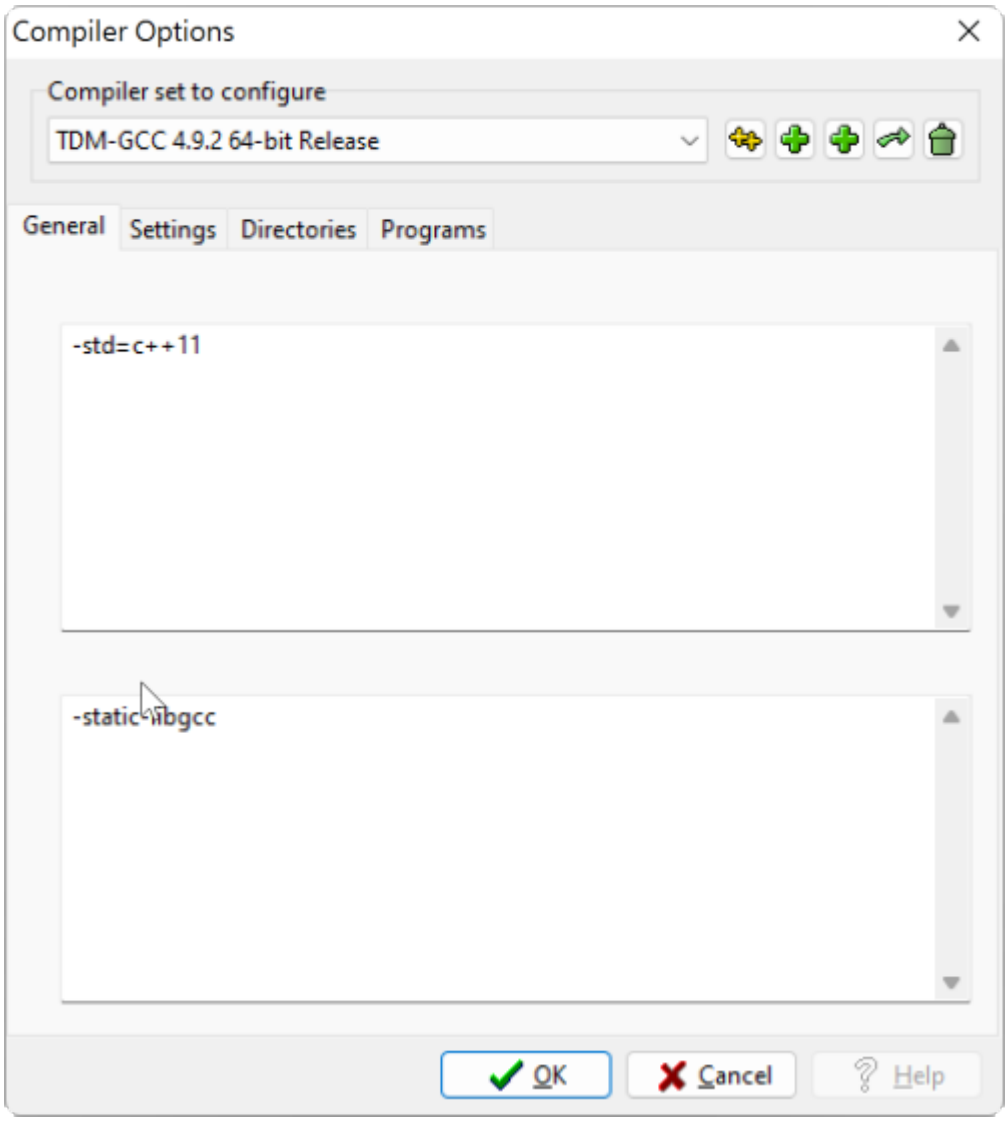
我們已經使用Dev-C++示範了Example 1的hello.cpp程式的開發與執行過程，建議讀者依照前述的步驟實際進行一遍，以掌握Dev-C++的使用方式。最後，我們將Dev-C++時常會用到的「編譯」、「執行」、「編譯並執行」以及「全部重新建置」等常用的功能，特別為讀者歸納其操作方法：

功能	選單	快速鍵	工具列圖示	備註
編譯	「執行>編譯」	F9		將原始程式加以編譯以產生可執行檔。
執行	「執行>執行」	F10		執行可執行檔。
編譯並執行	「執行>編譯並執行」	F11		將原始程式加以編譯以產生可執行檔，並且加以執行。
全部重新建置	「執行>全部重新建置」	F12		不論原始程式是否修改過，強制將其加以編譯以產生可執行檔。

Tab. 1: Dev-<nowiki

附錄1.1.2 如何在Dev-<nowiki>C++</nowiki>裡設定編譯器參數

本書有部份的範例程式在編譯時，需要設定相關的參數才能正確編譯，例如部份範例使用到C++11標準的語法，因此必須在編譯時加上「-std=gnu11」或「-std=<nowiki>C++</nowiki>11」的參數才行。讀者可以透過Dev-<nowiki>C++</nowiki>選單中的「工具(Tools)>編譯器選項(Compiler Options)」來設定相關的編譯器參數，請參考<imgref Fig_DevCPPCompilerOpt>」 <WRAP center> <imgcaption Fig_DevCPPCompilerOpt center Dev-C++的編譯器選項設定。>



</imgcaption> </WRAP>

記得要勾選Add the following commands when calling the compiler.選項，並在下的文字輸入區域內輸入所要加入的編譯器參數，例如在figure ##加入了-std=C++11的參數。新增(或修改）完參數後，只要按下OK下次編譯程式時就會發生作用。

事實上，如果時常有需要變更「編譯器參數」的需求時，使用本小節的方法並不是很方便，建議讀者可以改用下一小節的方法，直接在「命令提示字元(Command Prompt)裡執行編譯器指令並給定其所需的參數，這樣會更為便利。

附錄1.1.3 使用命令提示字元視窗在Windows上以指令來編譯及執行範例程式

由於Dev-C++的安裝內含了TDM-GCC編譯器 – 相容於Windows作業系統的GCC編譯器分支，因此我們也可以在Windows的命令提示字元視窗裡，使用「指令」來完成範例程式的編譯與執行。當然，你仍然可以在Dev-C++裡完成這些工作，此處的目的僅是為讀者提供額外的操作方法而已。

以下的說明皆假設Dev-C++是安裝在預設的C:\Program Files (x86)\Dev-Cpp目錄

之下，若你將Dev-C++安裝在不同的目錄，請自行將本節的內容進行相關的修改。

TDM-GCC預設會被安裝在Dev-C++的安裝目錄下的MinGW64\bin裡(意即C:\Program Files (x86)\Dev-Cpp\MinGW64\bin)請讀者先將TDM-GCC編譯器所在的目錄加入到Windows系統的PATH環境變數裡，以便利未來的使用。以Windows 11為例，你可以透過「控制台>系統及安全性>系統>系統資訊>進階系統設定>環境變數」來將環境變數的編輯視窗叫出，請參考figure 7

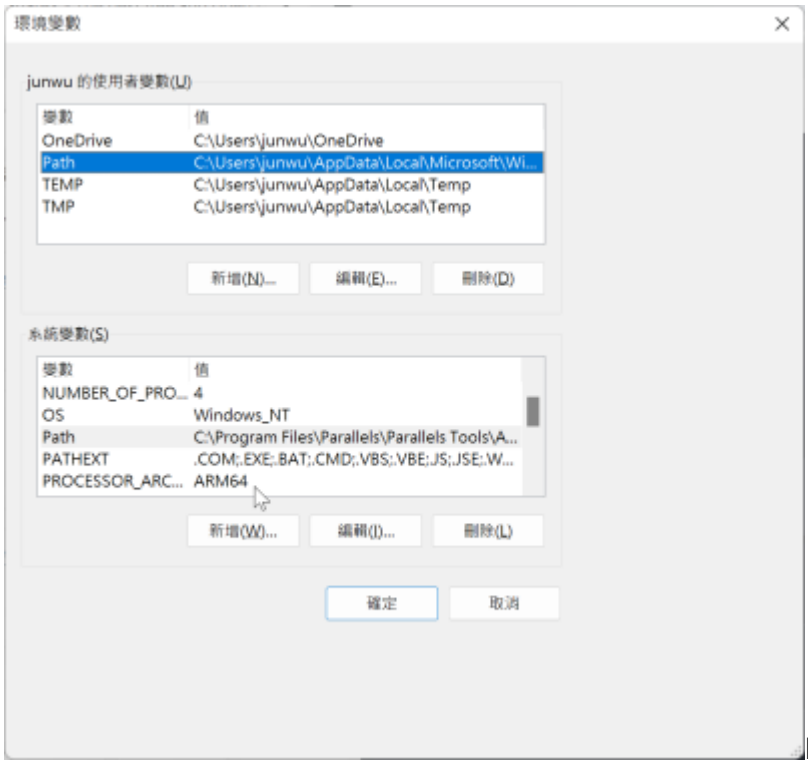


Fig. 7: Windows 11的環境變數設定畫面之1。

請依需求選擇編輯「自己的使用者變數」或「系統變數」裡的PATH[1)]，按下編輯以開啟如figure 8的對話窗，按下「新增」按鈕並加入C:\Program Files (x86)\Dev-Cpp\MinGW64\bin將TDM-GCC所在的目錄新增到PATH裡。

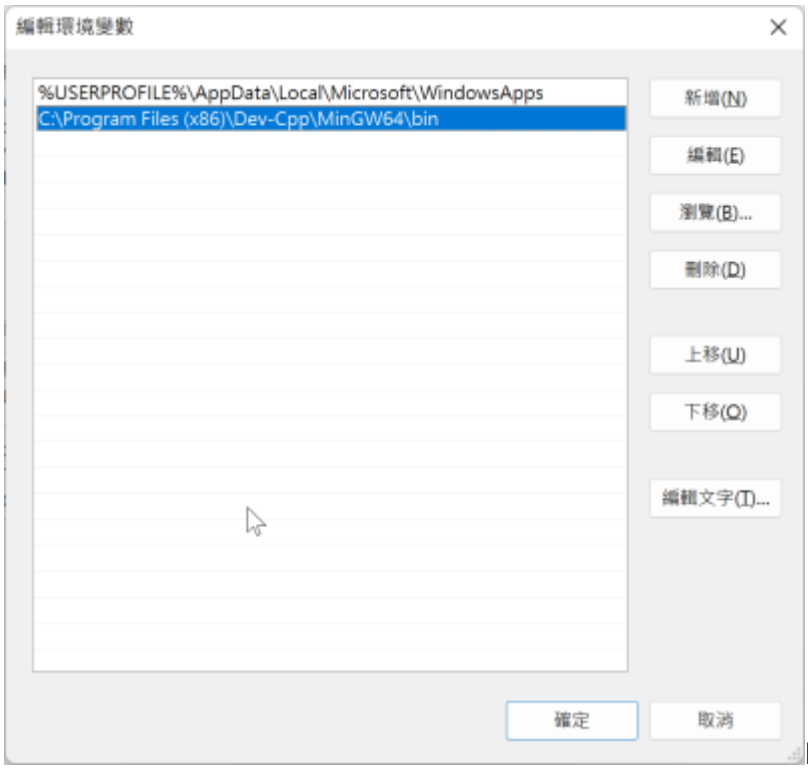
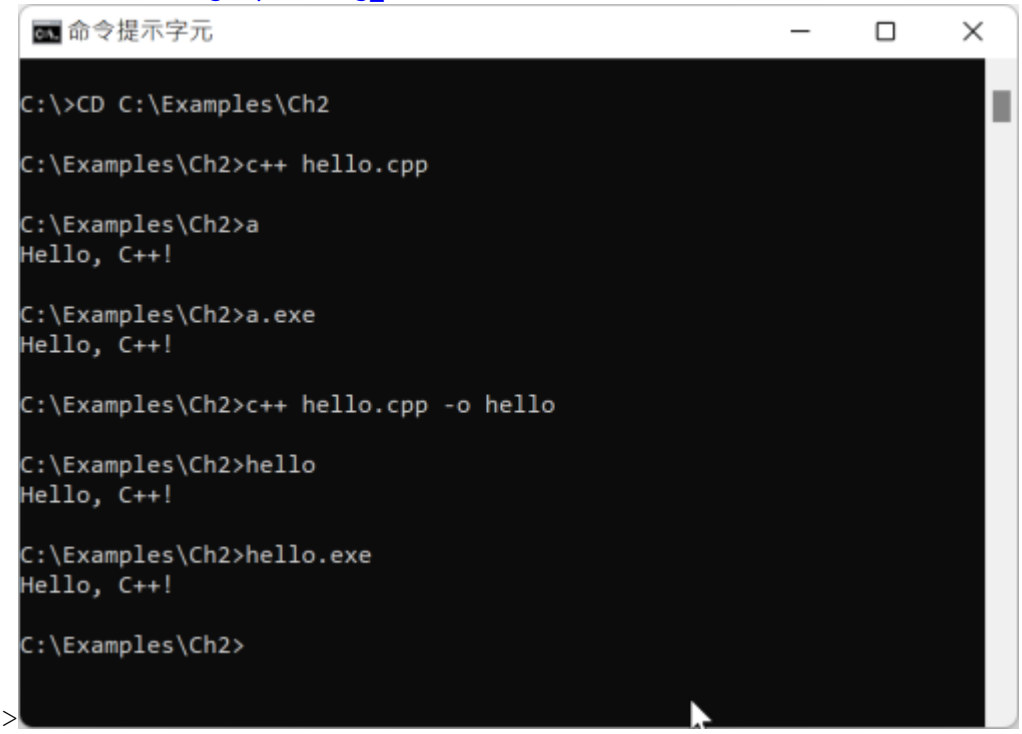


Fig. 8: Windows 11的環境變數設定畫面之2。

完成上述設定後，你就可以在任何目錄裡直接下達C++或「g」指令(TDM-GCC的C++編譯器的檔名)，來進行C++程式的編譯了。請打開「命令提示字元(Command Prompt)並依下列步驟操作：

- 步驟1:下達CD指令，切換到範例程式所在的目錄，例如CD C:\Examples\Ch2
- 步驟2:下達C++或g指令，進行原始程式的編譯，例如<nowiki>C++</nowiki> hello.cpp或「g++ hello.cpp如果編譯的結果正確，預設將產生一個名為a.exe的可執行檔。如果要改變可執行檔的名稱，也可以用-o參數指定，例如<nowiki>C++</nowiki> hello.cpp -o hello * 步驟3:輸入可執行檔的檔名(主檔名即可，但若下達包含副檔名在內的完整檔名也可以)，即可加以執行。我們將以上的步驟以Example 1的hello.cpp為例，彙整示範如<imgref Fig_Win11Execution>所示<WRAP center> <imgcaption Fig_Win11Execution center Windows 11的環境變數設定畫面



之2。>

</imgcaption> </WRAP> 從figure ##中可看出，預設的可執行檔(a.exe)□可以用□a□或「a.exe□來加以執行；同樣的，若使用□-o hello□做為參數，則其所產生的可執行檔將會是□hello.exe□□可使用□hello□或「hello.exe□加以執行。 最後，本書其實會擷取編譯及執行結果的「文字內容」，來代替命令提示字元視窗的截圖，請參考以下的編譯及執行結果的「文字內容」：

```
C:\>CD C:\Examples\Ch2

C:\Examples\Ch2><nowiki>C++</nowiki> hello.cpp

C:\Examples\Ch2>a
Hello, <nowiki>C++</nowiki>!

C:\Examples\Ch2>a.exe
Hello, <nowiki>C++</nowiki>!

C:\Examples\Ch2><nowiki>C++</nowiki> hello.cpp -o hello

C:\Examples\Ch2>hello
Hello, <nowiki>C++</nowiki>!

C:\Examples\Ch2>hello.exe
Hello, <nowiki>C++</nowiki>!

C:\Examples\Ch2>
```

如此一來，不論文字指令與執行結果，比起截圖更容易看得更為清楚。

附錄1.2 Linux與MacOS作業系統

由於MacOS源自BSD□它和Linux系統一樣都是衍生自Unix作業系統，因此在MacOS和Linux作業系統裡都可以使用終端機(Terminal)來進行程式的編譯與執行，本附錄將這兩種作業系統一同進行說明。假設Example 1的hello.cpp被下載於□~/examples/ch2□目錄裡，請開啟一個「終端機(Terminal)□視窗，並依下列步驟操作： * 步驟1:下達cd指令，切換到範例程式所在的目錄，例如□cd ~/examples/ch2□□ * 步驟2:下達C++或g指令，進行原始程式的編譯，例如□C++ hello.cpp□或「g++ hello.cpp□□如果編譯的結果正確，預設將產生一個名為□a.out□的可執行檔。如果要改變可執行檔的名稱，也可以用□-o□參數指定，例如□C++ hello.cpp -o hello□□

- 步驟3:輸入□./a.out□來執行編譯後所產生的□a.out□可執行檔，若是使用□-o hello□參數將可執行另行命名，則請輸入□./hello□□注意，此處指令中的「./」代表所要執行的是在目前目錄裡的可執行檔。

我們將以上的步驟以Example 1的hello.cpp為例，彙整如下：

```
[user@urlinux ~]$ cd ~/examples/ch2
[user@urlinxu ch2]$ <nowiki>C++</nowiki> hello.cpp
[user@urlinux ch2]$ ./a.out
Hello, <nowiki>C++</nowiki>!
[user@urlinux ch2]$ <nowiki>C++</nowiki> hello.cpp -o hello
[user@urlinux ch2]$ ./hello
```

```
Hello, <nowiki>C++</nowiki>!
[user@urlinux ch2]$
```

[1)] 使用者變數的設定只能套用在使用者本身，但系統變數的設定則可套用在系統內的所有使用者。