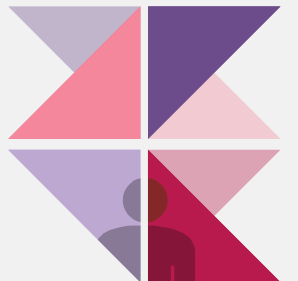




# Introduction of C





# Index

## 01. History

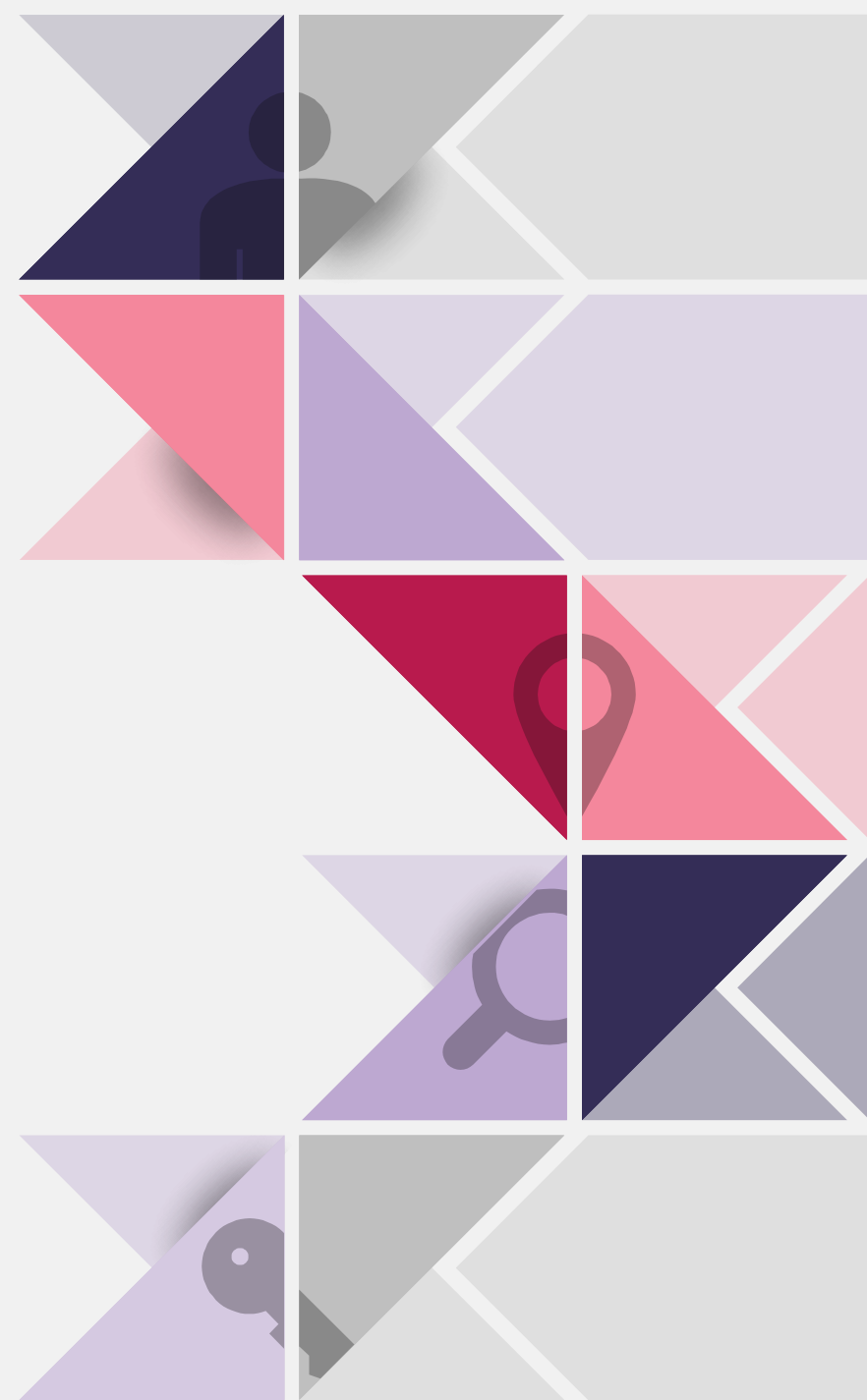
- Alan Mathison Turing
- Why C?
- Inspiration

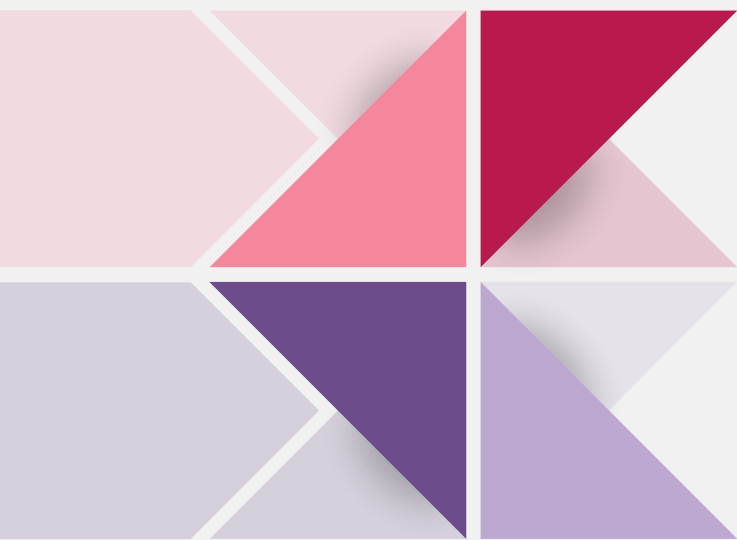
## 02. Compile and Execution

- Process
- Environment

## 03. Let's go C

- Introduction





# 01

## History

# History

Computer Science Father

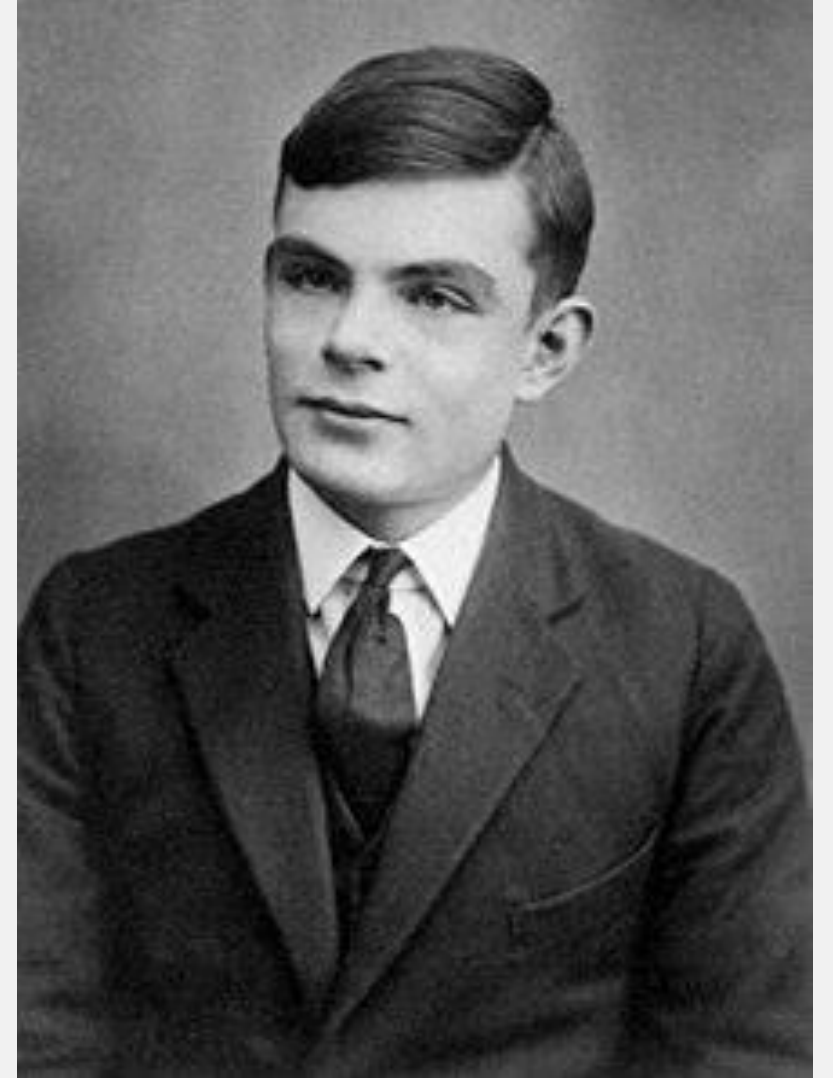
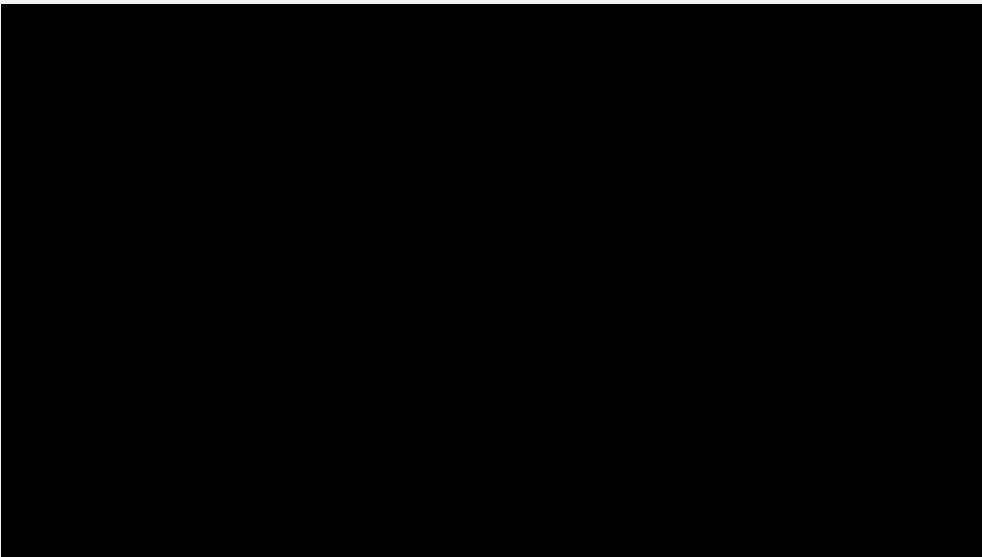
Alan Mathison Turing

艾倫·麥席森·圖靈

1912.06 - 1954.06 (41歲)

研究領域: 數學、密碼分析、邏輯學及電腦科學、數理生物學

運動領域: 是一位世界級的長跑運動員，馬拉松最好成績是2小時46分03秒(手動計時)，比1948年奧林匹克運動會金牌成績慢11分鐘。1948年的一次越野賽跑中，他跑贏了同年奧運會銀牌得主湯姆·理查茲。



[Link](#)

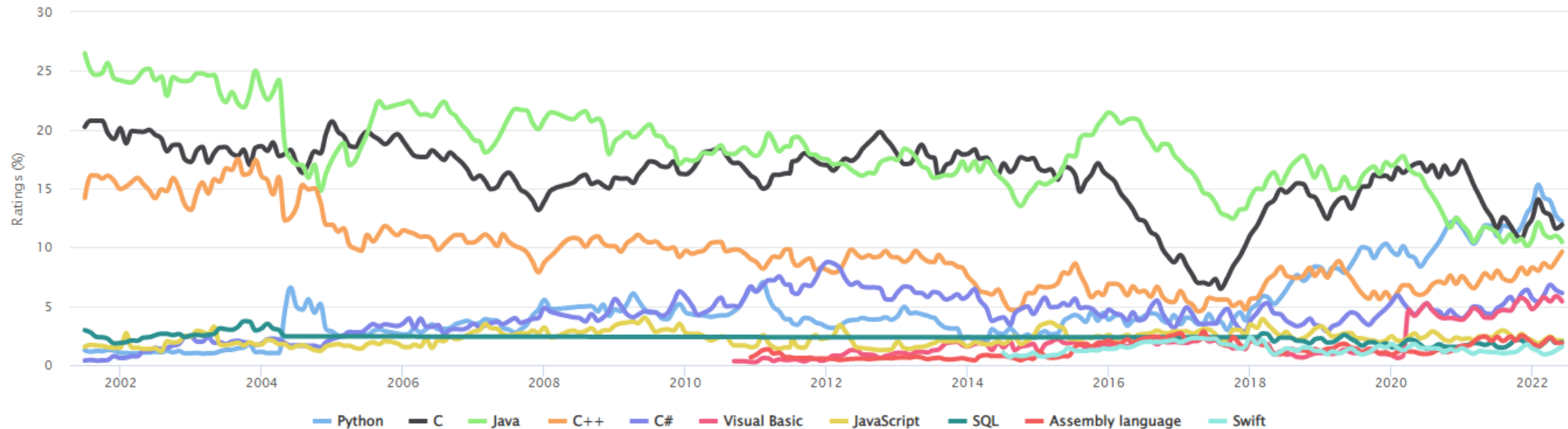
# History

## Why C?

| Jun 2022 | Jun 2021 | Change | Programming Language   | Ratings | Change |
|----------|----------|--------|--|---------|--------|
| 1        | 2        | ▲      |  Python | 12.20%  | +0.35% |
| 2        | 1        | ▼      |  C      | 11.91%  | -0.64% |
| 3        | 3        |        |  Java   | 10.47%  | -1.07% |
| 4        | 4        |        |  C++    | 9.63%   | +2.26% |

TIOBE Programming Community Index

Source: [www.tiobe.com](http://www.tiobe.com)

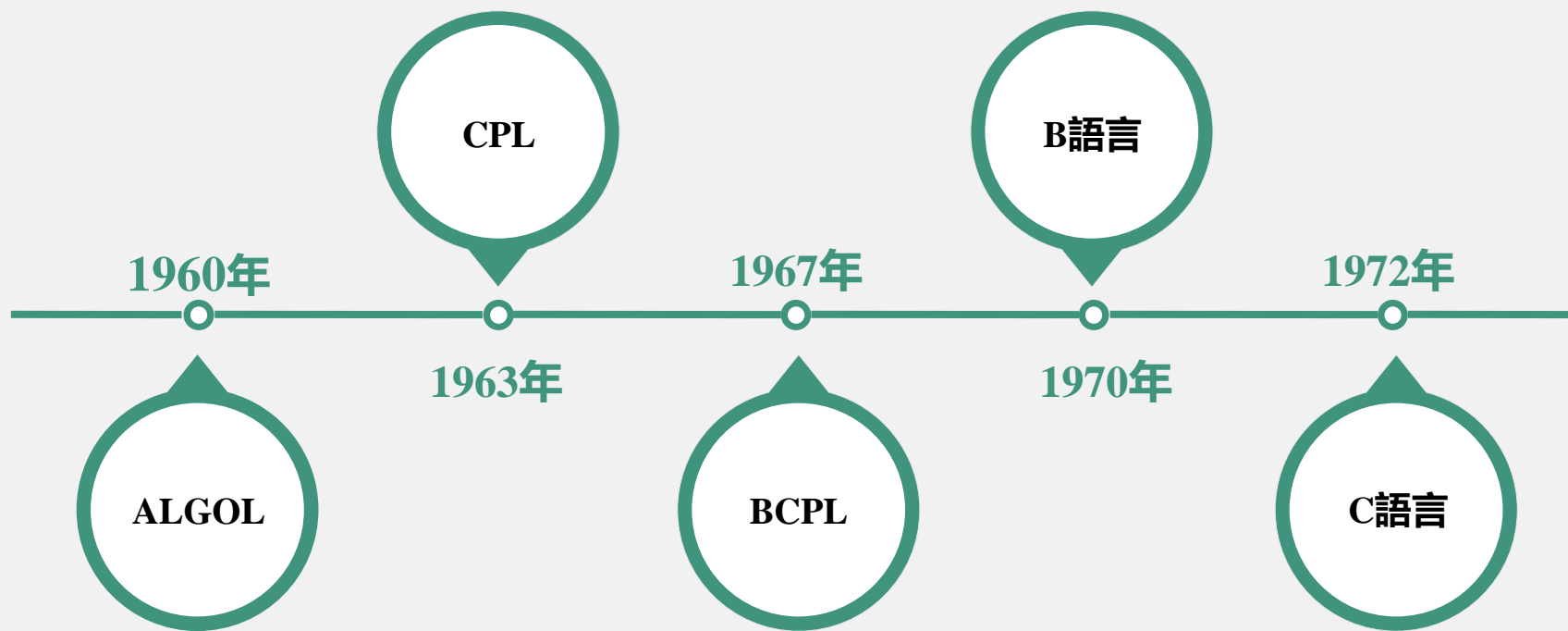


[Link](#)

# History

## Inspiration

C是美國貝爾實驗室的Dennis Ritchie以B語言為基礎所開發的，現今Windows與UNIX系列的作業系統內大部分也是用C語言設計的



# History

Inspiration



Martin Richards



Ken Thompson

Dennis Ritchie

DEC PDP-7



[Link](#)

# History

Inspiration

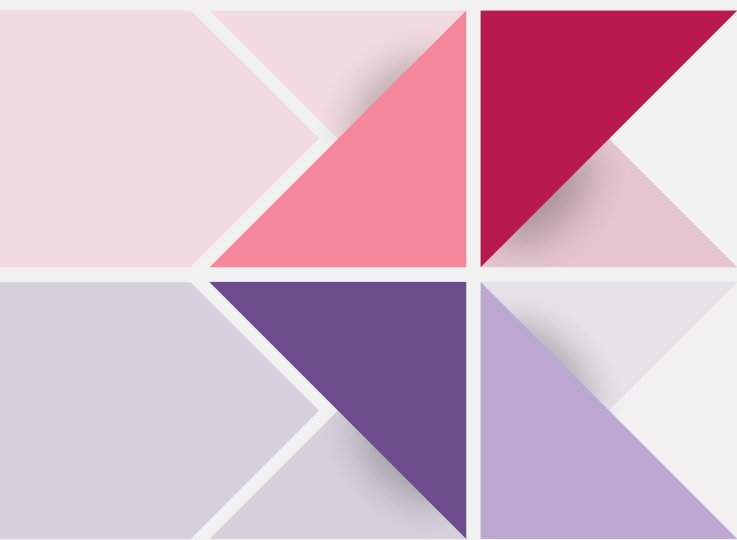
## 優點

1. 簡潔方便 - 只有32個關鍵字，9個控制語句
  2. 允許直接存取記憶體位址，控制硬體
  3. 可移植性高 - 不同機器上的C編譯程式有86%是公用程式碼，編譯程式方便移植
- ...

## 缺點

語法限制較不嚴格，使得程式安全性較低，因此從應用的角度來看，C語言使用者對於程式設計的熟練度要高一點





**02**

# **Compile and Execution**

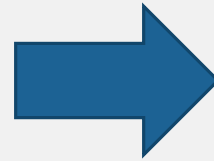
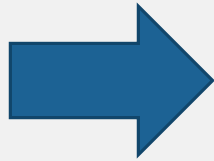
# Compile and Execution

Process

## 使用電腦完成想要的目的

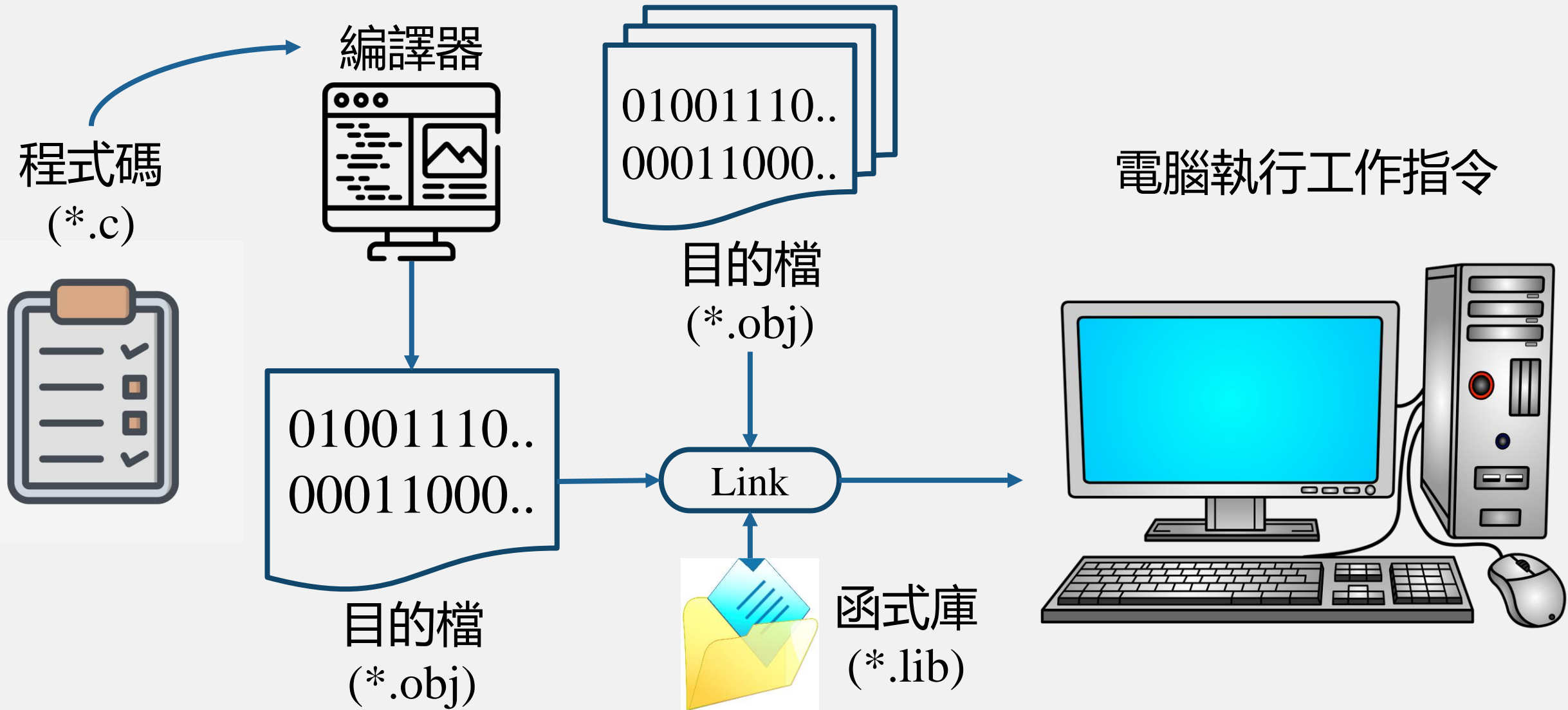
程式決定指令執行內容 (程式碼)

電腦執行工作指令



# Compile and Execution

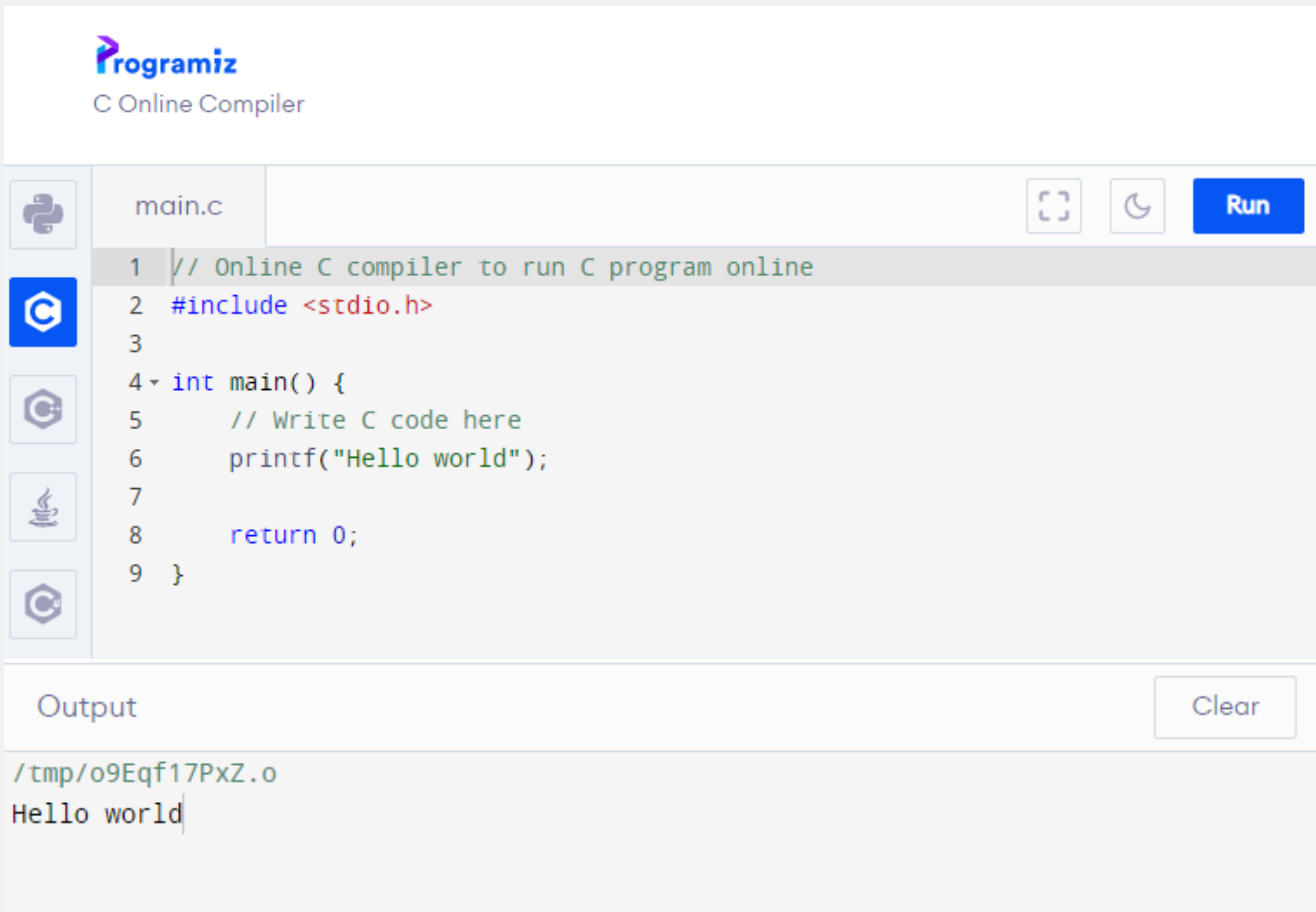
## Process



# Compile and Execution

## Environment

### [Programiz](#)

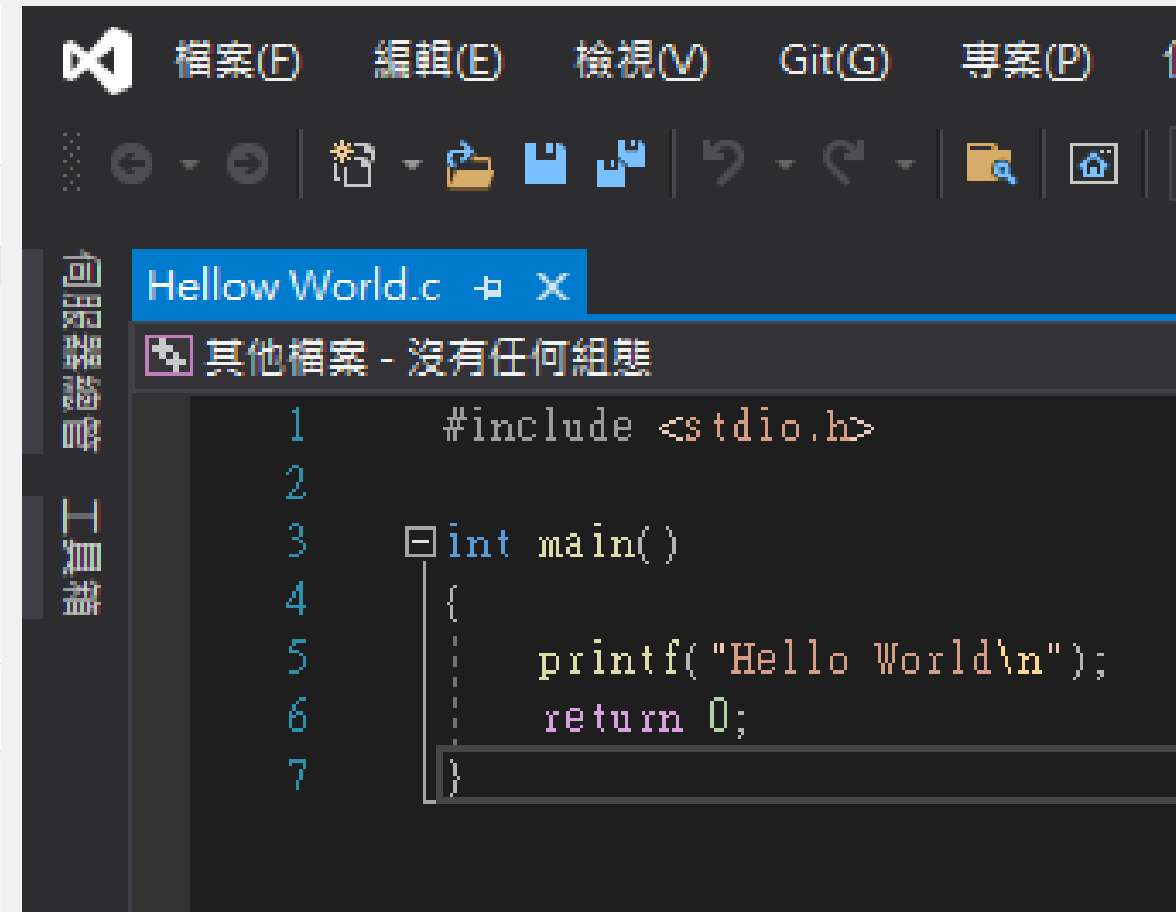


The screenshot shows the Programiz C Online Compiler interface. At the top, the logo and name 'Programiz C Online Compiler' are visible. Below this, a file named 'main.c' is open in the editor. The code is as follows:

```
1 // Online C compiler to run C program online
2 #include <stdio.h>
3
4 int main() {
5     // Write C code here
6     printf("Hello world");
7
8     return 0;
9 }
```

On the right side of the editor, there are icons for full screen, refresh, and a blue 'Run' button. Below the editor is an 'Output' section with a 'Clear' button. The output shows the file path '/tmp/o9Eqf17PxZ.o' and the text 'Hello world'.

### [Visual Studio](#)



The screenshot shows the Visual Studio code editor interface. The top menu bar includes options like '檔案(F)', '編輯(E)', '檢視(V)', 'Git(G)', and '專案(P)'. Below the menu is a toolbar with various icons. The editor window shows a file named 'Hellow World.c' (note the typo in the image). The code is as follows:

```
1 #include <stdio.h>
2
3 int main( )
4 {
5     printf("Hello World\n");
6     return 0;
7 }
```

The left sidebar shows the 'Solution Explorer' and 'File Explorer' panels. The 'Solution Explorer' panel shows a project named '其他檔案 - 沒有任何組態'.

# Compile and Execution

Environment - Visual Studio

優點

整合了編譯器與介面

缺點

啟動慢，檔案開啟速度慢

## 下載



### Visual Studio 2022

Windows | 17.2 版

適用於 Windows 上的 .NET 和 C++ 開發人員的最佳全方位 IDE。全套工具和功能，提升和增強軟體開發的每個階段。

#### 社群

功能強大的 IDE，學生、開放原始碼參與者及個人均可免費使用

免費下載

#### Professional

Professional  
小組

免費

### Visual Studio Installer

即將完成... 一切即將就緒。

- 已下載
- 已安裝

# Compile and Execution

## Environment - Visual Studio

正在安裝 - Visual Studio Enterprise 2019 - 16.3.2


工作負載

個別元件


語言套件

安裝位置


Web 與雲端 (4)




ASP.NET 與網頁程式開發  
使用 ASP.NET Core、ASP.NET、HTML/JavaScript 及容器 (包括 Docker 支援) 建立 Web 應用程式。



Python 開發  
對 Python 進行編輯、偵錯、互動式開發及原始檔控制。




Azure 開發  
用於使用 .NET Core 和 .NET Framework 開發雲端應用程式及建立資源的 Azure SDK、工具及專案，並包含應用程式...




Node.js 開發  
使用非同步的事件驅動 JavaScript 執行階段 Node.js 建置可調整的網路應用程式。


Windows (3)



.NET 桌面開發  
使用 C#、Visual Basic 及 F#，利用 .NET Core 和 .NET Framework 建置 WPF、Windows Forms 與主控台應用程...



使用 C++ 的桌面開發  
使用您選擇的工具 (包括 MSVC、Clang、CMake 或 MSBuild)，建置適用於 Windows 的新式 C++ 應用程式。



通用 Windows 平台開發  
使用 C#、VB 或選用 C++，來建立適用於通用 Windows 平台的應用程式。

位置

C:\Program Files (x86)\Microsoft Visual Studio\2019\Enterprise [變更...](#)

繼續執行表示您同意所選 Visual Studio 版本的 [授權](#)。您也可以選擇下載其他軟體搭配 Visual Studio 一起使用。一如 [第三方聲明](#) 或其隨附的授權所述，此軟體為分開授權。繼續執行表示您也同意這些授權。

所需空間總計為 652 MB

在下載時安裝

安裝

安裝詳細資料

Visual Studio 核心編輯器

Visual Studio 核心殼層體驗，包括語法感知程式碼編輯、原始程式碼控制及工作項目管理。

# Compile and Execution

## Environment - Visual Studio



### 設定新的專案

#### 主控台應用程式

C++ Windows 主控台

專案名稱(J)

HelloWorld

位置(L)

D:\Class\C\Example\01\

方案(S)

建立新方案

解決方案名稱(M) ⓘ

HelloWorld

☐ 將解決方案與專案置於相同目錄中(D)

### 建立新專案

#### 最近使用的專案範本(R)

您最近所存取範本的清單會顯示在這裡。

搜尋範本 (Alt+S)(S)

所有語言(L)

所有平台(P)

所有專案類型(T)



空白專案

使用適用於 Windows 的 C++ 從頭開始，不會提供任何起始檔案。

C++

Windows

主控台



主控台應用程式

在 Windows 終端機中執行程式碼，根據預設，會列印 "Hello World"。

C++

Windows

主控台



CMake 專案

建置不依賴 .sln 或 .vcxproj 檔案的新式跨平台 C++ 應用程式。

C++

Windows

Linux

主控台



Windows 傳統式精靈

使用精靈建立您自己的 Windows 應用程式。

C++

Windows

桌面

主控台

程式庫



Windows 傳統型應用程式

此專案適用於在 Windows 執行並有圖形化使用者介面的應用程式。

C++

Windows

桌面



空白方案

下一步(N)

# Compile and Execution

## Environment - Visual Studio

```
HellowWorld.cpp  HellowWorld
1  // HellowWorld.cpp : 此檔案包含 'main' 函式。程式會於該處開始執行及結束執行。
2  //
3
4  #include <iostream>
5  #include <stdio.h>
6  int main()
7  {
8      printf("Hello World!\n");
9      return 0;
10 }
11
12 // 執行程式: Ctrl + F5 或 [偵錯] > [啟動但不偵錯] 功能表
13 // 偵錯程式: F5 或 [偵錯] > [啟動偵錯] 功能表
14
15 // 開始使用的提示:
16 // 1. 使用 [方案總管] 視窗，新增/管理檔案
17 // 2. 使用 [Team Explorer] 視窗，連線到原始檔控制
18 // 3. 使用 [輸出] 視窗，參閱組建輸出與其他訊息
19 // 4. 使用 [錯誤清單] 視窗，檢視錯誤
20 // 5. 前往 [專案] > [新增項目]，建立新的程式碼檔案，或是前往 [專案] > [新增現有項目]，將現有程式碼檔案新增至專案
21 // 6. 之後要再次開啟此專案時，請前往 [檔案] > [開啟] > [專案]，然後選取 .sln 檔案
22
```



### Microsoft Visual Studio 偵錯主控台

Hello World!

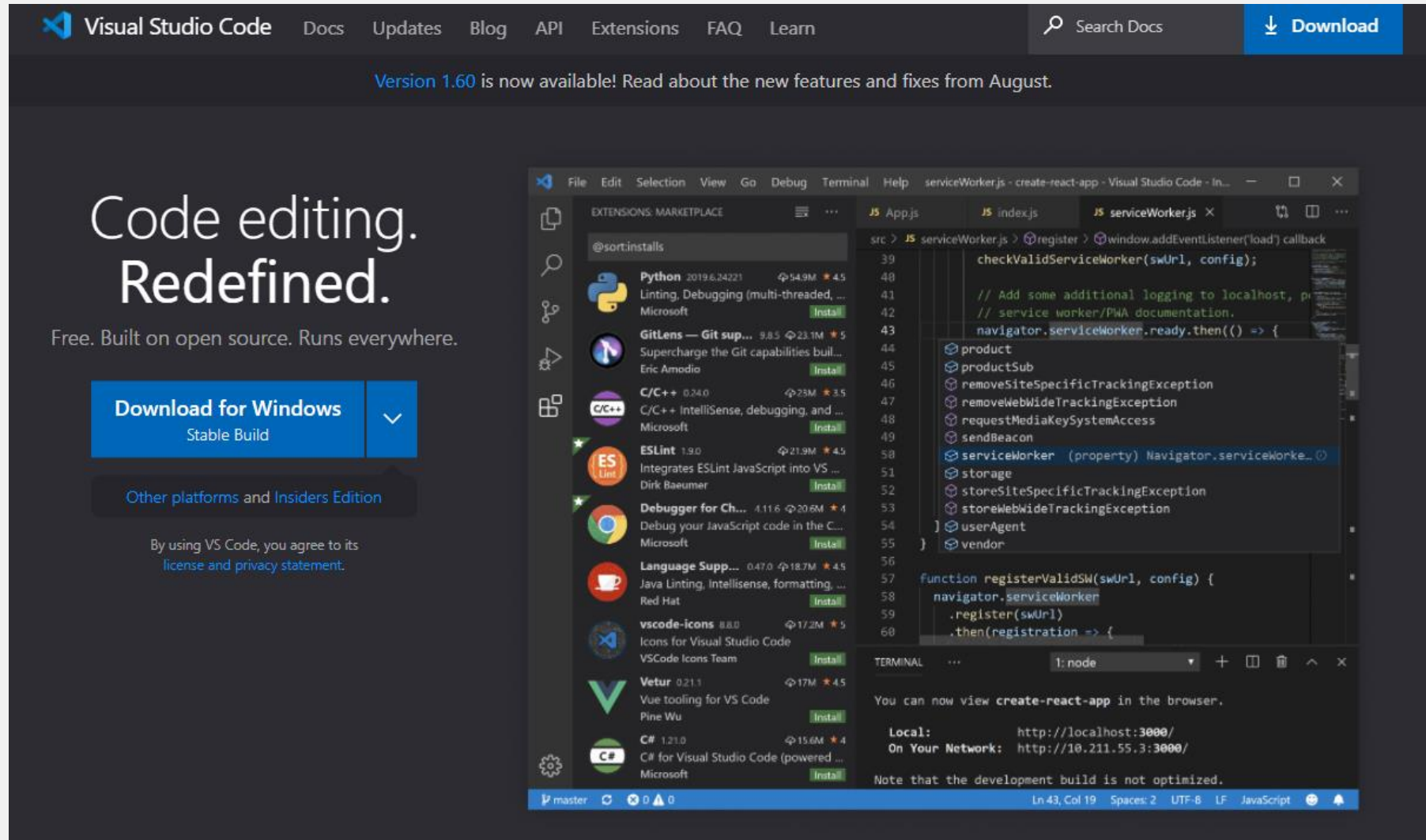
D:\Class\C\Example\01\HellowWorld\Debug\HellowWorld.exe (處理序 7704) 已結束，出現代碼 0。  
若要在偵錯停止時自動關閉主控台，請啟用 [工具] -> [選項] -> [偵錯] -> [偵錯停止時，自動關閉主控台]。  
按任意鍵關閉此視窗...



# Compile and Execution

Environment - Visual Studio Code (VS Code)

## [Visual Studio Code](#)



The image shows the Visual Studio Code website and a code editor window. The website header includes links for Docs, Updates, Blog, API, Extensions, FAQ, and Learn, along with a Search Docs button and a Download button. A banner announces Version 1.60. The main content area features the text "Code editing. Redefined." and "Free. Built on open source. Runs everywhere." with a "Download for Windows Stable Build" button. Below this, it mentions "Other platforms and Insiders Edition" and a link to the license and privacy statement. The code editor window shows a file named "serviceWorker.js" with JavaScript code for a service worker. The left sidebar displays the "EXTENSIONS: MARKETPLACE" with a list of extensions including Python, GitLens, C/C++, ESLint, Debugger for Chrome, Language Support for Java, vscode-icons, Vetur, and C#.

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn

Search Docs Download

Version 1.60 is now available! Read about the new features and fixes from August.

## Code editing. Redefined.

Free. Built on open source. Runs everywhere.

[Download for Windows](#)  
Stable Build

[Other platforms and Insiders Edition](#)

By using VS Code, you agree to its [license and privacy statement](#).

**EXTENSIONS: MARKETPLACE**

- @sort:installs
- Python** 2019.6.24221 54.9M 4.5  
Linting, Debugging (multi-threaded, ...  
Microsoft [Install](#)
- GitLens — Git sup...** 9.8.5 23.1M 5  
Supercharge the Git capabilities buil...  
Eric Amodio [Install](#)
- C/C++** 0.24.0 23M 3.5  
C/C++ IntelliSense, debugging, and ...  
Microsoft [Install](#)
- ESLint** 1.9.0 21.9M 4.5  
Integrates ESLint JavaScript into VS ...  
Dirk Baeumer [Install](#)
- Debugger for Ch...** 4.11.6 20.6M 4  
Debug your JavaScript code in the C...  
Microsoft [Install](#)
- Language Supp...** 0.47.0 18.7M 4.5  
Java Linting, Intellisense, formatting, ...  
Red Hat [Install](#)
- vscode-icons** 8.8.0 17.2M 5  
Icons for Visual Studio Code  
VSCode Icons Team [Install](#)
- Vetur** 0.21.1 17M 4.5  
Vue tooling for VS Code  
Pine Wu [Install](#)
- C#** 1.21.0 15.6M 4  
C# for Visual Studio Code (powered ...  
Microsoft [Install](#)

**JS serviceWorker.js**

```
src > JS serviceWorker.js > register > window.addEventListener('load') callback
39
40
41
42 // Add some additional logging to localhost, p
43 // service worker/PWA documentation.
44 navigator.serviceWorker.ready.then(() => {
45
46   product
47   productSub
48   removeSiteSpecificTrackingException
49   removeWebWideTrackingException
50   requestMediaKeySystemAccess
51   sendBeacon
52   serviceWorker (property) Navigator.serviceWorke...
53   storage
54   storeSiteSpecificTrackingException
55   storeWebWideTrackingException
56   userAgent
57   vendor
58
59 function registerValidSW(swUrl, config) {
60   navigator.serviceWorker
    .register(swUrl)
    .then(registration => {
```

**TERMINAL** ... 1: node

You can now view **create-react-app** in the browser.

Local: http://localhost:3000/  
On Your Network: http://10.211.55.3:3000/

Note that the development build is not optimized.

Ln 43, Col 19 Spaces: 2 UTF-8 LF JavaScript

# Compile and Execution

## Environment - Compile (Mingw)

[Link](#)

MinGW Installation Manager Setup Tool

mingw-get version 0.6.2-beta-20131004-1



Written by Keith Marshall

Copyright © 2009-2013, MinGW.org Project

<http://mingw.org>

This is free software; see the product documentation or source code, for copying and redistribution conditions. There is NO WARRANTY; not even an implied WARRANTY OF MERCHANTABILITY, nor of FITNESS FOR ANY PARTICULAR PURPOSE.

This tool will guide you through the first time setup of the MinGW Installation Manager software (mingw-get) on your computer; additionally, it will offer you the opportunity to install some other common components of the MinGW software distribution.

After first time setup has been completed, you should invoke the MinGW Installation Manager directly, (either the CLI mingw-get.exe variant, or its GUI counterpart, according to your preference), when you wish to add or to remove components, or to upgrade your MinGW software installation.

View Licence

Install

Cancel

MinGW Installation Manager Setup Tool

mingw-get version 0.6.2-beta-20131004-1



Step 1: Specify Installation Preferences

Installation Directory

C:\MinGW

Change

If you elect to change this, you are advised to avoid any choice of directory which includes white space within the absolute representation of its path name.

User Interface Options

Both command line and graphical options are available. The command line interface is always supported; the alternative only if you choose the following option to ...

☒ ... also install support for the graphical user interface.

Program shortcuts for launching the graphical user interface should be installed ...

☐ ... just for me (the current user), or ... ☒ ... for all users \* ...

☒ ... in the start menu, and/or ... ☒ ... on the desktop.

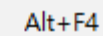
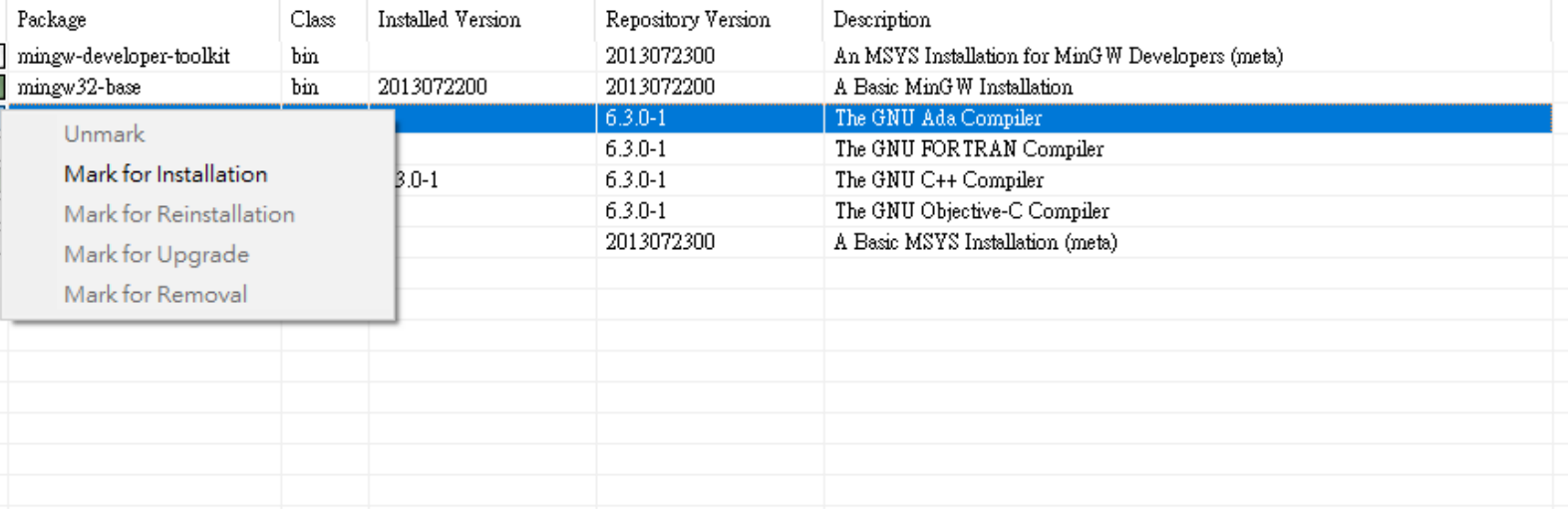
\* selection of this option requires administrative privilege.

View Licence

Continue

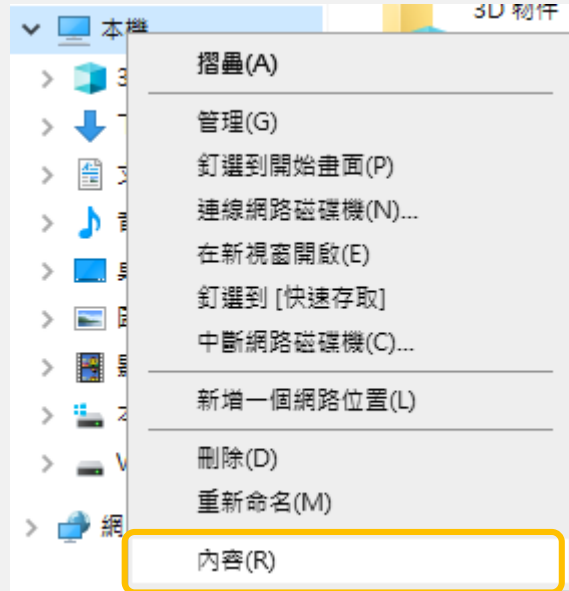
Cancel

## Environment - Compile (Mingw)



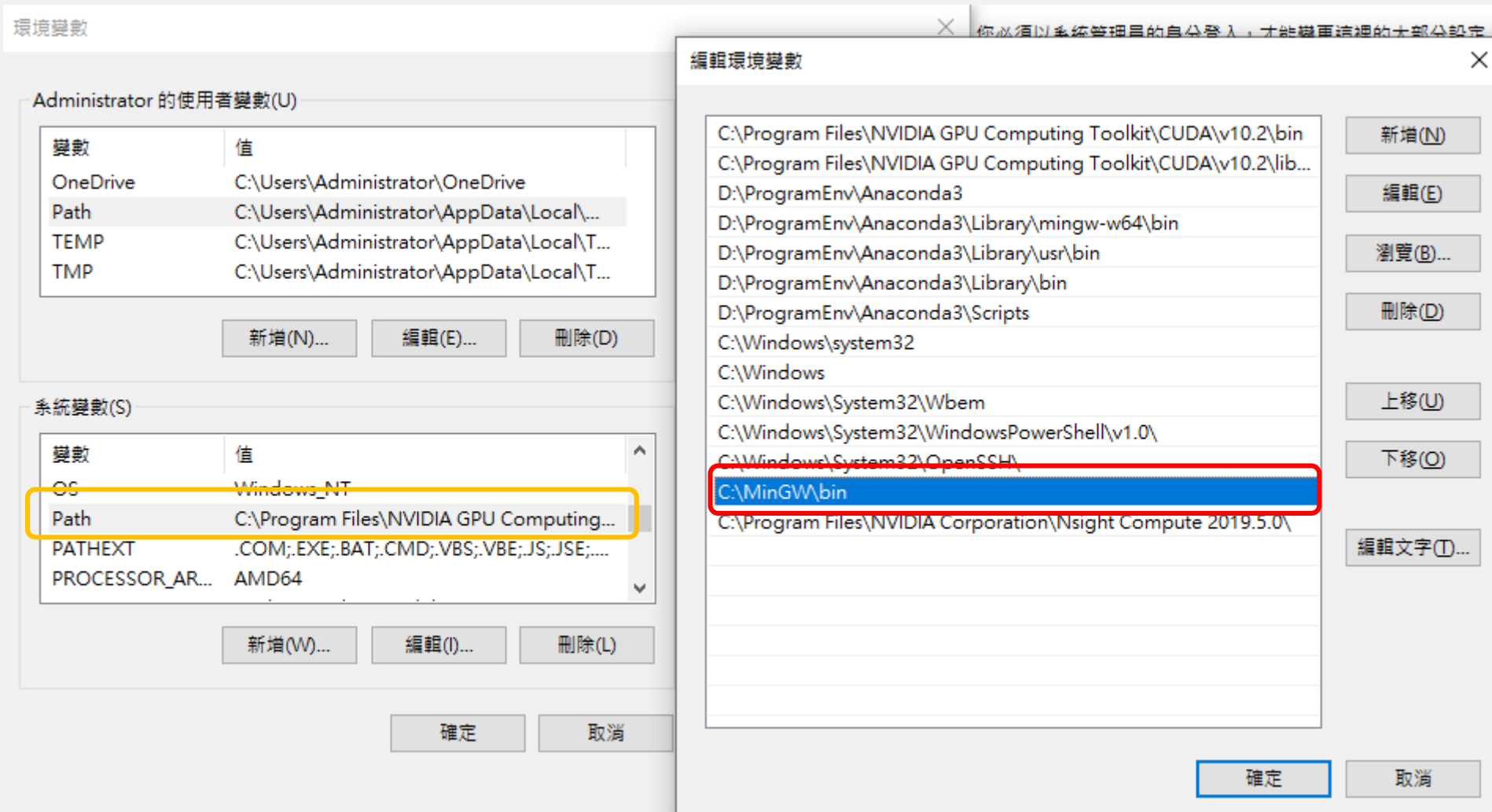
# Compile and Execution

## Environment - Compile (Mingw)



# Compile and Execution

## Environment - Compile (Mingw)



# Compile and Execution

## Environment - Compile (Mingw)



```
系統管理員: 命令提示字元
Microsoft Windows [版本 10.0.19041.1165]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\Administrator>gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=c:/mingw/bin/../../libexec/gcc/mingw32/6.3.0/lto-wrapper.exe
Target: mingw32
Configured with: ../src/gcc-6.3.0/configure --build=x86_64-pc-linux-gnu --host=mingw32 --target=mingw32 --with-gmp=/mingw
--with-mpfr --with-mpc=/mingw --with-isl=/mingw --prefix=/mingw --disable-win32-registry --with-arch=i586 --with-tune=
generic --enable-languages=c,c++,objc,obj-c++,fortran,ada --with-pkgversion='MinGW.org GCC-6.3.0-1' --enable-static --en
able-shared --enable-threads --with-dwarf2 --disable-sjlj-exceptions --enable-version-specific-runtime-libs --with-libic
onv-prefix=/mingw --with-libintl-prefix=/mingw --enable-libstdcxx-debug --enable-libgomp --disable-libvtv --enable-nls
Thread model: win32
gcc version 6.3.0 (MinGW.org GCC-6.3.0-1)
```

# Compile and Execution

Environment - Execution

編譯

```
gcc "filename.c" -o "execute_filename"
```

系統管理員: C:\Windows\system32\cmd.exe

```
C:\Users\Administrator\Desktop\temp>gcc "Hellow World.c" -o hello
```



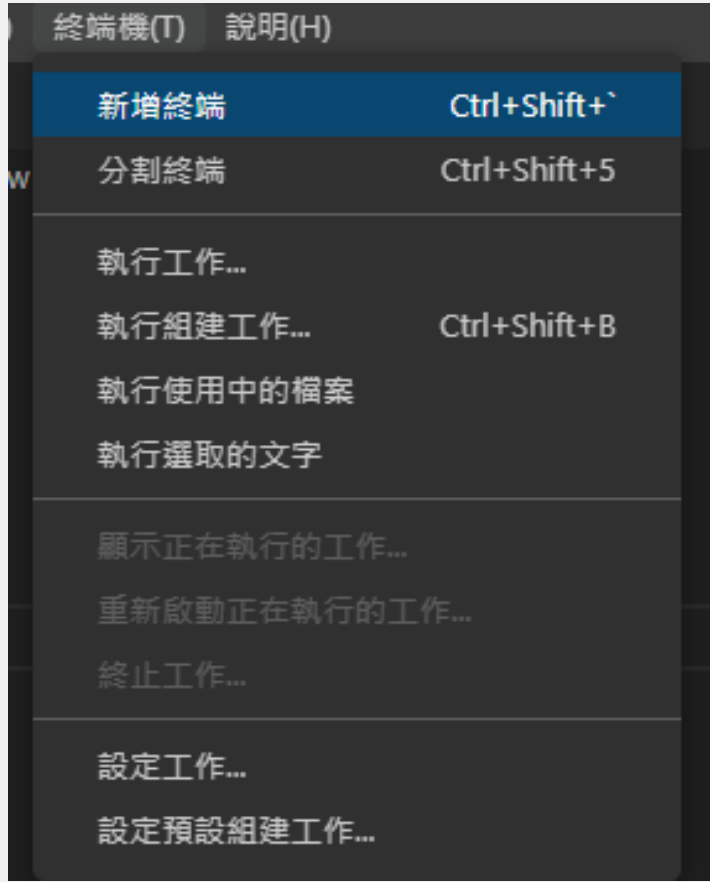
執行

```
execute_filename (.exe)
```

```
C:\Users\Administrator\Desktop\temp>hello
hello
C:\Users\Administrator\Desktop\temp>hello.exe
hello
```

# Compile and Execution

## Environment - Execution

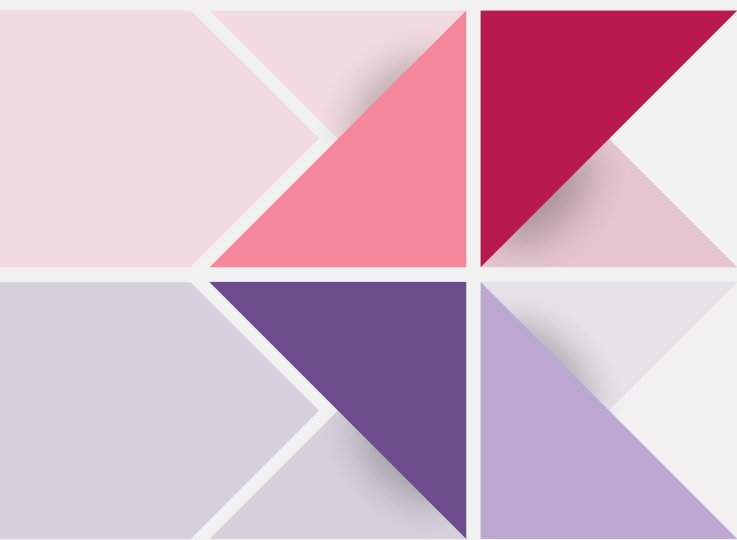


A screenshot of a terminal window with a dark background and white text. The terminal shows the process of compiling and running a C program. The prompt is 'C:\Users\Administrator\Desktop\temp>'. The first command is 'gcc "Hellow World.c" -o hello'. The second command is 'hello', which produces the output 'hello'. The third command is 'hello.exe', which also produces the output 'hello'. The prompt is now 'C:\Users\Administrator\Desktop\temp>' with a cursor.

```
C:\Users\Administrator\Desktop\temp>gcc "Hellow World.c" -o hello

C:\Users\Administrator\Desktop\temp>hello
hello
C:\Users\Administrator\Desktop\temp>hello.exe
hello
C:\Users\Administrator\Desktop\temp>
```





**03**

**Let's go C**

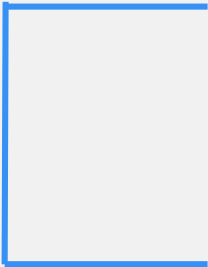
# Let' go C

## Introduction

A simple C program relies on three key language features:

- Directives
- Functions
- Statements

*directives*

*Functions*  `int main (void)`  
`{` *statements*  
`}`

# Let' go C

## Introduction

### An example: File.c

*Functions*

```
C Start C.c X
F: > Course Example > 01 > C Start C.c > main(void)
1
2  /*This is a C program*/
3  #include <stdio.h>
4
5  int main(void)
6  {
7      /*Print a sentence*/
8      printf("C Program GO!\n");
9      printf("歡迎各位進入了程式領域!\n");
10     return 0;
11 }
```

*directives*

*statements*

# Let' go C

## Introduction

### Directives

- Before a C program is compiled, it is first edited by a preprocessor
- Commands intended for the preprocessor are called directives
- Example:
  - ◆ #include <file>
    - ✓ Search from environment path
  - ◆ #include "file"
    - ✓ Search from current path

```
F: > Course Example > 01 > C Start C.c > main(void)
1
2 //This is a C program
3 #include <stdio.h>
4 #include "stdlib.h"
5
6 int main(void)
7 {
8     /*Print a sentence*/
9     printf("C Program GO!\n");
10    printf("歡迎各位進入了程式領域!\n");
11    return 0;
12 }
```

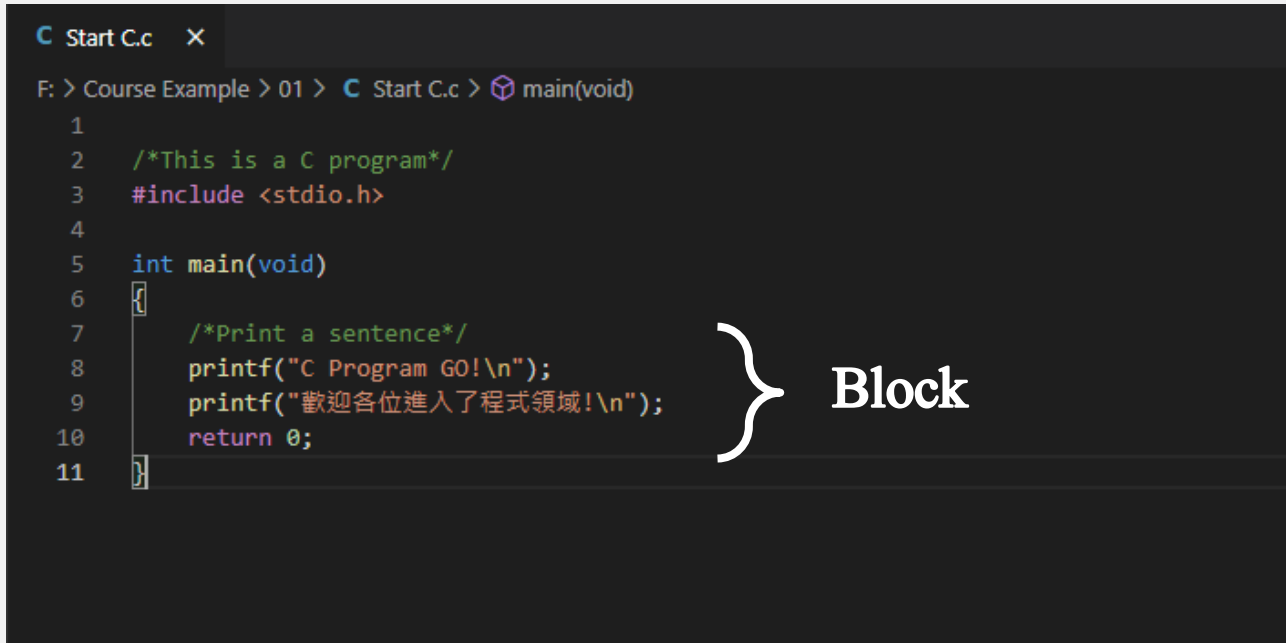
Header

# Let' go C

## Introduction

### main() function

- The main function is mandatory and gets called automatically while the program is executed
  - ✓ main() + curly brackets
  - ✓ Block is the statement in the curly brackets



The screenshot shows a code editor window titled "Start C.c". The file path is "F: > Course Example > 01 > Start C.c > main(void)". The code is as follows:

```
1
2  /*This is a C program*/
3  #include <stdio.h>
4
5  int main(void)
6  {
7      /*Print a sentence*/
8      printf("C Program GO!\n");
9      printf("歡迎各位進入了程式領域!\n");
10     return 0;
11 }
```

A white curly bracket on the right side of the code, spanning from line 6 to line 11, is labeled "Block", indicating that the code between the curly brackets is the main function block.

# Let' go C

## Introduction

## Statement

- It is a command to be executed
- Each statement ends with a semicolon
  - ✓ Exception: compound statement



```
C Start C.c X
F: > Course Example > 01 > C Start C.c > main(void)
1
2  /*This is a C program*/
3  #include <stdio.h>
4
5  int main(void)
6  {
7      /*Print a sentence*/
8      printf("C Program GO!\n");
9      printf("歡迎各位進入了程式領域!\n");
10     return 0;
11 }
```

statements

The image shows a code editor window titled 'C Start C.c'. The code is a C program with line numbers 1 through 11. Lines 8, 9, and 10 are highlighted with white arrows pointing to the word 'statements' on the right. The code is as follows:

```
1
2  /*This is a C program*/
3  #include <stdio.h>
4
5  int main(void)
6  {
7      /*Print a sentence*/
8      printf("C Program GO!\n");
9      printf("歡迎各位進入了程式領域!\n");
10     return 0;
11 }
```

# Let' go C

## Introduction - keywords & Identifiers

### keywords

- They are predefined and reserved words for programming with special meanings to the compiler
- They are parts of the syntax and cannot be as the identifiers

```
int money;
```

| C keywords |        |          |          |
|------------|--------|----------|----------|
| auto       | break  | case     | char     |
| continues  | do     | default  | const    |
| double     | else   | enum     | extern   |
| for        | if     | goto     | float    |
| int        | long   | register | return   |
| signed     | static | sizeof   | short    |
| struct     | switch | typedef  | union    |
| void       | while  | volatile | unsigned |

# Let' go C

## Introduction - keywords & Identifiers

### Identifier

- It refers to name for entity such as variable, function, structure ...
- It must be unique for an entity with unique name to identify

### Rules for naming a identifier

- ✓ A valid identifier have digits, letters, and underscores
- ✓ The first letter should be either a letter or an underscore
- ✓ The keywords cannot be used as identifiers
- ✓ The name of identifier should be meaningful (Not necessary)

```
int money;
```

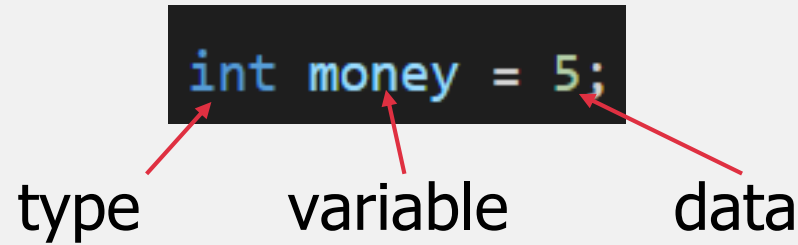


# Let' go C

## Introduction - Variables & Constants

### Variables

- Each variable should be given a unique name to indicate the storage area
- It is a symbolic representation of a memory location and is also a container to store data



The diagram shows the code `int money = 5;` on a black background. Three red arrows point from labels below to parts of the code: 'type' points to 'int', 'variable' points to 'money', and 'data' points to '5'.

```
int money = 5;
```

type                  variable                  data

### Rules for naming a variable

- ✓ Same as naming a identifier


# Let' go C

## Introduction - Variables & Constants

### Constants

- It means a unchangeable variable is created using keyword "const"

```
const int p = 3;  
p = 5;
```



```
example.c: In function 'main':  
example.c:7:7: error: assignment of read-only variable 'p'  
    p = 5;  
    ^
```

# Let' go C

## Introduction - Data Type

### Data Type

- It is the declaration for variable
- It determines the type and size of data associated with variable

| Type         | Size (bytes)          |
|--------------|-----------------------|
| int          | at least 2, usually 4 |
| char         | 1                     |
| float        | 4                     |
| double       | 8                     |
| short int    | usually 2             |
| unsigned int | at least 2, usually 4 |
| long int     | at least 4, usually 8 |

| Type                   | Size (bytes)                  |
|------------------------|-------------------------------|
| long long int          | at least 8                    |
| unsigned long int      | at least 4                    |
| unsigned long long int | at least 8                    |
| singed char            | 1                             |
| unsigned char          | 1                             |
| long double            | at least 10, usually 12 or 16 |

# Let' go C

## Introduction - Input / Output (I/O)

### Output - printf()

- The one of main output function is `printf()`
- The printf function must be supplied with a format string, followed by any values that are to be inserted into the string during printing

`printf(string, expr 1, expr 2, ...);`

- The format string may contain both ordinary characters and conversion specifications, which begin with the `%` character

```
int p = 3;
printf("The value is %d\n", p);
```

# Let' go C

## Introduction - Input / Output (I/O)

### Output - printf()

- Conversion specifications

| Type         | Specifier |
|--------------|-----------|
| int          | %d, %i    |
| char         | %c        |
| float        | %f        |
| double       | %lf       |
| short int    | %hd       |
| unsigned int | %u        |
| long int     | %ld, %li  |

| Type                   | Size (bytes) |
|------------------------|--------------|
| long long int          | %lld, %lli   |
| unsigned long int      | %lu          |
| unsigned long long int | %llu         |
| signed char            | %c           |
| unsigned char          | %c           |
| long double            | %Lf          |

# Let' go C

## Introduction - Input / Output (I/O)

### Output - printf()

- A conversion specification can have the form %m.pX or %-m.pX, where the m and p are integer constants and X is a letter

```
printf("%10.2f\n", i); // m = 10, p = 2, X = f
```

```
printf("%10f\n", i); // m = 10, p = missing, X = f
```

```
printf("%f\n", i); // m and p are miss, X = f
```

# Let' go C

## Introduction - Input / Output (I/O)

### Output - printf()

- A conversion specification can have the form %m.pX or %-m.pX, where the m and p are integer constants and X is a letter

```
printf("%10.2f\n", i); // m = 10, p = 2, X = f
```

```
printf("%10f\n", i); // m = 10, p = missing, X = f
```

```
printf("%f\n", i); // m and p are miss, X = f
```

- m specifies the minimum number of characters to print

```
int i = 123;
```

```
printf("%10d\n", i); // It will print •••••••• 123
```

- Putting a minus sign in front of m causes left justification

```
printf("%-10d\n", i); // It will print 123 ••••••••
```

# Let' go C

## Introduction - Input / Output (I/O)

### Output - printf()

- p, depending on the choice of X, indicates the minimum number of digits to display (extra zeros are added to the beginning of the number if necessary)

```
int i = 314;
printf("i = %.4d\n", i);

float f = 3.141592;
printf("f = %.4f\n", f);

printf("f = %10.4f\n", f);
```

```
i = 0314
f = 3.1416
f =      3.1416
```



# Let' go C

## Introduction - Input / Output (I/O)

### Input - scanf()

- The one of main output function is `scanf()`
- It can reads formatted input from the standard input, such as keyboards

`scanf(format, exp 1, exp 2, ...);`

- Same as `printf` function, the format string may contain both ordinary characters and conversion specifications, which begin with the `%` character
- In *exp*, the `&` symbol which normally precedes each variable in a `scanf` call is usually but not always required

`scanf("%d,%d", &x, &y);`

# Let' go C

## Introduction - Input / Output (I/O)

### Input - scanf()

- How does scanf work?

```
scanf("%d%d%f%f", &i, &j, &x, &y);
```

```
••1x-20•••.3x•••-4.0e3x  
ssrsrrrsssrrssssrrrrrr (s = skipped; r = read)
```

# Let' go C

## Introduction - Comments

### Comments

- It is a hint added by the programmer for making code easier to read
- Two ways to add comments
  - ✓ `//` - Single line comment
  - ✓ `/*...*/` - Multi-line comments

```
// Define a variable p
int p = 3;
/* Print value of p
   in the screen*/
printf("The value is %d\n", p);
```

# Let' go C

## Introduction - Operators

### Operators

- It is a symbol to operate on a value or a variable
- C has a rich collection of operators to perform various operators, such as
  - ✓ arithmetic operators
  - ✓ relational operators
  - ✓ assignment operators
  - ✓ increment and decrement operators
  - ✓ logical operators
  - ✓ ...

# Let' go C

## Introduction - Operators

|    |                |    |                       |
|----|----------------|----|-----------------------|
| +  | Addition       | >> | Bit right shift       |
| -  | Subtraction    | ++ | Prefix increment      |
| *  | Multiplication | -- | Prefix decrement      |
| /  | Division       | >  | Greater than          |
| %  | Remainder      | >= | Greater than or equal |
| +  | Positive       | <  | Less than             |
| -  | Negative       | <= | Less than or equal    |
| ~  | Complement     | == | Equality              |
| &  | And            | != | Inequality            |
|    | Or             | !  | Not                   |
| ^  | XOR            | && | Logical And           |
| =  | Assignment     |    | Logical Or            |
| << | Bit left shift |    |                       |

# Let' go C

## Introduction - Examples

1. Write a program to add numbers and print the result

```
Please enter first fraction: 5/6  
Please enter second fraction: 3/4  
The sum is 38/24
```

2. Write a program to read a number  $i$  satisfying  $100 \leq i \leq 999$  and then print each digit in one line

```
Please input a number (100-999)  
253  
2  
5  
3
```