

---

# Table of Contents

第零章 Introduction	1.1
第一章 環境與程式編輯器	1.2
第 1-1 節 JDK安裝與環境變數	1.2.1
第 1-2 節 程式編輯器	1.2.2
第 1-3 節 Github版本管理	1.2.3
第二章 基本輸出入	1.3
第2-1節 Java程式架構	1.3.1
第2-2節 基本輸出入	1.3.2
第2-3節 程式錯誤偵測與修復	1.3.3

# Introduction

這是一本透過使用範例循序漸進教授Java程式設計的書

# 環境設定與程式編輯器

## 本章學習目標

- 介紹Java程式開發所需系統環境設定。
- 介紹程式編輯器 Visual Studio Code
- 介紹程式編輯器 IntelliJ Community
- 介紹透過Git進行版本控制的軟體原始碼代管服務GitHub

## 章重點概述

本章主要介紹在Windows作業系統下設置良好的程式編輯與運作環境。此外，我們將介紹好用的程式編輯程式 Visual Studio Code 與 IntelliJ Community。JetBrain 的 Java 程式編輯器除了有良好的程式編寫輔助功能之外，更是Android APP的編輯工具。GitHub是一套良好的軟體原始碼代管服務工具，透過GitHub可以有效的管理軟體在開發過程中的各階段記錄。

## 基本說明

Java 程式是可以跨 Mac、Ubuntu 及 Windows 系列等不同作業系統的一種程式語言，也就是說當你寫好一個程式且完成編譯後即可在各種作業平台上執行，這與傳統的 C 語言、C++... 等是不一樣的。但這裡說到的可跨平台執行，其實是在這些作業系統上都安裝有 Java 的執行環境套件 JRE (Java Run time Environment)。

## 環境設定

首先你可以到 Oracle (甲骨文) 的網站去下載你要用的 JDK (Java Development tool Kit) 來用，在寫本文之時 (2019/07/09) 最新的 Oracle JDK 版本是 [JDK 12](#)。

Windows 10

Linux (Ubuntu 18.10)

Mac

## 程式編輯器

Visual studio code

Eclipse

IntelliJ Community



# JDK 安裝教學

## JDK 介紹

### Java 程式開發所需系統環境設定

要撰寫 Java 程式最重要的就是準備一個可以編譯與執行 Java 程式的環境。因此要能寫一個 Java 程式基本要安裝 Java JDK 及安裝程式編輯器。JDK 是指 Java Development Kit 是編譯 Java 程式的主要工具集。首先使用 Google 查詢 “Java JDK 1.12 download”，連結到下載頁面時會像圖 2-1 一樣必須點擊「Accept License Agreement」才會出現可以下載的連結，各位可以自己使用的作業系統選擇相對應的安裝檔。點擊安裝執行檔時 Windows 作業系統會問你是否要安裝，請點擊「是」（請見圖 2-2）。接著會出現開始安裝的提示（見圖 2-3）。

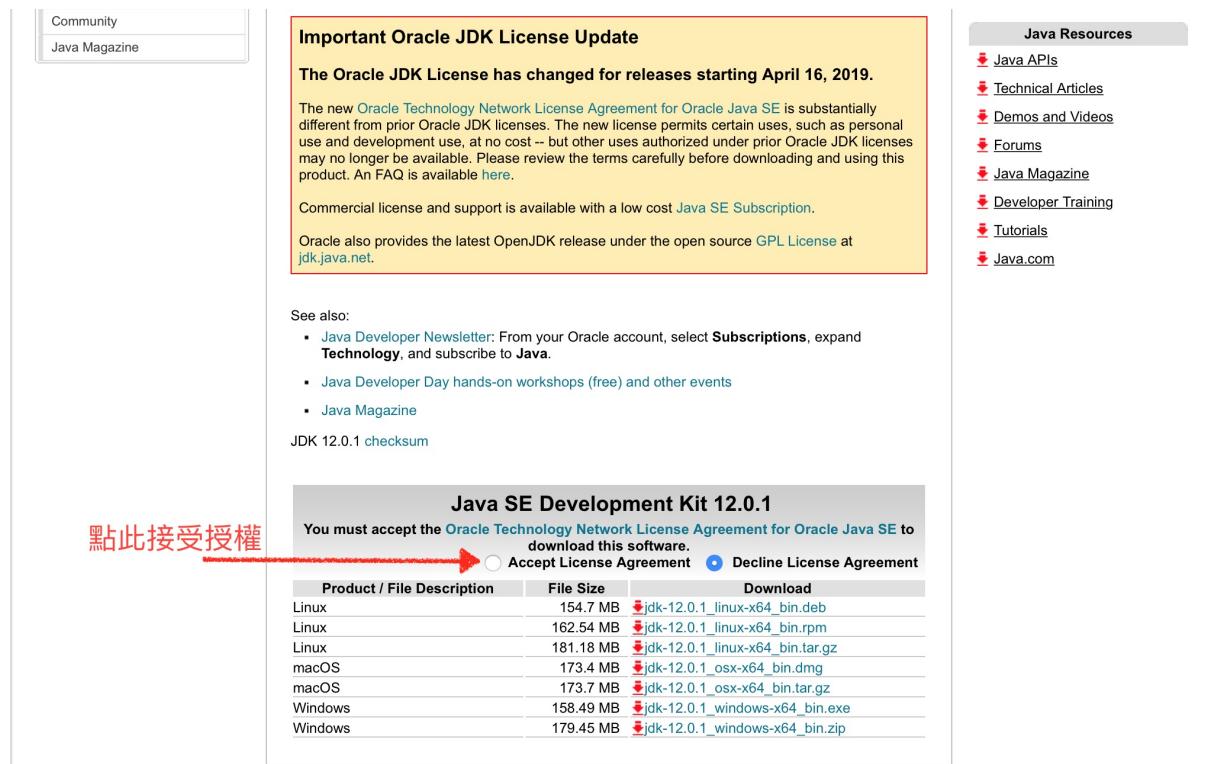


Figure 2-1: 下載 JDK 1.12

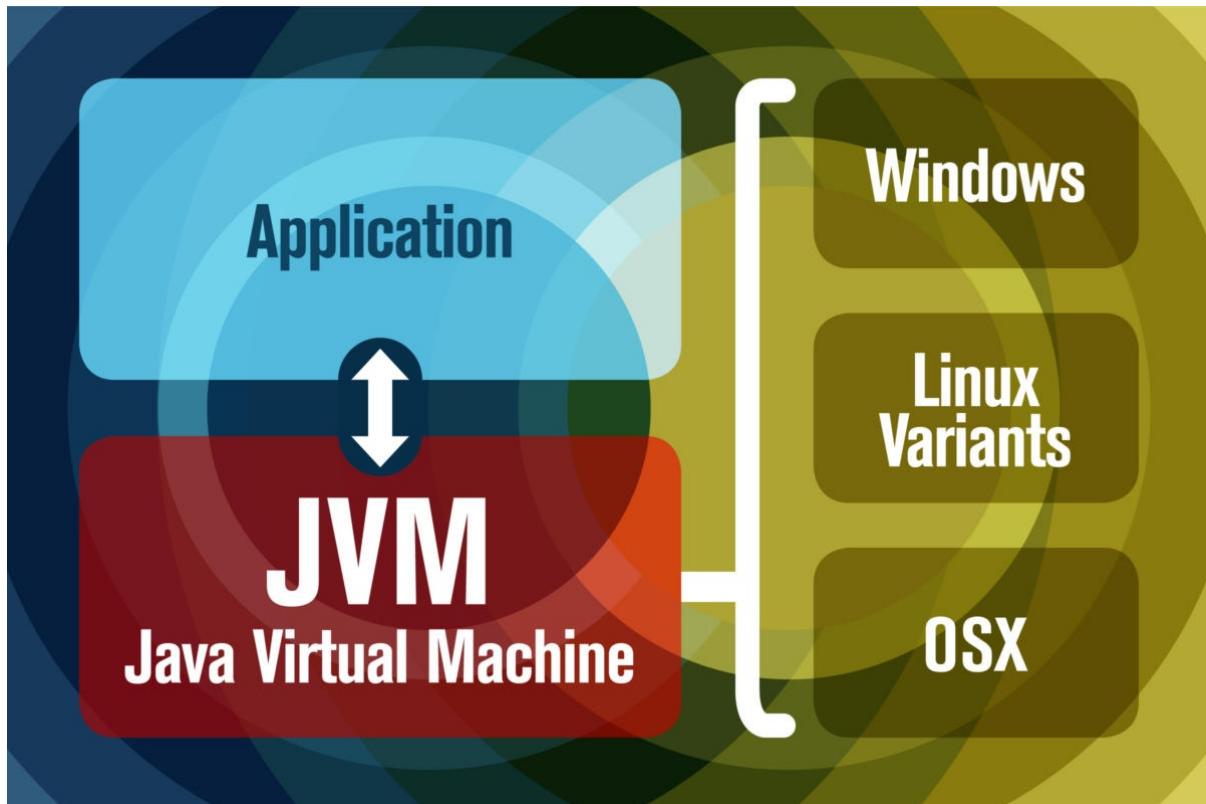


Figure 2-2: High-level view of the JVM (source: <https://www.javaworld.com/article/3272244/what-is-the-jvm-introducing-the-java-virtual-machine.html>)

當然你也可以下載 [Open JDK](#) 下來用。Oracle JDK 與 Open JDK 差別在於，使用 Oracle JDK 在連線 https 時因為沒有解讀憑證的套件，因此沒辦法順利讀取資料，但是 Open JDK 就不會有這樣的問題。[參考資訊 1 \(English\)](#) [參考資訊 2 \(中文\)](#)

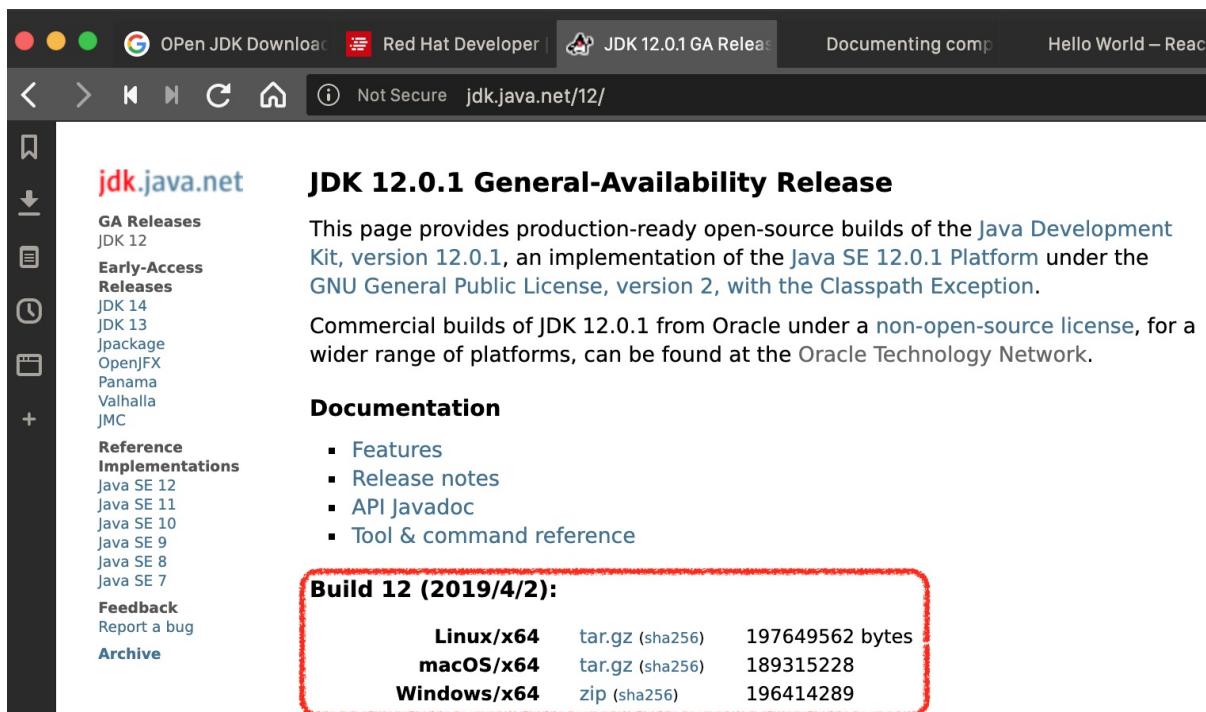


Figure 2-3: Open JDK 12 下載畫面



# 程式編輯器

## Visual Studio Code

Visual Studio Code 是 Microsoft 公司發展的程式編輯器，它可分別運行在 Windows 系列、Linux 系列及 Mac 的作業系統上。

- Step 1: [下載 Visual Studio Code](#)
- Step 2: 選擇編寫 Java 需要用到的 Extension
  - [Java Extension Pack](#)
  - [Language Support for Java\(TM\) by Red Hat](#)
  - [Debugger for Java](#)
  - [Java Test Runner](#)
  - [Maven for Java](#)
  - [Java Dependency Viewer](#)
  - [Visual Studio IntelliCode](#)
- Step 3: 開啟 Terminal 並輸入 `java -version` 看您的 Java 版本是否正確，同時測試 path 設定是否正確。如未完成設定請參考[Windows 安裝 JAVA 及設定環境](#)

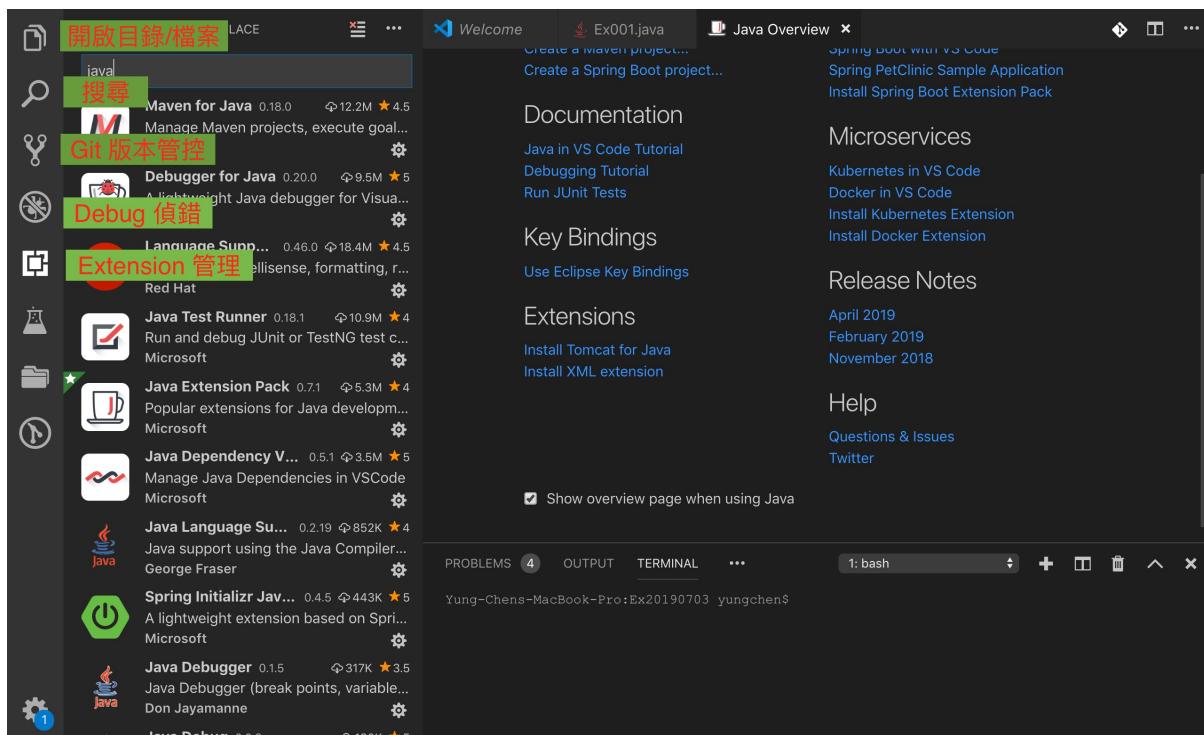


Figure 2-4: Visual Studio Code 介紹

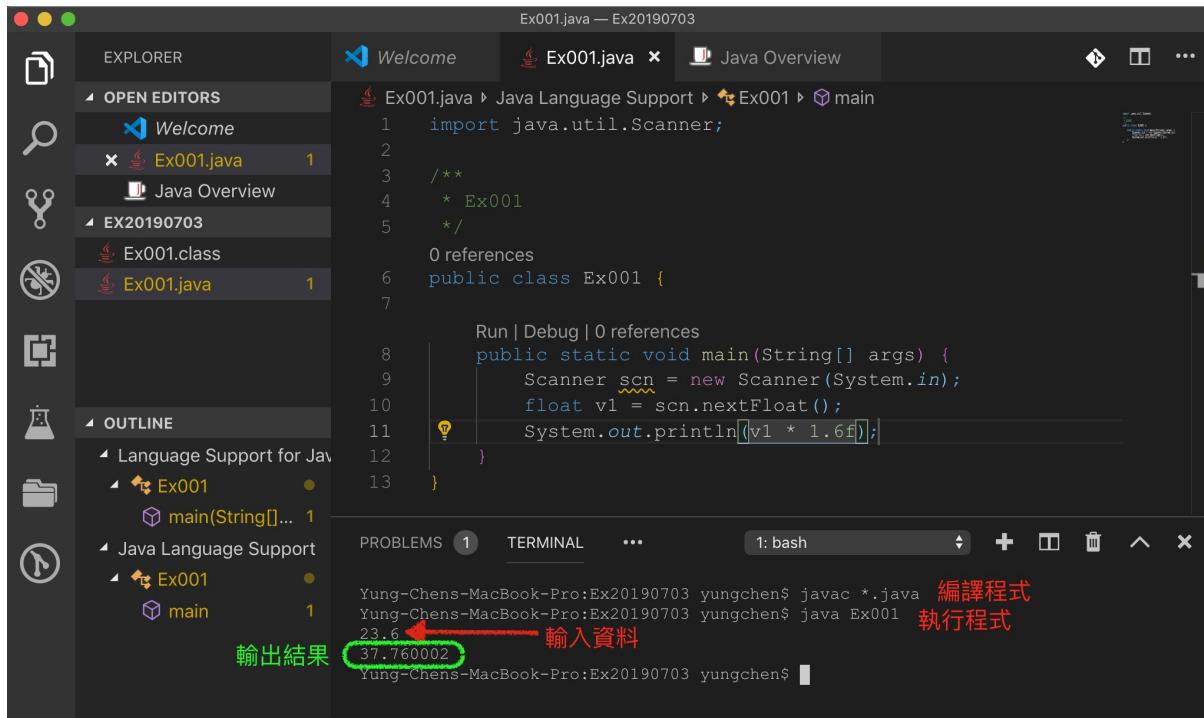


Figure 2-5: 編譯與執行 Java 程式

## Sublime Text 3

## IntelliJ Community

### 參考資料

- [Download Visual Studio Code](#)
- 新一代的編輯器 — VSCode
- [Java in Visual Studio Code](#)
- [Windows安裝JAVA及設定環境](#)

# 基本輸出入

## 本章學習目標

- 介紹Java基本程式結構
- 介紹程式基本輸入與四則運算
- 介紹程式基本輸出

## 章重點概述

本章主要介紹在介紹基本Java結構，並且透過簡單例子介紹Java程式的基本輸出入。

# Java程式架構

範例2-1 是一個秀出 Hello Java! 訊息的簡單程式，程式裡有一個類別及一個程式的啟始點 main() 函數。

## 範例:2-1

程式檔名: Main.java

```

1  /* Program file name: Main.java
2   * Subject: Show "Hello Java!" to user
3   */
4   public class Main { // 類別名字是 Main 必須與程式檔案同
5     public static void main(String args[]) { // main() 函數是程式的進入點
6       System.out.println("Hello Java!"); // 用來列印資料，列印完後會換行
7     }
8 }
```

### 說明

- 每一個 Java 檔中可以存在很多個 class (類別)，但是只能有且必須有一個類別是用 public (公開的)修飾子開頭，而且有public的 class 其名字必須跟檔案名字一模一樣(大小寫也要一樣)。
- 在 Java 裡，程式可以寫得非常龐大，但是只有一個進入點就是 main() 函式。
- 在 main() 函數中使用的參數 String args[] 是用來接收執行程式時使用者給定的參數，資料型態是 String (字串)，並用字串陣列 args[] 存放相關參數。
- // 與 /\* \*/ 是用來做註解用的，也就是當程式的說明用的不會被執行

### 輸出結果

```
Hello Java!
```

範例2-2 是用來測試使用者在執行程式後所帶的參數數量有多少。

## 範例:2-2

程式檔名: Ex2\_2.java

```

1  /* program name: Ex2_2.java
2   * subject: Test running java program with some
3   *           arguments in common line environment
4   */
```

```
5  public class Ex2_2 {  
6      public static void main(String args[]) {  
7          System.out.println(args.length);  
8      }  
9  }
```

## 說明

- 第 7 行: args.length 是用來取得 args 陣列的長度，也就是說可以算出使用者提供了多少個參數。

## 輸出結果

測試 1：執行時不加任何參數

```
java Ex2_1
```

output:

```
0
```

測試 2：執行時加了 3 個參數

```
java Ex2_1 arg1 arg2 arg3
```

output:

```
3
```

範例 2-3 是示範當使用者執行程式時加了兩個整數值當參數，程式去抓取那 2 個參數並進行相加計算，並把結果印出。

## 範例2-3

程式檔名: Ex2\_3.java

```
1  /**  
2   * Program file name: Ex2_3.java  
3   * Subject: run Ex2_3 programing with two integer  
4   * values then print out the sum value  
5   */  
6  public class Ex2_3 {  
7      public static void main(String[] args) {  
8          // 將使用者接在程式後的 第 1 個 參數從 字串型態 (String)轉成 整數 (int) 型態  
9  
10         int v1 = Integer.parseInt(args[0]);  
11         // 將使用者接在程式後的 第 2 個 參數從 字串型態 (String)轉成 整數 (int) 型態  
12  
13         int v2 = Integer.parseInt(args[1]);
```

```
14     // 印出 v1 + v2 的結果  
15     System.out.println(v1 + v2);  
16 }  
17 }
```

## 輸出結果

```
執行方式  
java Ex2_3 4 8  
  
output:  
12
```

## Java程式的基本輸入與輸出

很多時候一個程式的執行需要從使用者那裡取得輸入資料，經過執行後將結果顯示在螢幕上。

### 範例:2-4

程式檔名: Ex2\_4.java

```

1 // Program file name: Ex2_4.java
2 // subject: demo basic Input
3
4 // 引用 java.util.Scanner 函式庫
5 import java.util.Scanner;
6
7 public class Ex2_4 {
8     public static void main(String[] args) {
9         // 告訴一個名字叫做 scn 的掃描器 (Scanner) 物件從系統的基本輸入設備(System.
10 in)
11         // 接收資料，這裡提到的基本輸入設備可以把它想成是 "鍵盤"
12         Scanner scn = new Scanner(System.in);
13         // 顯示提示訊息，讓使用者知道是要輸入什麼
14         System.out.println("請輸入一段文字：");
15         // 透過 scn 物件去接收使用者輸入的整行資料（該行資料中有空白一樣照抓不誤），放到
16
17         // str 的字串變數之中
18         String str = scn.nextLine();
19         // 將字串變數 str 中的資料列印出來
20         System.out.println(str);
21     }
22 }
```

### 執行結果

```

Yung-Chens-MacBook-Pro:ch02 yungchen$ javac *.java
Yung-Chens-MacBook-Pro:ch02 yungchen$ java Ex2_4
請輸入一段文字： ← 提示訊息
亞洲大學 ← 使用者輸入的值
亞洲大學 ← 執行後輸出的結果

```

### 各種輸出格式

## 列印結果不換行

```

1 // Program file name: Ex2_5.java
2 // Subject:
3
4 public class Ex2_5 {
5     public static void main(String[] args) {
6         // 字串資料 str1 的內容是 "亞洲大學"
7         String str1 = "亞洲大學";
8         // 字串資料 str2 的內容是 "資工系"
9         String str2 = "資工系";
10        // 列印 str1 中的資料 不換行
11        System.out.print(str1);
12        // 列印 str2 中的資料 不換行
13        System.out.print(str2);
14        // 列印 str1 中的資料後會換行
15        System.out.println(str1);
16        // 列印 str2 中的資料後會換行
17        System.out.println(str2);
18        // 列印 str1 中的資料後因為加了 "換行符號" \n 所以會換行
19        System.out.print(str1 + "\n");
20        // 列印 str2 的資料後不換行
21        System.out.print(str2);
22    }
23 }
```

### 說明

- 因為 11 與 13 行都是不換行所以第 15 行執行時會接著印出後再換行
- 第19行 加了 \n 的換行符號所以有換行

### 執行結果

```

Yung-Chens-MacBook-Pro:ch02 yungchen$ javac *.java
Yung-Chens-MacBook-Pro:ch02 yungchen$ java Ex2_5
亞洲大學資工系亞洲大學
資工系
亞洲大學
資工系 Yung-Chens-MacBook-Pro:ch02 yungchen$ █
```

## 程式偵錯與修復