# **測試環境設定**

## **測試範例**

撰寫一個簡單的 Java 程式，計算數字的階乘，並觀察 Copilot 如何協助產生程式碼。

**提示：**

Java

// 寫一個方法計算數字的階乘

顯示更多行

**Copilot 產生：** Copilot 會自動產生 factorial 方法與 main 方法，減少撰寫樣板程式碼的時間。透過這種小型程式測試，可確保 Copilot 設定正確，並準備好處理更複雜的任務。

# **基本 Java 程式設計與 AI 協作**

* View(檢視)->Command Palette(命令選擇區)..(按Ctrl + Shift + P)
* `
* Java:Create Java Project → Maven → No Archetype…
* →com.example→ sample2

## 

## **寫簡單 Java 程式**



Copilot 能快速產生常見任務的程式碼，例如計算、字串處理等。以下為判斷質數的範例：

**提示：**

Java

// 寫一個 Java 方法判斷數字是否為質數

Main.java

package com.example;

public class Main {

// 寫一個 Java 方法判斷數字是否為質數

public static boolean isPrime(int n) {

if (n <= 1) return false;

for (int i = 2; i < n; i++) {

if (n % i == 0) return false;

}

return true;

}

public static void main(String[] args) {

int number = 29; // 可以更改這個數字來測試其他值

if (isPrime(number)) {

System.out.println(number + " 是質數");

} else {

System.out.println(number + " 不是質數");

}

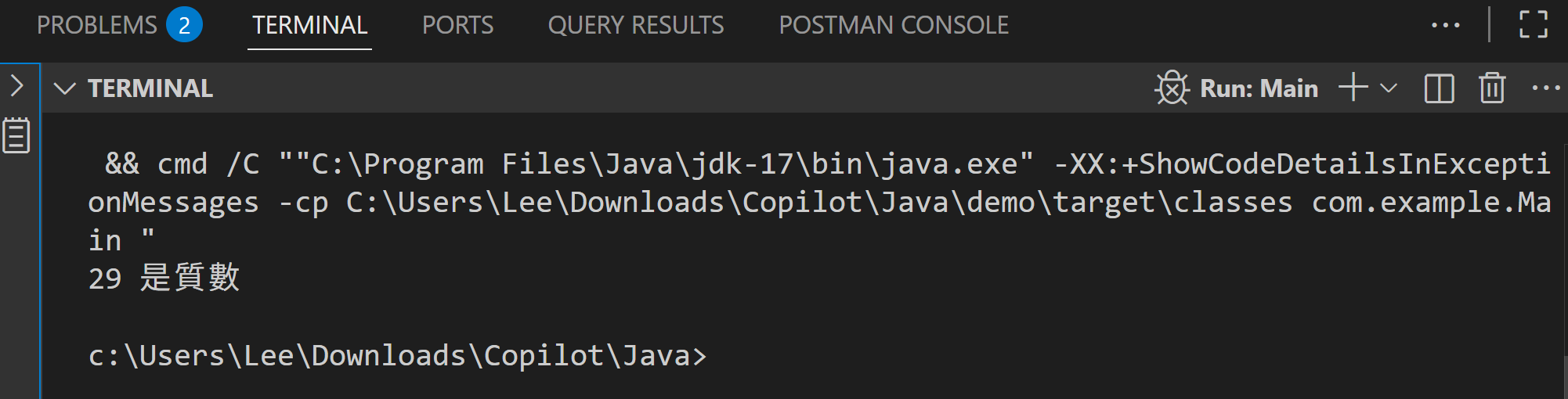
}

}

**Copilot 產生：** Copilot 會產生 isPrime(int n) 方法，並優化效率，例如提前排除 1 與 2、3 的倍數，並使用 6k±1 規則減少不必要的迴圈。main 方法則示範如何呼叫並印出結果。

Main.java 程式碼位置 → 右鍵 →Run Java

## 



# **使用 Java 集合與串流**

## **範例一：過濾使用者年齡**

**提示：**

Java

// 寫一個方法，從 User 物件清單中過濾出符合年齡條件的使用者

// User 類別定義

package com.example;

import java.util.List;

import java.util.stream.Collectors;

public class Main {

// 寫一個方法，從 User 物件清單中過濾出符合年齡條件的使用者

public static List<User> filterUsersByAge(List<User> users, int minAge) {

return users.stream()

.filter(user -> user.getAge() >= minAge)

.collect(Collectors.toList());

}

// User 類別定義

public static class User {

private String name;

private int age;

public User(String name, int age) {

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

}

// 主方法示範如何使用 filterUsersByAge 方法

public static void main(String[] args) {

List<User> users = List.of(

new User("Alice", 30),

new User("Bob", 20),

new User("Charlie", 25)

);

int minAge = 25;

List<User> filteredUsers = filterUsersByAge(users, minAge);

System.out.println("符合年齡條件的使用者：");

for (User user : filteredUsers) {

System.out.println(user.getName() + ", 年齡: " + user.getAge());

}

}

}

**Copilot 產生：**

* 使用 stream() 與 filter()，快速過濾出年齡大於等於指定值的使用者。
* 產生 User 類別與 filterUsersByAge 方法，並在 main 方法示範如何使用。

Main.java 程式碼位置 → 右鍵 →Run Java

## 

## 

## **範例二：產生隨機產品名稱**

**提示：**

Java

// 寫一個方法，產生隨機產品名稱清單

package com.example;

import java.util.List;

import java.util.Random;

import java.util.stream.Collectors;

import java.util.stream.IntStream;

public class Main {

// 寫一個方法，產生隨機產品名稱清單

public static List<String> generateRandomProductNames(int count) {

String[] sampleProducts = {"手機", "電腦", "平板", "耳機", "智慧手錶"};

Random random = new Random();

return IntStream.range(0, count)

.mapToObj(i -> sampleProducts[random.nextInt(sampleProducts.length)])

.collect(Collectors.toList());

}

public static void main(String[] args) {

List<String> productNames = generateRandomProductNames(10);

productNames.forEach(System.out::println);

}

}

**Copilot 產生：**

* 使用形容詞與名詞陣列，隨機組合產生產品名稱。
* 產生 generateRandomProductNames 方法，並在 main 方法示範如何使用。