1-9

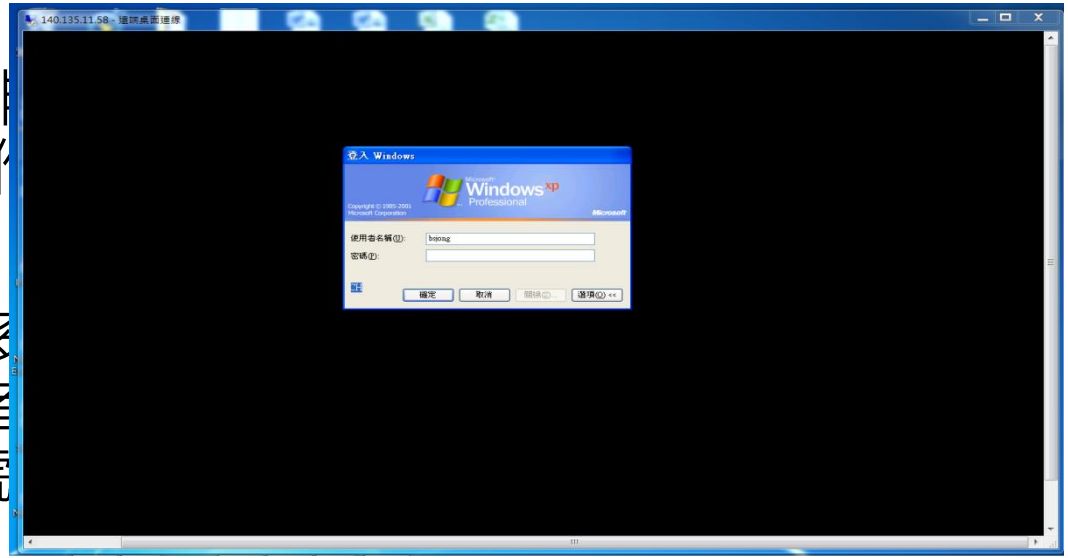
作業系統的保護機制

作業系統的保護機制

- 學習完本單元，您將可以：
  - 了解雙模保護機制及系統呼叫的關係

系統保護（System Protection）

- 防止非經授權者使用系統資源，同時確保不正確的程式不會因此造成其他程式執行不正確。
- 為保護電腦系統不被非
進行登錄(Login)之動作
- 一部電腦可同時提供多
人破壞，並限定使用者
理...等，這些都是帳號



雙模保護 (Dual Mode Protection)

- 利用硬體及軟體設計成使用者模式（User Mode）及監督者模式（Monitor Mode）。
  - 一般使用者登錄後在使用者模式進行作業，使用者模式下不能直接作影響系統核心的事 。
  - 超級使用者（Super User）可以登錄或轉換為監督者模式，以進行系統設定、維護，轉換...等工作。

雙模保護機制及系統呼叫的關係

- 作業系統在監督者模式下執行，在監督者模式內所使用之命令為特權指令（Privileged Instruction）。
- 暫停系統（Halt System）、打開 / 關閉中斷（Turn On/Off Interrupt）、轉換模式（Change Mode）、輸出/輸入指令...等均為特權指令。
- 在使用者模式下，程式執行任何動用系統資源之動作，均以系統呼叫(System Call)藉由監督者模式代為處理。

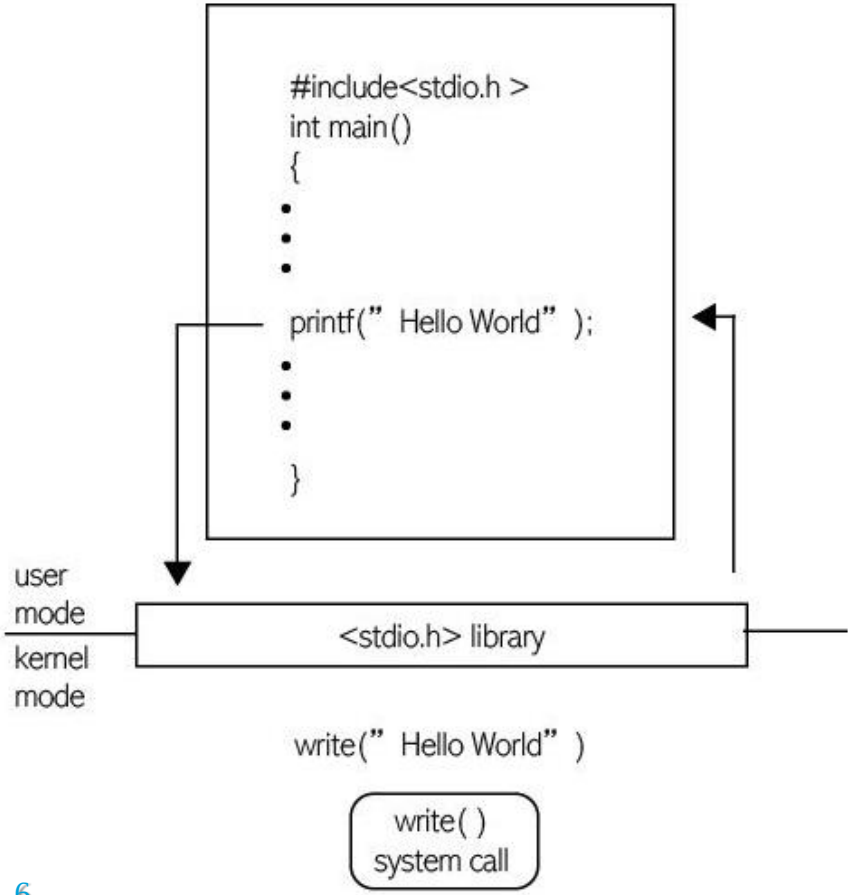
系統呼叫（System Call）(1)

- 系統呼叫是程式設計者所撰寫之程式與作業系統的介面。
- 作業系統必須對系統資源作保護，任何使用系統資源之程式必須透過系統呼叫，由作業系統決定並安排使用資源。
- 使用者經由外殼命令或應用程式呼叫系統程式庫（System Library），再由系統呼叫要求作業系統核心執行，當工作是牽涉到週邊設備時，作業系統核心再呼叫設備驅動程式（Device Driver），並由設備驅動程式控制週邊設備進行輸出/輸入之工作。

系統呼叫之系統程式庫（System Library）

	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Protection	SetFileSecurity() InitlializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

系統呼叫（System Call）(2)



- 輸出敘述(Statement)被編譯成呼叫系統函數，而系統函數本身又被轉譯成系統呼叫。

應用程式介面（Application Program Interface)

- 應用程式介面（Application Program Interface) 簡稱為API，可以視為比系統呼叫更高階的函數。
- 一個API函數的呼叫，可以看作執行多個連續有意義的系統呼叫。
- 三種常用的API：
  - Win32 API (Windows)
  - POSIX API (UNIX, Linux, and Mac OS X)
  - Java API (Java Virtual Machine)