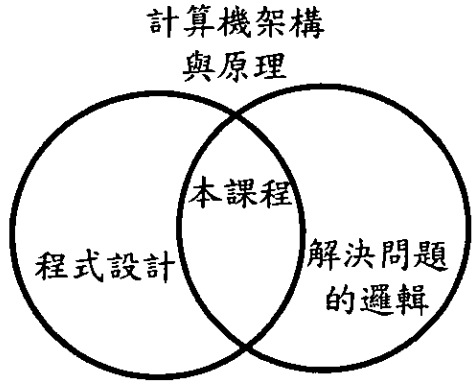
柒、 核心科目課程

7.1 計算機概論課程

計算機概論課程是大一入門課程，主要介紹計算機運作之基本觀念與基礎知識，並訓練學生能運用所學程式語言撰寫程式以解決一些簡單問題，若程式撰寫的基本功夫沒練好，將影響學生學習其他課程之學習興趣，故可依學生程度而調整上課時數，且在課堂檢討學生常犯錯誤，使其不害怕寫程式。本章首先敘述課程設計理念，進而敘述課程大綱。

7.1.1 課程設計理念



教育目標：

1. 訓練學生瞭解計算機的基本運作原理與如何設計程式以解決問題。
2. 訓練學生熟悉兩種程式語言(包含一種傳統的循序式語言與物件導向式程式語言）的結構及使用，及運用物件導向觀念設計程式。
3. 訓練學生具備程式偵錯與測試能力。

課程內容：

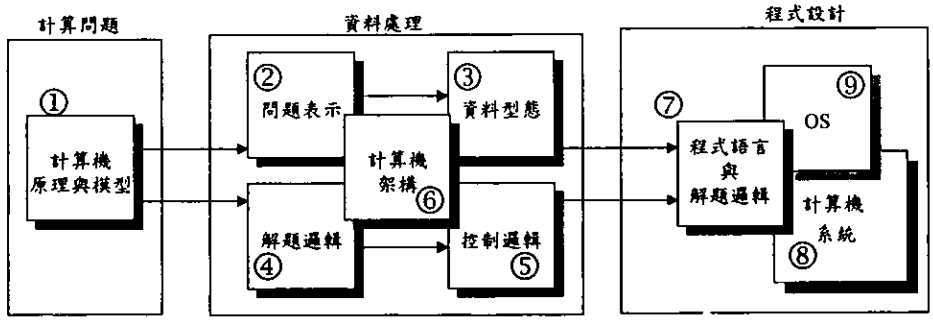
1. 電腦基本概念：硬體基本概念，作業系統，重要系統架構演進、多工、多緒、多處理器概念，執行環境概念，電腦網路。
2. 程式語言結構：Names, Scopes 及 Bindings，控制流程，資料型態，副程序及 Control Abstraction，建構可執行之程式。
3. 物件導向觀念：軟體元件、類別與物件、屬性、操作、方法、訊息、封裝、繼承。
4. 應用程式設計能力訓練。

設計構想與進行方式：

1. 課程的進行由淺顯易懂的計算機原理，配合實務操作，並經由程式語言的引導，讓學員能從解決問題的程式語言中，交互驗證計算機的運作。希望微觀(micro view)的從計算機資料處理，到宏觀(macro view)的problem-solving 的過程，來介紹計算機、軟體程式、人與計算的關係。
2. 課程分成理論授課與實務操作兩部分，必須搭配一起進行，同時給予學生適度的作業練習與程式撰寫，以驗證所學
3. 適度介紹程式偵錯(debugging)的技巧與管理，並需要每週至少一小時與學生一起分析解沃程式設計的錯誤與困難，並於適當時間實施期中期末測驗，以驗證學習成效。

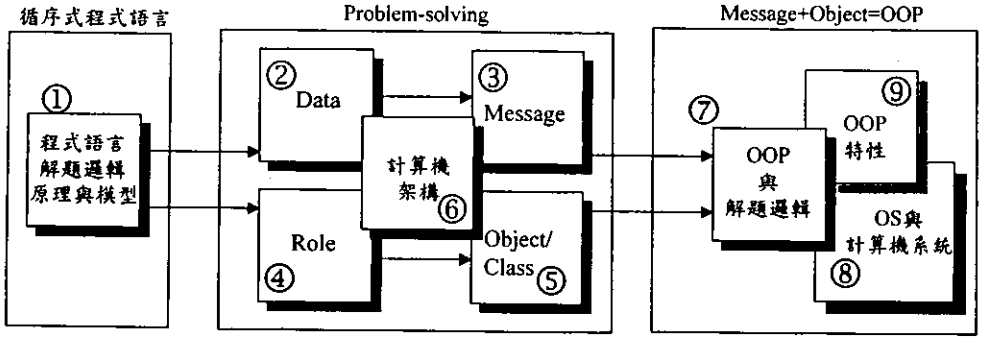
課程分成下列兩部分，分於上下學期實施：

- PART-I: (1) 從人類解決問題的需求出發，強調「計算(computing)」為解決問題的基本元素，並藉此介紹計算機的各部功能。(2) 解決問題的資料處理方式介紹問題的表示，並在計算機架構下對應資料型態。(3)以資料處理的解題邏輯，介紹計算機的處理方式(如內儲程式計算機架構)，並介紹計算機的控制邏輯功能。(4)介紹(2)及(3)的同時，需同步以程式語言的資料表示方式與控制邏輯加以對照式驗證，依照資料處理由小而大，由簡而繁的順序介紹程式如何進行資料的處理。(5)然後，由程式設計的角度看計算機四大元件的運作原理，讓學生知道在處理資料的過程中，程式如何在計算機中被處理（檔案，OS，網路等）。使學員能夠以一種循序式程式語言自行撰寫簡單的程式。概念如下圖。



- PART-II: (1) 從 PART-I 中介紹的第一種循序式程式語言的限制與問題解決(problem solving & modeling)的角度，來(2)介紹解決問題不是以程式為思考的核心，而是以角色(role)與訊息(message)溝通的方式來進行。(3)由解決問題時的問題描述與解題邏輯對應至物件與訊息，在計算機系統的架構下，逐步導出物件導向式程式語言的基本原理。(4)課程引用同一個例子貫穿整個物件導向程式設計的介紹：由物件(object)概念逐漸介紹到類別(class)，經由實例說明繼承(inheritance)及資料封裝(encapsulation)等物件式程式語言的重要特色。(5)以 OOP 的特色由簡而繁說明簡單物件與物件間，至複雜物

件間的繼承與交互合作的關係。同時，適度介紹其在計算機中運作方式與循序式程式的異同。概念如下圖。



分述如下

- PART-I: 從人類解決問題的需求出發，介紹計算機的沿革，強調「計算(computing)」為解決問題的基本元素，並藉此介紹計算機的各部功能。然後，由程式設計的角度看計算機四大元件的運作原理。從簡單的數字計算，到資料龐大而複雜的問題，引導學員逐步瞭解循序式程式語言（如 VB, C 等）的發展與特色，更重要的是讓學員知道在處理資料的過程中，程式如何在計算機中被處理（檔案，OS，網路等）。使學員能夠以一種循序式程式語言自行撰寫簡單的程式。課程設計共分為分成 14 個單元，共 80 小時，每週上課 5 小時:

Unit 1: 序論

- 簡介人類對計算機的需求並介紹計算機擅長於處理大量、重複計算的工作。介紹電腦的誕生及演變過程，並說明為何各種問題需要轉換成計算問題。各種電腦應用系統之硬體、軟體、分類（超級電腦、迷你電腦、個人電腦、PC/Notebook/PDA）之 簡介。同時需搭配動手認識電腦的實習，目的在於引發學生對計算機的興趣，並給予梗概式的認識。

Unit 2: 從人 problem-solving 的需求看計算機

- 點出計算(computing)與人 problem-solving 的關係。介紹各種不同資料形式與其運算操作的種類，並進一步說明操作型態與計算機器的設計需求。

Unit 3: 電腦/計算機系統

- 說明電腦/計算機系統由硬體、軟體、系統、人所組成。計算機四大功能運算、儲存、輸出、輸入的功能與之間的關係。同時以此概念式的方塊圖說明計算問題如何被處理。

Unit 4: 計算問題的表示方式

- 簡介表示問題的數字系統與數碼表示方式，說明 digit/bit/byte 與各種數字系統及轉換方式。介紹數字以外的問題表示方式，如符號系統表示，聲音與影像的表示。強調問題必須表示成數字型態的問題(呼應計算機四大功能& the black box)才能被計算機處理。

Unit 5: 計算機『運算』概念

- 延續前一單元說明資料如何被計算機處理的需求、概念、與演進。開始說明何謂『程式』與程式的基本元素(程式基本方塊)，並以范紐曼(Von Neumann) 『內儲程式』概念計算機作為說明的基礎。同時搭配一種程式語言作為對照式的驗證與練習，給予基本計算數字型態的資料處理指令的練習，讓學生對「計算程式」有基本認識。

Unit 6: 計算機『儲存資料』概念

- 說明在 Unit 4 所表示出來的資料如何儲存在計算機中。說明計算過程對儲存體的需求與概念，包含資料儲存與程式的儲存。資料儲存部分介紹各種 Data types 在記憶體的位置與表示方式；程式的儲存部分介紹程式在記憶體中的位置與讀取到執行的過程。同時程式語言的練習，將各種不同資料型態表示、處理並儲存在程式中，作為對照式的驗證與練習，讓學生對「資料儲存」、「資料處理」與程式間的關係有基本認識。

Unit 7: 計算機的『控制單元、輸入、輸出』

- 介紹計算機的控制單元、輸入、輸出等功能。同時藉由控制的概念說明處理資料時，程式如何進行輸入輸出的動作。程式部分則介紹簡單的程式指令，布林代數運算及 IF...THEN...ELSE 的程式控制流程，並以此進行 I/O 的練習。

Unit 8: 計算機資料處理—I (資料處理流程)

- 介紹計算機進行資料處理的流程，強調資料處理是計算機的主要功能，並說明資料處理的型態與 problem-solving 的類型間的關係。藉此介紹進階的資料表示方式如 Array/Queue/Stack 等類型與原。再者，說明此種資料結構的適用時機與 problem-solving 對應各種控制結構 control structures(For, While, Do, Switch, etc.)。旨在讓學生瞭解計算機程式處理資料時對資料與程式控制間的關係。

Unit 9: 計算機資料處理—II (模組化程式設計)

- 延續 Unit 8 的資料處理觀念，說明將經常用到的資料處理步驟進行模組化的原因、優點與種類。介紹模組化程式設計的觀念，變數或參數的使用範圍與在記憶體中的關係。讓學生將前的程式加以模組化，並呼叫系統提供的現有的函數或副程式，藉由程式的練習，比較模組化程式的優點與特色。

Unit 10: 整合性描述

- 描述電腦的五大單元如何共同完成（程式的）計算工作，需對應至

程式與資料在五大單元的關連性，計算機提供的 BIOS+ROM+RAM 等硬體支援的提供。藉此讓學生從頭思索 Unit-1~9 所述計算機、程式、資料間的互動關係，並進行較大型的練習問題。

Unit 11: 資料處理—III (大量資料的處理)

- 延續 Unit 8 的資料處理觀念，說明計算機處理大量資料的時機與方式。藉著檔案系統的介紹讓學生瞭解檔案類型、檔案變換、檔案操作（建立、新增、刪除、修改等）與程式的關係。

Unit 12: 作業系統概念

- 介紹程式設計師應瞭解的作業系統概念與所提供的服務與資源。從對計算機資源的概念談程式對 OS 服務的需求，如 IO, file systems 等等。從程式的生命週期（開啟、執行、等待、保護）介紹單一程式與 OS 的關係，同時回顧計算機的五大功能中經由 OS 保護或控制的部分。

Unit 13: 作業系統與程式的互動

- 進一步從介紹程式設計師應瞭解的多工，多緒，多處理器的概念與此種環境下作業系統概念與所提供的服務與資源。並由此介紹重要系統架構演進(如中介軟體，組譯器與編譯器，遠端呼叫,主從式架構計算環境與各種 OS 的種類與適用時機)
- 開始讓學生進行整合式的程式練習。

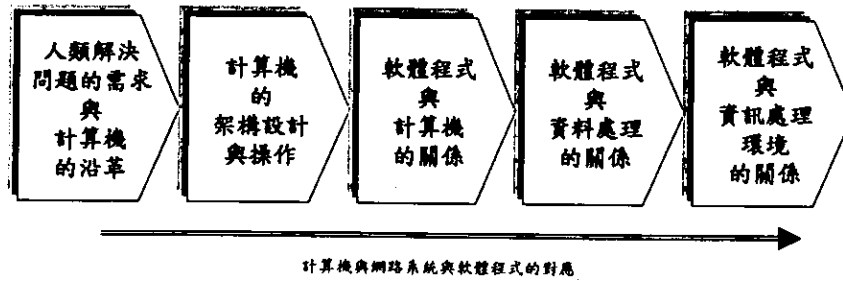
Unit 14: 電腦網路與網際網路

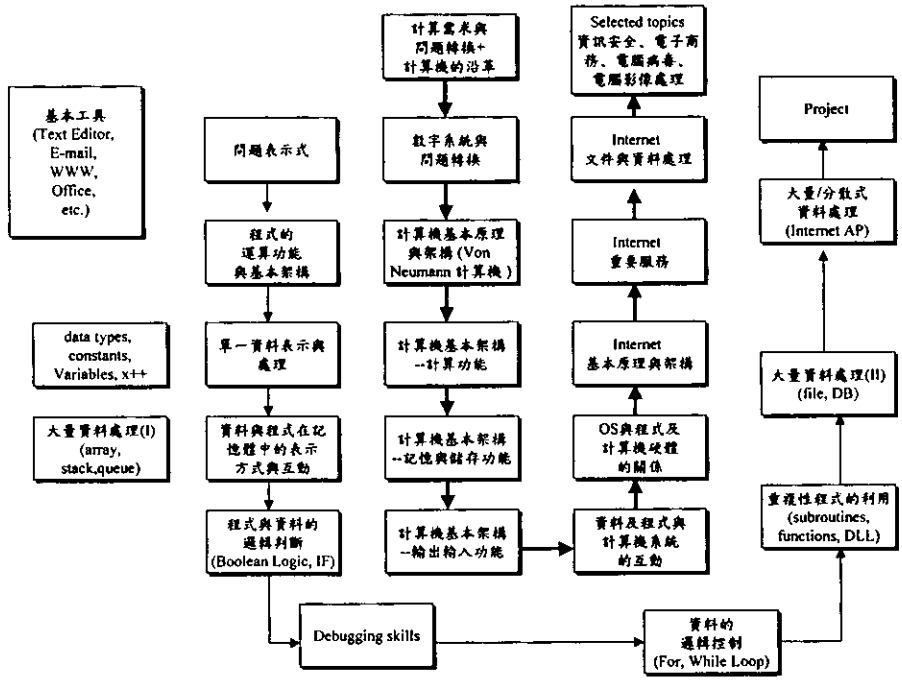
- 從「資源共享」的角度談對網路的需求與概念。經由共同約定式的通訊協定到標準化約定的通訊協定談 protocol 與網路類型（LAN/MAN/WAN/Wireless）、網際網路的沿革、Internet 的發展歷史及台灣 Internet 的沿革、IP/router，Domain name、標記語言(markup language)如 HTML 與 XML 的概念與各種網際網路的應用與重要服務。實習部分則讓學生練習以現有軟體連上 Internet，並實際架設製作網頁（站），並利用目前所學的程式技術(Java Applet, VB Script, Java Script)結合現有程式模組或函式庫撰寫簡單的 Internet 應用小程式。

Unit 15: 資訊安全與電腦應用的未來趨勢

- 介紹資訊安全與電腦應用的未來趨勢。

各單元進行的順序的關係如下圖所示：





- PART-II: 從 PART-I 中介紹的第一種循序式程式語言的限制與問題解決(problem solving & modeling)的角度，來介紹解決問題不是以程式為思考的核心，而是以角色(role)與訊息(message)溝通的方式來進行。逐步導出物件導向式程式語言的基本原理。課程引用同一個例子貫穿整個物件導向程式設計的介紹：比較分析其利用循序式程式語言解題時的限制，由物件(object)概念逐漸介紹到類別(class)，經由實例說明繼承(inheritance)及資料封裝(encapsulation)等物件式程式語言的重要特色。同時，適度介紹其在計算機中運作方式與循序式程式的異同。本課程僅介紹基本而重要的 OOP 技術，進階部分則在「軟體系統發展」課程中補充介紹。課程設計共分為分成 14 個單元，共 80 小時，每週上課 5 小時(上課學生必須先修 PART-I 課程)：

Unit 1: 以傳統程式語言發展軟體系統的限制

- 說明傳統程式語言發展軟體系統的限制，以導出物件導向式程式語言的需求。需以一個由傳統程式語言發展的軟體程式為例，說明其在設計、開發、維護、上的缺點與限制。可以讓學生用以前開發的程式，分析說明相對於本單元的限制。

Unit 2: Objects + Messages=OOP

- 介紹角色扮演(role-playing)式的問題解決策略，說明解決問題的角色與其所具有的屬性(attribute)，對應 Data structure+algorithm=program 的觀念，說明 Objects + Messages= OOP 的重要性，而物件(object)對應角色是解決問題的基本單元。用一個較生活化的實例說明本單元的內容的重要觀念。

Unit 3: From Role Model/Object to CLASS

- 從物件的多元性與從屬關係導出類別(class)的觀念，並以實例說明

類別的定義與組成元素。以 Unit-2 的實例說明物件與類別的關係。

Unit 4: Problem Solving with Objects and Classes

- 說明以 OOP 方式解決問題時的步驟，並對應 Unit-1 的傳統程式語言發展軟體系統的限制，以說明 OOP 的優點。

Unit 5: Programming with Objects

- 開始說明以 OBJECT 為程式設計的基本單元，與物件導向程式的組成元素。說明如何建置一個 OOP 的程式，以 Unit-2 的實例說明如何在程式中定義並使用一個物件。再者，說明物件封裝屬性與繼承（完全相同屬性）的基本觀念。

Unit 6: Inheritance I—the relationship between Object and Class

- 介紹「繼承」的觀念與處理相同/差異的 properties，只談到屬性完全相同的屬性（垂直繼承+不衝突的屬性），以 Unit-2 的實例說明如何在程式中定義並使用物件的繼承關係。

Unit 7: Summary –I

- 對應 Unit-1 的傳統程式語言發展軟體系統的限制，以說明 OOP 的優點。

Unit 8: Life-cycle of Objects

- 說明軟體物件的生命週期與資料範圍(scope)，用以說明物件將資料隱藏起來的處理方式，與其在計算機系統如何被處理（物件起始化、執行、結束、多執行緒等重要觀念）。以 Unit-2 的實例說明程式中的物件的生命週期。

Unit 9: Use Package – I & Integration I

- 使用系統提供的軟體物件進行程式的輸出入(Window-based GUI, file I/O, etc.)。以 Unit-2 的實例說明如何在程式中加入 GUI 的物件。同時可以讓學生以至目前為止的 OOP 技術，設計一個較大型的作業。

Unit 10: Inter-Access of Properties among Objects

- 談物件屬性的公開與私密性（private/public），說明物件之間橫向與縱向的屬性呼叫與對應的方法(methods)，以 Unit-2 的實例說明程式中的物件如何使用父系物件的屬性，與其他類別的屬性。

Unit 11: Inheritance II—the relationship between Object and Class

- 說明屬性繼承發生衝突時的處理方式，介紹 subclass/superclass 的定義與使用，其相關的程式技巧。以 Unit-2 的實例說明程式中的物件增加與父系物件不同或衝突的屬性。

Unit 12: Abstract Class v.s. Normal Class

- 介紹概念式物件的意義與使用時機。說明介面物件與多重繼承的觀念。以 Unit-2 的實例說明程式中的物件如何使用概念式物件。

Unit 13: Use Package II --GUI & Container

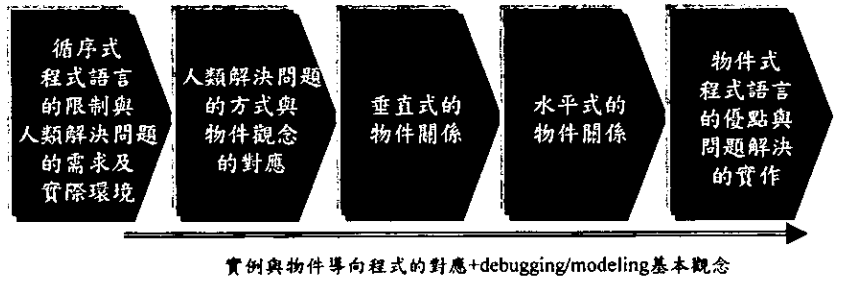
- 使用更多系統提供的軟體物件(.io, .util, .net)。以 Unit-2 的實例說明如何在程式中加入 GUI 與網路功能的物件。說明 event delegation model, container 等觀念同時可以讓學生以至目前為止的 OOP 技

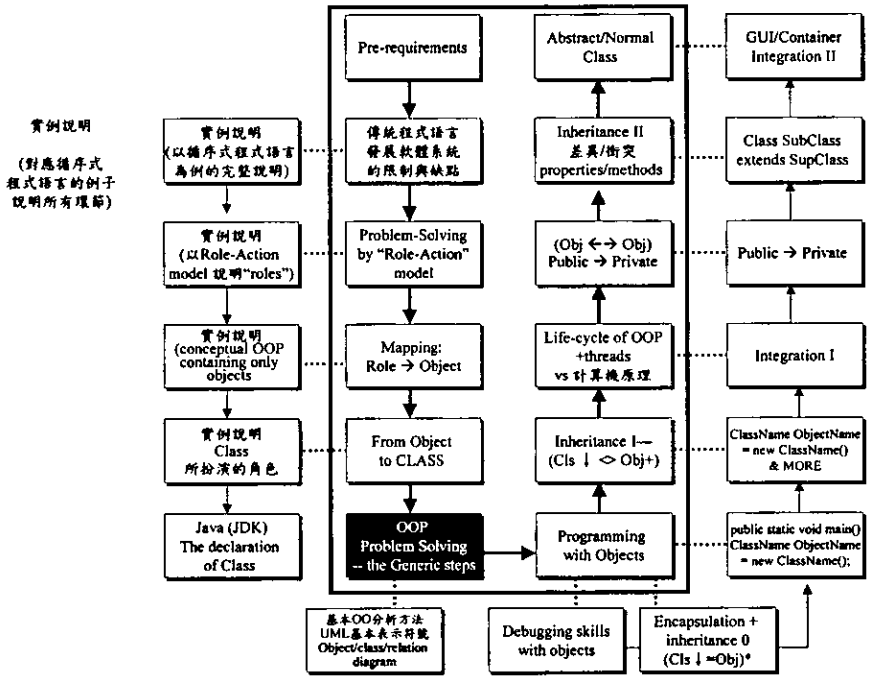
術，設計一個大型的 project。

Unit 14: Summary II

- 對應 Unit-1 的傳統程式語言發展軟體系統的限制，以說明 OOP 的優點。

各單元進行的順序的關係如下圖所示：





7.1.2 課程大綱

計算機概論（一）課程大綱

Core Knowledge	授課時數(Hr)	Programming Skills+LAB	實習時數(Hr)
1. 序論 ■ 對計算機的需求—人、資料(大量、重複計算) ■ 電腦的誕生及演變過程 ◆ 齒輪與碼表的模型 ◆ 速度(Hz)與散熱問題 ■ 問題的轉換—(各種問題→計算問題) ■ 硬體、軟體、電腦的分類 ◆ 超級電腦、迷你電腦、個人電腦、PC/Notebook/PDA ■ 各種電腦應用系統 ■ 簡介與其他	3	● 基本工具認識安裝(e.g., text editor) ● LAB: 各種應用軟體 ● LAB: 各種電腦外觀、種類、型態、廠牌... ● LAB: PC DIY**	3
2. 從人 problem-solving 的需求看計算機 ■ 點出計算(computing)與人 problem-solving 的關係 ■ The needs of representation and operations of data on machines ■ The requirements of such a "machine"	1	● LAB:各種軟體程式 ■ 系統程式 ■ 應用程式 ■ BIOS/firmware	
3. 電腦/計算機系統--硬體、軟體、系統、人 ■ 計算機四大功能 ■ 運算、儲存、輸出、輸入(只談外觀、種類、型態、廠牌...) ■ A Black Box Model – Computer ■ 問題的轉換—(各種問題→計算問題)	2	● LAB: Windows 小算盤 ● LAB: UltrEditor: 各種資料型態的內部表示法 ● LAB: UltrEditor: 程式執行檔的內部表示法	
4. 數字系統與數碼表示方式 ■ digit/bit/byte + base 2/8/16/32 ■ 數字系統轉換 ■ 符號系統表示 ■ 聲音與影像的表示 ■ 計算機運算模型---問題必須表示成數字型態的問題 (呼應計算機四大功能& the black box)	4	● Homework Assignment:	

5.	計算機『運算』概念 ■ 需求與概念 ■ 計算機的操作-->開關-->1000110-->程式(不談語言演進) ■ 何謂『程式』? ◆ 程式的基本元素(程式基本方塊) ■ 計算機運算模型--『內儲程式』概念計算機 ◆ 需求 ◆ 范紐曼(Von Neumann) 『內儲程式』概念計算機 ◆ 何謂「程式」 ◆ 『內儲程式』概念計算機架構與運算原理 ◆ 此架構下的各種工作 ■ CPU/register (program counter)	3	● Programming-00 (建議用 VB/Java) ■ program 基本結構 ■ commands; statements ■ sequential execution ■ compiler ■ Ex: simple commands, e.g., printout 3+5 (numeric +-*÷only) (暫不談 control commands) ● Homework Assignment:	3
6.	計算機『儲存』概念 ■ 需求與概念 ■ 資料儲存 ◆ 沒有記憶體的計算機 CPU 如何運作 ◆ 記憶體—physical CKT/memory space/logic meaning ◆ Data types in memory ■ 程式儲存 ◆ 程式在記憶體中的位置 ◆ 讀取程式 ■ 程式執行—interaction among CPU/program/memory/data ■ CPU/register/memory	2	● Programming-01 ■ More commands; statements ■ data types ■ data/constants/variables/program in memory space ■ the use/call of data in programs ■ Ex: simple data operation, e.g., printout 3+5, "Red"+"Apple", "a"→ "A" (暫不談 control commands) ● Debugging Skills in Programming with dumping memory for viewing constants and variables--I (配合記憶體與程式執行的觀念) ● Homework Assignment:	3
7.	計算機『控制單元、輸入、輸出』概念 ■ 需求與概念 ■ 程式如何輸入輸出資料 ◆ 控制單元 ◆ 輸入—執行的步驟 ◆ 輸出—執行的步驟 ■ 資料處理—0 (資料的表示與輸出入處理)	2	● Programming-02 ■ Simple calculation & variable assignment (y=x+1; x=x+1) ■ 布林代數運算元與運算子 ■ 布林代數運算 in VB (>=< IF…THEN…ELSE) ■ Simple I/O control (暫不談其他 control commands) ● Debugging Skills in Programming with dumping memory for viewing constants and variables+check points--II (配合記憶體與程式執行的觀念) ● Homework Assignment:	6
8.	資料處理—I (資料處理結構與流程)	6	● Programming-03:大量資料的處理	3

● 資料處理是計算機的主要功能 ● 資料處理的型態 v.s. problem-solving ■ 處理結構性資料的方式 ◆ Array/Queue/Stack ◆ 類型與原理 ◆ 適用時機＋比較		I ■ Array/Queue/Stack 類型與原理 ● Debugging Skills in Programming with array/queue/stack ● Homework Assignments	
9. 資料處理—I (資料處理結構與流程) ■ problem-solving 對應各種 control structures ■ 程式的 control 結構 ◆ control structure 的種類 ◆ control structure 的應用時機 ■ control structure 的原理	6	● Programming-03:大量資料的處理 I ■ 程式的 control 結構 ■ control structure 的種類 ■ control structure 的應用時機 ■ control structure 的原理 ■ 以 do loop 處理資料 ◆ 在 Memory 如何動作 ◆ For/While in do loop ■ 整合性練習 ● Debugging Skills in Programming with control structures ● Homework Assignment:	3
10. 資料處理—II (模組化程式設計) ■ what/when 模組化程式設計 ■ why no in a single program? ■ subroutine 的觀念 ■ subroutine 的種類 ■ call-by-value/reference/address ■ modulized programming ■ scope of program/variable ■ 主從程式間的關係（對應 program counter/memory space）	6	● Programming-04:大量資料的處理 II ■ Procedure/function 的類型 ■ 將前面的程式碼加以模組化 # ● Debugging Skills in Programming with modulized structures ● Homework Assignment:	2
11. 整合性描述 ■ 電腦的五大單元如何完成（程式的）計算工作? ■ 對應至 code/data/extra/stack segment 觀念 ■ BIOS+ROM+RAM, etc.: relationships and block diagrams ■ 對應 unit-3 的觀念	1		0
12. 資料處理—III (大量資料的處理) ■ what/when 大量資料的處理 ■ why no in a single program with many constants and variables? ■ 檔案類型 ■ File Handle/File System	2	● Programming-05:大量資料的處理 III ■ IO by Files ■ File handle (creation/deletion/ modification) ■ Data to/from files ■ File 的轉換 by programs ● Debugging Skills in Programming with files ● Homework Assignment:	2

13. 作業系統概念 ■ 需求與概念 ■ computational resources (分配與控制) ■ 再談何謂「程式」與 program counter ■ services provided by OS ◆ IO, file systems, etc. ■ 單一 CPU 系統與程式 ■ creation/execution/protection/waiting/termination of a program ■ 單一程式與 OS 的關係 ◆ program 的生命週期(對應 unit-9 modulized programs) ◆ flow-char/control sequence ◆ 程式執行環境設定 ■ 系統軟體	2	● LAB: Windows (task manager/ MyComputer/ server/ monitor) ● Programming-06 ■ Use services provided by OS (e.g., printing, file systems, sound, video, etc.) ■ Windows API/DLL ● Homework Assignment:整合型練習—using array/stack/queue + files + sub-routines + OS services to take care different size of data ● Integrated project (given by some existing algorithms)	2
14. 作業系統與程式的互動 ■ 多工, 多緒, 多處理器概念與程式間的關係 ■ Program vs OS (多重程式與多工多處理器 OS 的關係, program 的生命週期 ■ 重要系統架構演進 ◆ 中介軟體, 組譯器與編譯器 vs 作業系統 ◆ 遠端呼叫 ◆ 主從式架構計算環境與程式執行 ◆ OS 的種類與適用時機 (Windows/Unix/Linux/Mac OS, etc)	2		
15. 電腦網路與網際網路 ■ 需求與概念 ■ 網路的興起--網路的需求+資源共享 ■ Protocol 概述—共同約定式→標準化約定 ■ 網路類型 --LAN/MAN/WAN/Wireless ■ 網際網路的沿革&Internet 的發展歷史 ■ ARPAnet，NSFNET，台灣 INTERNET 的沿革，TANet, TANet2 ■ Internet 上的身分識別 ◆ IP/router，Domain name	6	● LAB: Access to the Internet ● LAB: 網路的架設實務 ■ File server/Printer Server ■ 網路線/網路卡 ■ HUB/Modem/ADSL ■ 在 Windows 的相關設定 ● LAB: ■ 建立 Homepage ■ 作業系統環境之建立與設定 ■ 區域網路與網際網路設定 ■ 伺服器管理 ● Homework Assignment: ■ Programming-07: Access Internet with Java Applet (Java Script/VB Script) ■ Programming-08: connecting	4

■ Internet 上的文件 ◆ The concept of markup language ◆ HTTP/HTML ◆ XML—concept & applications ■ Internet 上的重要服務與伺服器管理與重要功能介紹 ◆ firewall/telnet/ftp/SSH/RPC/ email/news/proxy/search engine/ICQ/E-mail/ ftp/BBS/Plugins/其他		to the Internet (optional:: Connect to Internet or Intranet, Access to file server/printer server (using VB/Java components))	
16. 資訊安全與電腦應用的未來趨勢 ■ 資訊安全 ■ 電腦應用的未來趨勢	1		0
時數建議—四學分五小時（16 週共 80 小時）	49		31

計算機概論（二）課程大綱

Core Knowledge	授課時數(Hr)	Programming Skills+LAB	實習時數(Hr)
0. Pre-requirements ● Procedure-based language 基本觀念與實作 ● Program running on a computer (CPU, Memory, I/O, OS) ● Von Neumann's Computer Architecture ● 程式語言--review ■ 機器語言/組合語言/高階語言 ■ 4GL/自然語言 ■ Compiler/Interpreter ■ Programming Language – Definition, Usage, Declaration	0	● Step-by-Step statement; ● Constant & Variable ● Data type ● Data type transformation 未來可對應到 ◆ data type →(Class) ◆ variable →(Object) ◆ DTT→(Polymophism)* ● Statement (Std-in and Std-Out) ● Control Structures ◆ If else ◆ Do-loop (For/While) ● Array/Queue/Stack ● Procedures/Functions (by reference and by value) ● File (FILESTREAM) ● Java environment* ■ JDK、javac、java(virtual machine)、path environment setup、Java Code Convections	0
1. 以傳統程式語言發展軟體系統的限制 ● One Big Main File ● Main File + Functions/Procedures ● Reusable codes ● Complete and complex ALGORITHM design ● Cooperative development ● Data Structure + algorithm = Program ● 缺點 ● Program management ● Debugging ● Reusability ● Development Hour/Man ● Hard to design complete ALGORITHM ● The needs of programming languages	3	以實例說明之 Ex. ● A simple Lotto/Workflow program implemented by traditional languages. (需搭配完整的程式實例) ● Homework Assignment:	2
2. Objects + Messages=OOP ● Problem-solving by roles, not only by algorithms ● Modeling the roles by objects	6	以人類的問題解決方式與環境說明之 ● 分析 Lotto/Workflow problem (不需談程式)+其中的 role/object	3

● Role playing by "attributes" ● The mission of roles = handling data ● Role actions(intra-/inter-roles) = program ● The Role Model = Object ● Problem-solving by OOP = ● 只有 Object 是不夠的, An object is an instance of Class		● the role model of object in problem solving ● A "conceptual" OOP containing only objects	
3. From Role Model/Object to CLASS ■ Various objects ■ Class, why? ■ The definition of Class ■ The component of Class ■ Data members (properties) ■ Member functions (methods)	6	I: 以人類的思考邏輯說明之 ● 分析 Lotto/Workflow program 中的 Class (不需談程式) II: In Java (JDK) ● The declaration of Class ● Lotto/Workflow Class in Java	3
4. Problem Solving with Objects and Classes ■ Generic steps ■ 對應 Unit-1 Class/Object 的優點是什麼?	2	● 問題轉換 ● Lotto/Workflow problem→OOP 的步驟 (不需談程式)	1
5. Programming with Objects ■ The definition of Application ■ The declaration of Application ■ The definition of Object ■ The construction of Object ■ Simple program in object/class ■ Encapsulation + property inheritance 0 ● Object 封裝屬性，方法的特性 ● 基本繼承的觀念	6	以傳統程式語言的觀點說明對 inheritance 的需求 I: 以人類的思考邏輯說明之 ● 分析 Lotto/Workflow program 中的 Class/Object (不需談程式) II: In Java (JDK) ● The entrance of program execution ● public static void main() ● ClassName ObjectName= new ClassName(); ● Ex: Handling Lotto/Workflow Objects in Java ● Debugging Skills in Programming with objects --I (配合記憶體與程式執行的觀念) ● Homework Assignment:	2
6. Inheritance I—the relationship between Object and Class ■ What/Why inheritance ■ 繼承什麼東西--definition ■ 處理相同/差異的 properties ■ 只談到屬性完全相同的屬性垂直繼承+不衝突的屬性	5	I.以人類的思考邏輯說明之 分析 Lotto/Workflow program 中 Class/Object 的 inheritance 關係 (不需談程式) II: In Java (JDK) Object/Class Inheritance in the Lotto/Workflow problems using Java Debugging Skills in Programming with object inheritance --I (配合記憶體與程式執行的觀念) Homework Assignment:	2
7. Summary I -- OOP 對應傳統循序式	1		0

語言的優勢			
8. Life-cycle of Objects ● The Life-Cycle & Life-Scope of Object ● Data driven/Event driven ■ The invoking of objects by data/events ■ Initialization, Survival, Finalization ● The scope of objects ■ Data hiding ■ Access control of properties ■ Access control of methods ● The control of objects in a view of OS ■ Objects in memory spaces ■ Data/message passing among objects ● Threads（the needs and basic conceptual model）for managing objects in OS ● 對應 Von Neumann 計算機架構 ● OOP 的程式架構 （對應 unit I-12）	6	I.以計算機的運作角度說明之 Data/Even driven in the Lotto/Workflow problems(不需談程式) 說明 class/object/property 在計算機的四大元件+OS 的概念模型（不需談程式) II: In Java (JDK) ■ Constructor ■ this&this() ■ Destructor ■ Exercises ■ Debugging Skills in Programming with object (配合記憶體與程式執行的觀念) ■ Homework Assignment:	2
9. Use Package – I ● File Access using Java Packages ■ java.io	3	■ File objects, FileStreamReader, FileStreamWriter ■ GUI version "Hello, World" ■ 將 Lotto/Workflow 的問題資料儲存起來 Integration I ■ 完整的 JAVA 作業/project ■ Learning Management & Summarization of Bugs	2
10. Inter-Access of Properties among Objects ● Public → Private ● Private/Public properties ■ Access "Properties" ■ Private/Public--definitions ■ The access of Object's properties ■ The access of Object's methods	6	I. 以人類的思考邏輯說明之 ■ 分析 Lotto/Workflow program 中 Class/Object 之間(垂直＋水平) private/public 屬性關係(不需談程式) II: In Java (JDK) ■ (public v.s private, not including protected) ■ setProperties & getProperties	1
11. Inheritance II—the relationship between Object and Class ● 處理差異/衝突的 properties/methods ● The definition of subclass and superclass ● The declaration of subclass ● The component of Subclass	6	I. 以人類的思考邏輯說明之 ■ 分析 Lotto/Workflow program 中 Class/Object 的 inheritance 關係(不需談程式) II: In Java (JDK) ■ Class SubClassName extends SupClassName {...}	1

■ Data members(properties) ■ Member functions (methods) ■ The inherited data members ■ The inherited member functions		■ Inheritance in the Lotto/Workflow problems using Java ■ Exercises	
12. Abstract Class v.s Normal Class ● Definition ● Interface ● Multiple Inheritance	3	I. 以人類的思考邏輯說明之 分析 Lotto/Workflow program 中需要 Abstract Class 的時機 (不需談程式) II: In Java (JDK) Shape(abstract class) v.s. Triangle, Circle, Square (concrete class)	1
13. Use Package II --GUI & Container ● Using Java Packages ■ java.io, java.util, java.net, etc. ■ Swing I ● Event delegation model ● GUI Components ■ Containers, etc. ■ Swing II	3	GUI/Internet version of the Lotto/Workflow problems	1
14. Summary II -- OOP 對應傳統循序式語言的優勢	2	Integration II ■ 完整的 JAVA 作業/project ■ Learning Management & Summarization of Bugs	1
時數建議—每週 5 小時（共 80 小時）	58		22