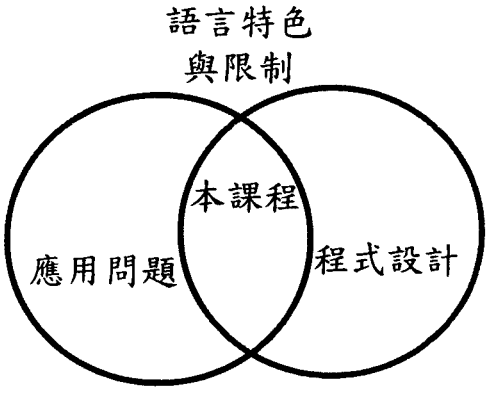
7.5 程式語言

7.5.1 課程設計理念

前言：學生已熟悉數種程式語言，且可運用這些語言設計程式，唯欲善用程式語言設計優質程式，實有必要程式語言的由來、類別及各自的弱點，進一步可自我學習新的程式語言，比較此新語言與其他語言之差異，使能依問題應用領域之不同而選擇適當語言。



教育目標：

1. 使學生了解程式語言之類別及應用範圍
2. 使學生了解各類程式語言演進的緣由，特性，及弱點
3. 訓練學生具有自我學習新程式語言的能力

課程內容：

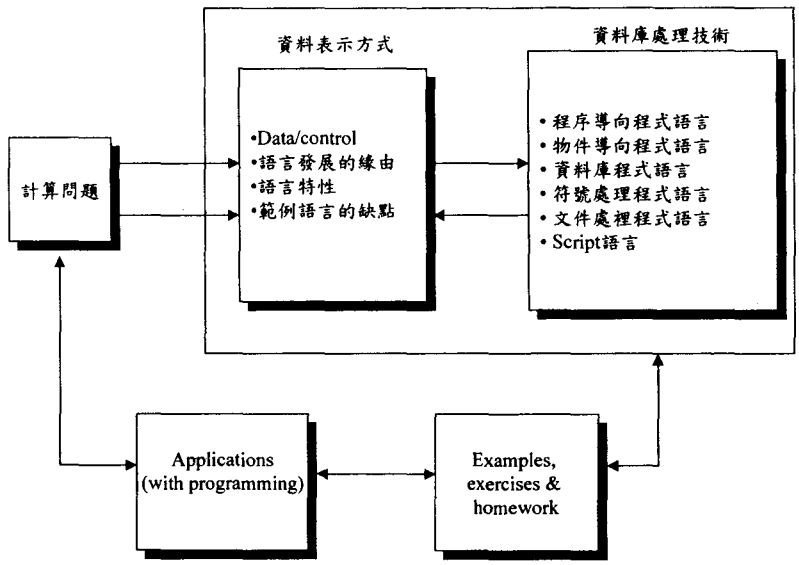
1. 程序導向程式語言
2. 物件導向程式語言
3. 資料庫程式語言
4. 符號處理程式語言
5. 文件處裡程式語言
6. Script語言

設計構想與進行方式：

課程的進行依語言特性分成：程序導向程式語言；物件導向程式語言；資料庫程式語言；符號處理程式語言；文件處裡程式語言；Script語言。每一類語言依其發展之時間由教師理論授課，而每一語言課程授課的流程：

1. 先強調語言發展的緣由：目的讓學生了解影響該程式語言演進之主要因素，如應用之影響、程式結構之影響、計算機架構之影響、程式撰寫的方便性之影響、軟體設計模式之影響。
2. 語言特性：目的讓學生了解該程式語言的一些重要性質。
3. 舉一範例：以實例說明，讓學生榮一體會與了解。

4. 語言的缺點：說明該程式語言之缺點及其使用注意事項。



本課程共分成 7 個主題，共 48 小時，各主題說明如下：

單元一、程式語言之目的，分類與演進(3hr)

1. 導論
2. 程式語言的目的
3. 程式語言的類別、性質與主要應用
4. 程序導向程式語言
5. 物件導向程式語言
6. 資料庫程式語言
7. 符號處理程式語言
8. 文件處裡程式語言
9. Script語言

單元二、程序導向語言(Procedure-Oriented Language)（16hr）

1. 背景—機器語言與組合語言
2. 機器語言
3. Von Neumann Machine
4. 組合語言
5. FORTRAN
6. COBOL
7. BASIC
8. PASCAL
9. C
10. Ada
11. VB

單元三、物件導向語言（5hr）

1. small talk
2. C++
3. Jave

單元四、資料庫語言（5hr）

1. SQL, SQL1
2. SQL2
3. SQL3

單元五、符號處理程式語言（4hr）

1. LISP
2. Prolog

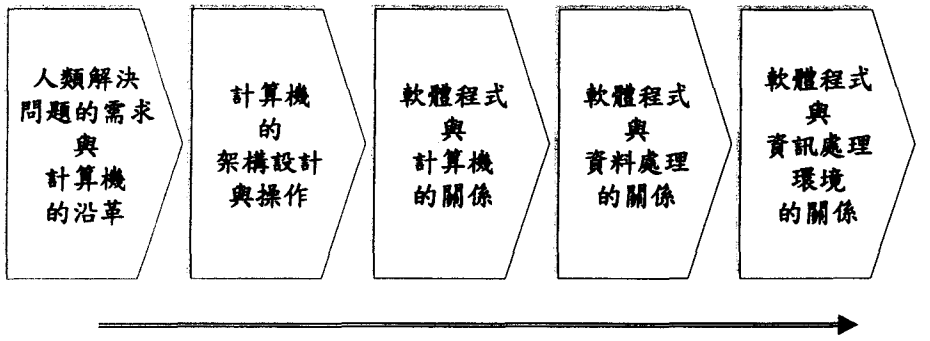
單元六、文件處裡程式語言（7hr）

1. Tex
2. Postscript
3. Latex
4. PDF

單元七、Script語言（8hr）

1. Perl
2. Tcl/tk
3. Python
4. Ruby
5. PHP

各單元進行的順序的關係如下圖所示：



計算機與網路系統與軟體程式的對應

7.5.2 課程大綱

Core Knowledge	Programming skills/LAB	時數
單元一、程式語言之目的，分類與演進 2. 導論 ■ 程式語言的目的 ■ 程式語言的類別、性質與主要應用 ◆ 程序導向程式語言 ◆ 物件導向程式語言 ◆ 資料庫程式語言 ◆ 符號處理程式語言 ◆ 文件處裡程式語言 ◆ Script 語言 ■ 語言的執行 ◆ compiler ◆ assembler 3. 影響程式語言演進之主要因素 ■ 應用 ■ 程式結構 ■ 計算機架構 ■ 程式撰寫的方便性 ■ 軟體設計模式 4. 程式語言的發展史		3
單元二、程序導向語言(Procedure-Oriented Language) 1. 背景—機器語言與組合語言(1956) ■ 機器語言(1 hr) ■ Von Neumann Machine ■ 動機 ◆ ■ 語言特性 ◆	● 用 JAVA 語言來模擬 FORTRAN 的 Static storage ● 用 JAVA 語言來模擬 COBOL 的報表輸出 ● 用 JAVA 語言來模擬 PASCAL 的 link list ● 用 JAVA 語言來模擬 C 的 low level control ● 用 JAVA 語言來模擬 Ada 的	16

■ 範例 ■ weakness ◆ 2. 組合語言(1956) ■ 動機 ◆ 提高程式可讀性、Relocatable Program Concept ■ 語言特性 ◆ Operation Code、Register 名稱用符號代替 ◆ 變數名稱取代位址 ◆ 巨集指令(Macro) ■ 範例 ■ weakness ◆ Relocatable Loader 之導入 — 程式起始位置之變更 3. FORTRAN(1957) ■ 動機 ◆ 近似人常用的自然語言— 第一個高階語言 ◆ Formula Translation ■ 特性 ◆ Declaration statements, execution statements ◆ Simple variables, array variables ◆ Assignment statements, control statements, I/O statements, subroutine ◆ Statements, FORMAT statements ◆ Static storage ◆ Automatic type conversion ◆ Library ◆ Comment ■ 範例 ■ Weakness ◆ Static storage ◆ No recursive feature ◆ FORMAT statements	concurrent	

- ◆ Difficult to handle input/output

4. COBOL (1959)
   - ■ 動機
     - ◆ Business data processing
     - ◆ Report, numeric
     - ◆ Fortran 無法處理 string, file
   - ■ 特性
     - ◆ 程式看起來很像是英文的口語
     - ◆ manipulate input/output file
     - ◆ divides programs into: identification, environment, data, procedure divisions.
     - ◆ 提供 records 的資料結構
   - ■ 範例
   - ■ Weakness
     - ◆ 無結構化

5. BASIC(1964)
   - ■ 動機
     - ◆ Students easy to learning
     - ◆ Fortran, COBOL 變數一定要宣告
     - ◆ Fortran, COBOL 語言 compiler 如有錯很難除錯
   - ■ 特性
     - ◆ Var. cannot be declared
     - ◆ Interpreter
     - ◆ Friendly error messages
     - ◆ GoSub…Return
     - ◆ Respond fast for small programs
   - ■ 範例
   - ■ Weakness
     - ◆ 無副程式參數傳遞的觀念
     - ◆ Interpreter (slow: 每次執行須重新編譯)

6. PASCAL (1970)
   - ■ 動機

◆ Fortran, COBOL, BASIC 語言無結構化概念 ◆ 之前的語言無 dynamic (pointer), record 的資料結構 ■ 特性 ◆ Variables must be defined ◆ 提供 dynamic (pointer)的資料結構 ◆ well-structured ◆ 提供 Sets, files, records 的資料結構 ■ 範例 ■ Weakness ◆ One pass compiler--procedures 要依順序寫(被呼叫要寫在前) 7. C (1971) ■ 動機 ◆ 之前的語言採 one pass compiler ◆ A general-purpose programming language ◆ For UNIX ◆ 之前的語言少提供 library routines ■ 特性 ◆ access to low level hardware ◆ library routines ◆ UNIX system ◆ weak type check ■ 範例 ■ Weakness ◆ flexible but dangerous 8. Ada (1970) ■ 動機 ◆ 美國防部所制定 ◆ 之前的語言無 concurrent 的能力 ◆ 之前的語言無結構化概念 ◆ 支援 embedded and real-time systems ■ 特性 ◆ concurrent (parallel)		

◆ semaphores ◆ bounded buffer ◆ structured ◆ run-time checking ■ 範例 ■ Weakness 9. VB (1991) ■ 動機 ◆ 之前的語言處理視窗介面難 ◆ 之前的語言與資料庫之間難連結 ■ 特性 ◆ rapid development of GUI for Windows ◆ complex database object library ◆ network-savvy ◆ multithreaded ■ 範例 ■ Weakness ◆ 程式碼大		
單元三、— 物件導向語言 1. Small talk (1969) ■ 動機 ◆ object-oriented ◆ 跨平台 ■ 特性 ◆ first OOP ◆ Everything is an object ◆ Everything is available for modification ◆ Types are dynamic ◆ Bytecodes ◆ Multithreaded ◆ Concurrency ◆ 繼承（不支援多重繼承） ■ 範例	● Java Applet	5

- Weakness
  - garbage collection 不完整
  - Bytecodes--執行速度慢

2. C++ (1983)

- 動機
  - C + O.O.
- 特性
  - Compatible with C
  - 多重繼承的觀念
  - Strict compile-time check
  - Template
  - Exception handling
- 範例
- Weakness
  - Single-threaded
  - 無 garbage collection
  - 多重繼承的觀念-- 一個物件的兩個父物件可能有衝突

3. JAVA (1995)

- 動機
  - independent of the host platform
  - for networking
  - securely
- 特性
  - portable
  - bytecodes
  - 完整的 garbage collection
  - network-savvy
  - multithreaded
  - Class library
- 範例
- Weakness
  - Bytecodes--執行速度慢

單元四、— 資料庫語言 ■ SQL(1970), SQL1(1989) ■ 動機 ◆ 針對關聯式資料庫發展的語言 ■ 特性 ◆ Data definition in SQL-- table definition, schema definition and update, user defined domain, relational constraints, relational catalogues ◆ SQL queries -- selection, insertion, deletion, update ■ 範例 ■ Weakness ◆ 缺少 Changing security settings ◆ syntax rules, create queries 較不方便 ■ SQL2(1992) ■ 動機 ◆ Enhance SOL1 ■ 特性 ◆ new reserved words, wildcard ◆ GRANT, REVOKE ◆ DISTINCT ◆ LIMIT TO ■ 範例 ■ Weakness ◆ 無法處理 Multimedia ■ SQL3(1999) ■ 動機 ◆ applications 的需求改變--multimedia ■ 特性 ◆ for objects-oriented ◆ for data-mining ◆ for spatial-data	●	5

◆ for temporal-data ◆ for on line analytical ◆ for data-warehousing ◆ for multimedia-data ◆ 將程式區分成兩部 – 1. core specification: for RDBMS, 2. option specialized packages: for applications ■ 範例 ■ Weakness		
單元五、—符號處理程式語言 ■ LISP(1958) ■ 動機 ◆ 人工智慧計算的需求 ■ 特性 ◆ rule-based ◆ lumda-calculus & functional computation ◆ list as data and program ◆ dynamic data typing (late binding) ◆ symbolic computation ◆ focused on problem solving strategies ◆ suitable for AI applications (lower level) ◆ fast prototyping ◆ stack-based interpretation ■ 範例 ■ Weakness ◆ some data is not suitable to be described as lists or functions ◆ procedure-like computation ◆ still too –lower for AI applications ◆ slow interpretation (compared with C) ◆ large memory consumption ■ Prolog(1970) ■ 動機	●	4

◆ 單一一種程式計算方式（logic interpretation）： 以 logic clauses 處理所有運算並表示所有資料型態 ■ 特性 ◆ data=program; logic clauses as data and program ◆ dynamic data typing (late binding) ◆ symbolic computation ◆ focused on problem solving strategies ◆ suitable for AI applications (higher level) ◆ fast prototyping ◆ Interpretation ■ 範例 ■ Weakness ◆ slow interpretation (compared with C) ◆ huge memory consumption		
單元六、—文件處裡程式語言 ■ Tex(1970) ■ 動機 ◆ 製作排版文件，處理數學公式與各種符號。 ■ 特性 ◆ 制訂排版指令 ◆ 以寫程式方式進行排版與文件製作（文件＝程式） ◆ 編譯式語言 ◆ 分離文件製作與輸出 (tex， dvi， ps) tex→dvi→ps→... ◆ 可跨平台 ■ 範例 ■ Weakness ◆ 不容易記憶各種排版指令 ◆ 需要進行編譯(compilation) ◆ 需安裝字型與相關函式庫		7

- Postscript(1976)
  - 動機
    - 於各種終端機顯示文件，由各種印表機輸出文件。增加文件的可攜性
  - 特性
    - 制訂螢幕輸出與印表機輸出指令
    - 分離文件製作與輸出
    - 提供各種跨硬體的文件可攜性
    - interpretation
    - 提供多種轉換成 PS 格式的工具
  - 範例
  - Weakness
    - 文件檔案龐大
    - 需安裝字型與相關函式庫

- Latex(1985)
  - 動機
    - tex 指令太多，不容易使用。LaTeX 於 tex 中加入巨集，並提供各種可重複套用的版形。
  - 特性
    - 同 tex，但加入巨集，並提供各種可重複套用的版形
    - 使用者需記憶的指令較少
  - 範例
  - Weakness
    - （仍然）不容易記憶各種排版指令
    - 需要進行編譯(compilation)
    - 需安裝字型與相關函式庫

- PDF
  - 動機
    - PS 型態文件檔案太大，不利於流通於網路上。PDF 提升網路流通的可攜性文件。
  - 特性
    - 同 PS

◆ 文件較精簡，SIZE 較小 ◆ 可嵌入 browser ◆ 提供多種轉換成 PDF 格式的工具 ■ 範例 ■ Weakness ◆ 需安裝字型與 drivers		
單元七、—Script 語言 1. Perl(1987) ■ 動機 ◆ 對問題中的文字字串進行快速、複雜的文字處理。Perl 是目前 Script Language 中最熱門的語言，由於一開始結合了 C， awk，sed， sh， and BASIC，等語言的特性，因此在 Script Language 引起十分廣泛的回響，特別是在 Unix-like 的作業系統上，傳統的網管與系統管理的工作，八成幾乎主要使用 Perl 語言來完成，另外 Perl 社群亦建立 Comprehensive Perl Archive Network (CPAN).來統一管裡各個模組，因此 Perl 成為支援性最好的 Script Language. 然而 Perl 亦繼承了 sed/awk 的負擔，整個語言的設計十分不易閱讀，程式設計師，往往一個月後，無法看懂之前所寫的程式，換言之，程式的可擴充性相對較不易。 ■ 特性 ◆ regular expression ◆ Perl takes features from other languages，such as C， awk， sed， sh， and BASIC，among others. ◆ Perls database integration interface (DBI) supports third-party databases including Oracle， Sybase， Postgres， MySQL and others. ◆ Perl works with HTML， XML， and other mark-up languages.	●	8

- ◆ Perl supports both procedural and object-oriented programming.
- ◆ Perl interfaces with external C/C++ libraries through XS or SWIG.

■ 範例

■ Weakness

- ◆ 執行效率較差
- ◆ Readable is much poor!

2. Tcl/tk(1988)

■ 動機

- ◆ 對電腦各種（周邊）設備進行處理，給予程式設計師以單一程式語言處理所有電腦相關設備。

■ 特性

- ◆ EDA/CAD: Tcl has become the defacto standard in EDA and CAD applications.
- ◆ Test Automation: Tcl is the de facto standard for automated testing.
- ◆ Dynamic Web Content: Some of the industry's most popular high-traffic web sites are Tcl-powered.
- ◆ Network and System Management: Tcl provides a platform for network and system management applications with its rapid GUI development， easy extensibility and cross-platform support.

■ 範例

■ Weakness

- ◆ The main advantage of Tcl is that *all* its variables are strings
- ◆ The main advantage of Tcl is that it has no syntax.

3. Python(1991)

■ 動機

◆ Python 係針對 Perl 的程式碼不易閱讀與不易擴充問題，重新精簡了程式的元素，例如利用縮排來取代程式區塊，去除資料型別的概念，並將物件導向觀念引進，因此相對於 Perl 來說，程式碼顯得易懂且易於擴充。以下就 Python 在幾個特色表現優異之處，一般來說 Python 的 Advanced data structures (> Tcl， Perl)、Python 的 Powerfull data-parallel arrays (> Tcl)、Readbility and modularity (> Perl)、 High-level (> C， C++， Fortan) Platform independence (> C++， Java)。

■ 特性

◆ Interpreted， High level， Object-Oriented
◆ Flexible and Extensible
◆ Introspection， self-documenting
◆ Platform Independent
◆ Open Source
◆ Rapidly gaining acceptance

■ 範例

■ Weakness

◆ The main disadvantage of python (apart from not enough people knowing about it) is that it is interpreted from bytecode and therefore executes slower than C++.

4. Ruby(1993)

■ 動機

◆ Ruby(紅寶石)比起其他的 script language 來說，最大的特色來自於這個語言設計時特別強調完整的物件導向概念，除與 Perl 一樣，可用來做系統管理工具，另外也有正規表示式與強大的檔案和字串處理能力，但是確沒有像 Perl 一般不易閱讀的語法，比起 python 來說更 OO， 更有彈性， 因為 Ruby 全都是 OO.。

■ 特性 ◆ simpler syntax ◆ redefinable methods. ◆ Complete, full, pure object oriented language: OOL. ◆ OO in Ruby is carefully designed to be both complete and open for improvements. ◆ single inheritance only， *on purpose*. ◆ true closures. Not just unnamed function， but with present variable bindings. ◆ Ruby features a true mark-and-sweep garbage collector. ◆ needs no variable declarations. ◆ OS independent threading. ◆ highly portable: ■ 範例 ■ Weakness 5. PHP(1995) ■ 動機 ◆ C-like syntax, suitable for web。 ■ 特性 ◆ interpretation ◆ 跨平台 ◆ fast prototyping ◆ connect to DB ◆ widely adpoted ■ 範例 ■ Weakness ◆ 需安裝相關函式庫 ◆ 執行效率較差		
	合計	48