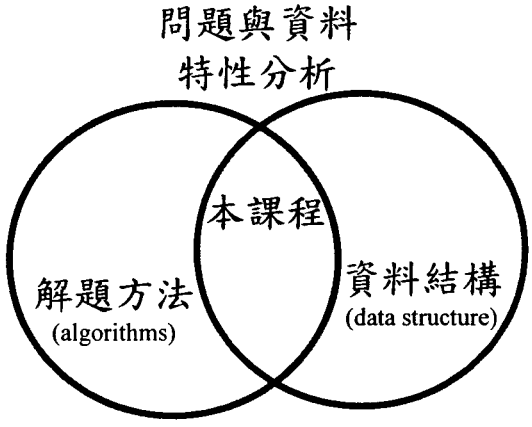
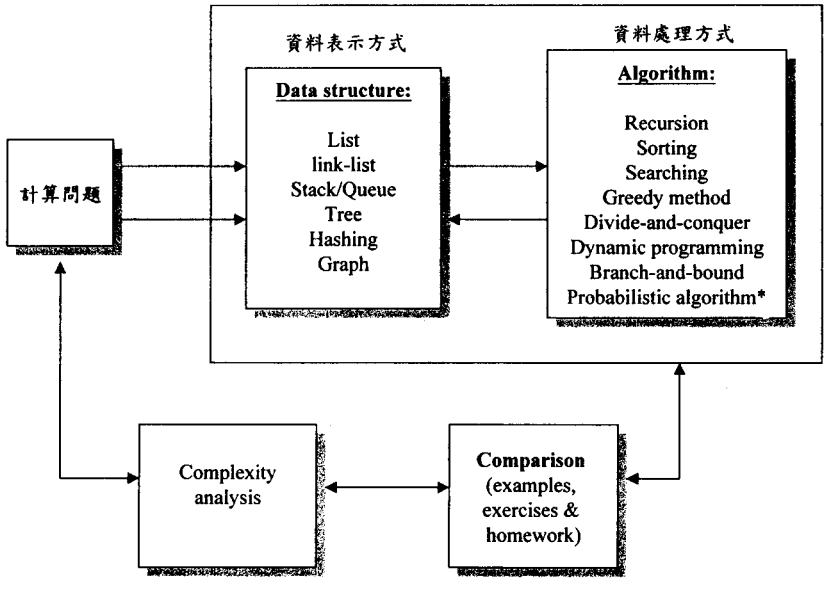
7.2 資料結構與演算法

7.2.1 課程設計理念

「資料結構與演算法」課程主要訓練學生如何運用適確的資料結構與解題策略來解決問題，是程式設計人員必備的核心知識與技術。程式除了需能解決問題外，當資料量龐大時，其執行的效能更是設計考量的重點，程式效能與選用之資料表示方式及演算法有密切關係。本課程的目標，在使學員充分了解各種常用的資料結構及演算法，使其正確的應用在程式設計上以發展出高品質的程式。



本課程的設計著重實際應用，避免艱深的問題解析。課程中儘量使用與生活相關的範例解說，配合程式實作與比較，使學員能活用資料結構與演算法。實作練習中並以物件導向的概念，要求學員將程式寫成元件，以供日後使用。



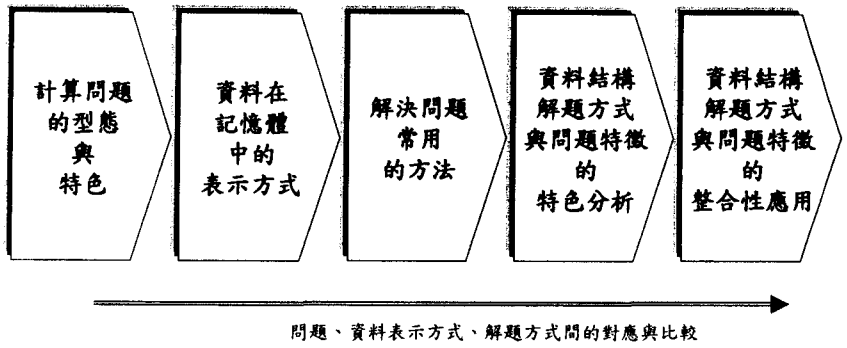
教育目標：

本課程的目標，在使學員充分了解各種常用的資料結構及演算法，且能將其正確的應用在程式設計上以發展出高品質的程式。

設計構想與進行方式：

本課程的設計著重實際應用，避免艱深的問題解析。課程中儘量使用與生活相關的範例解說，配合程式實作與比較，使學員能活用資料結構與演算法。實作練習中並以物件導向的概念，要求學員將程式寫成元件，以供日後使用。在課程進行中，前後單元有關的題材，儘量用同樣的範例說明，以供對照比較，使學員對整個課程融會貫通。

各單元進行的順序的關係如下圖所示：



課程內容：

本課程的內容涵蓋鏈結串列(Linked List)、堆疊(Stack)、佇列(Queue)、遞迴(Recursive)、樹狀結構(Tree)、資料排序(Sorting)、搜尋(Search)、圖形(Graph)、和5種基本的演算法。

Topic 1：課程開始時，以解決問題的觀點，說明資料有那些型態、需要如何處理，並以矩陣相乘的範例讓學員體會資料結構和資料處理的關係。

Topic 2：從陣列在應用上的限制開始，導入鏈結串列的需求，並比較兩者的優缺點及應用場合。

Topic 3、Topic 4：堆疊與佇列亦為密切相關的資料結構，也需要比較其個別的應用時機。

Topic 5：介紹遞迴的程式技巧，以供接續的各單元應用。

Topic 6：第六單元樹狀結構從一般樹的缺點帶入二元樹的主題，再從二元樹可能遇到的問題引入各種特殊樹; 此單元很多內容都深具實用價值，故安排較長的時間，以便充分的探討。

Topic 7：資料排序與搜尋，先介紹一些常用的排序方法，使學者了解各種不同排序方法的效能和適用場合。排序的目的是為了方便搜尋，因此接著介紹各種排序對應的搜尋演算法。

Topic 8：探討圖形。在介紹了圖形的定義與表示法等基本觀念後，以生活上的最短路徑問題說明其應用。

Topic 9：介紹解題的5種演算法，屬於進階的應用。

7.2.2 課程大綱

Core Knowledge	Programming skills/LAB	時數
1. 資料結構概論 ■ 資料與資訊 ■ 資料型態 ◆ 基本資料型態 – 字元、整數、浮點數 ◆ 延伸資料型態 - 陣列、鏈結串列、堆疊、佇列、類別 ■ 資料的操作 – 建立、取用、新增、刪除 ■ 資料結構 + 演算法 = 程式 ■ 評估程式的準則 ◆ 執行速度快 ◆ 佔用記憶體空間少 ■ 資料結構的意義 ■ 演算法的意義 ■ 陣列的意義(複習) ■ 以矩陣相乘運算為例，說明原始資料的類型及採用不同的資料結構，對程式執行速度與記憶體使用的影響。 ■ 空間複雜度的意義 ■ 時間複雜度的意義及 Big-O 表示法 ■ 當資料量很大時，時間複雜度的各種等級的關係 $\log n < n < n \log n < n^2 < n^3 < 2^n$	● 以陣列實作兩個 100×100 的矩陣相乘運算，使用兩種類型的矩陣: 1、所有元素都不為 0，及 2、大部份元素都為 0(稀疏矩陣)。	3
2. 鏈結串列 ■ 陣列結構在新增與刪除資料時的問題 ■ 鏈結串列的意義 ◆ 有序的串列 ◆ 使用動態配置的分散記憶體空間 ■ 鏈結串列的運算 – 走訪、連結、節點的插入、節點的刪除、結構的反轉 ■ 陣列與鏈結串列的比較	● 利用鏈結串列實作兩多項式相乘。就多項式缺項的多寡，比較使用鏈結串列實作與使用陣列實作的優劣。 ● 以程式實作鏈結串列的各種運算。	3

◆ 資料的新增與刪除 ◆ 記憶體空間的大小 ◆ 資料的存取 ■ 鏈結串列的延伸 ◆ 環狀單向鏈結串列 ◆ 雙向鏈結串列 ◆ 環狀雙單向鏈結串列 ■		
3. 堆疊 ■ 堆疊的定義 ■ 堆疊的基本運算 ■ 堆疊的實作 ◆ 用陣列結構 ◆ 用鏈結串列 ■ 堆疊的應用 ◆ 副程式的呼叫與返回 ◆ 運算式的轉換與求值 ■ 應注意的問題 – 資料是否超過堆疊最大容量	● 使用堆疊計算中序運算式的值。	3
4. 佇列 ■ 佇列的定義 ■ 佇列的基本運算 ■ 線性佇列與環狀佇列的實作 ◆ 用陣列結構 ◆ 用鏈結串列 ■ 優先佇列 ■ 雙佇列 ■ 佇列的應用 - 買票問題	● 實作排隊買票問題	3
5. 遞迴 (Recursive) ■ 遞迴的意義 ■ 比較遞迴與迴圈的優缺點，以實作階乘 (Factorial) 運算為例。 ◆ 執行速度 ◆ 使用記憶體空間	● 分別以遞迴與迴圈實作階乘 (Factorial) 運算。	3
6. 樹狀結構 ■ 樹的定義	● 實作二元樹的走訪。 ● 實作 AVL 樹。	12

■ 樹的表示法 – 陣列表示法、鏈結串列表示法 ■ 一般樹的缺點 – 節點的分支數差異大時，浪費儲存空間。 ■ 二元樹的基本性質 ■ 二元樹的儲存方式 ■ 二元樹的建立 ■ 二元樹的走訪 – 前序、中序、後序走訪 ■ 引線二元樹 ■ 二元搜尋樹 ■ 二元樹的應用 – 運算式處理 ■ 二元樹的缺點 – 可能產生歪斜樹 ■ AVL 樹 ■ m 元搜尋樹及 B 樹 ■ Huffman 樹		
7. 資料排序與搜尋 ■ 排序的定義 ■ 內部排序法 ◆ 氣泡排序法 ◆ 選擇排序法 ◆ 二元樹排序法 ◆ 堆積排序法 ◆ 快速排序法 ■ 外部排序法 ◆ 合併排序法 ■ 各種排序法的比較 ■ 搜尋的定義 ■ 在循序結構上的搜尋 ◆ 循序搜尋法 ◆ 二分搜尋法 ◆ 內插搜尋法 ■ 索引結構的搜尋 ◆ 直接索引 ◆ 二元搜尋樹的索引 ◆ B 樹的索引	● 實作下列一種排序。可採分組方式，各組實作不同的排序。 ◆氣泡排序 ◆選擇排序 ◆二元樹排序 ◆堆積排序 ◆快速排序 ◆合併排序 ● 實作下列一種搜尋。可採分組方式，各組實作不同的搜尋，但使用相同的資料。例如: 資料為某班學生的成績，考慮分別使用學生姓名、學號、座號為 key 的情況。 ◆循序搜尋 ◆二分搜尋 ◆內插搜尋 ◆雜湊法	10

■ 雜湊法 (Hashing)		
8. 圖形 ■ 圖形的定義 ■ 圖形的表示法 ■ 表示圖形的資料結構 ◆ 鄰接矩陣(adjacency matrix) ◆ 鄰接串列(adjacency list) ◆ 加權圖的鄰接矩陣 ■ 圖形的走訪 ◆ 廣度優先走訪(Breadth First Search,BFS) ◆ 深度優先走訪(Depth First Search,DFS) ■ 擴張樹 (Spanning Tree) ■ 最短路徑 (Shortest Path) ■ 拓樸排序 (Topological Sorting) ■ 關鍵路徑 (Critical Path)	● 實作 Dijkstra 演算法找最短路徑。 ● 實作拓樸排序。	6
9. 解題方法 ■ 貪婪演算法 (Greedy Algorithms) ◆ 貪婪演算法的精神 ◆ 貪婪演算法實例一：Minimum Spanning Trees (Kruskal and Prim Algorithms) ◆ 貪婪演算法實例二：Huffman 樹 ■ 分裂征服演算法 (Divide-and-Conquer Algorithms) ◆ 分裂征服演算法的精神 ◆ 分裂征服演算法實例一：二分搜尋法 ◆ 分裂征服演算法實例二：快速排序法 ■ 動態規劃演算法 (Dynamic Programming) ◆ 動態規劃演算法的精神 ◆ 動態規劃演算法實例一：Knapsack Problem	● 實作 Game tree	20

◆ 動態規劃演算法實例二：Matrix-Chain Problem ■ 分支設限演算法 (Branch-and Bound) ◆ 分支設限演算法的精神 ◆ 分支設限演算法實例一：Game Trees ◆ 分支設限演算法實例二：3-Coloring Problem ■ 機率演算法 (Probabilistic Algorithms) ◆ 機率演算法的精神 ◆ 機率演算法實例一：Randomized Quicksort ◆ 機率演算法實例二：Random Search		
	合計	63