C++ 程式設計 (week 1)

姓名: _____ 103/02/18(二)

1. 針對下面這些 C 語法請勾選你自己的狀態

完全瞭解	有些疑點	偶爾出錯	大致瞭解	常常出錯	不清楚	無法運用		其他狀況
☐	☐	☐	☐				a. 各種基本資料型態	_____
☐	☐	☐	☐				b. 運算式,運算子優先順序、結合性	_____
☐	☐	☐	☐				c. 條件敘述 (if, switch, ?=)	_____
☐	☐	☐	☐				d. 迴圈 (for, while, do/while)	_____
☐	☐	☐	☐				e. 函式使用 (輸入/輸出參數)	_____
☐	☐	☐	☐				f. 陣列 (一維/二維/多維)	_____
☐	☐	☐	☐				g. 字串與相關函式	_____
☐	☐	☐	☐				h. 遞迴函式	_____
☐	☐	☐	☐				i. 使用者自定資料型態 (struct)	_____
☐	☐	☐	☐				j. 指標與動態記憶體配置	_____
☐	☐	☐	☐				k. 文字模式檔案處理	_____
☐	☐	☐	☐				l. 二進位模式檔案處理	_____
							m. 其他語法: _____	

2. 過去幾個學期寫程式作業時, 覺得自己**產生**程式的能力屬於哪一類?
   - ☐ 沒有問題, 由課本的基本範例很容易轉換為作業需求的程式
   - ☐ 比較慢, 需要參考相關的範例或是提示
   - ☐ 很難下手, 需要參考同學/學長或是網路上找到的相關程式
   - ☐ 其它狀況: _____

3. 針對下面這些資料結構與相關演算法項目請勾選你自己的狀態

完全瞭解	大致瞭解	有些疑問	寫過程式	不清楚	無法運用	
☐	☐	☐	☐			a. 陣列 array
☐	☐	☐	☐			氣泡排序法 Bubble sort
☐	☐	☐	☐			選擇排序法 Selection sort
☐	☐	☐	☐			插入排序法 Insertion sort
☐	☐	☐	☐			計數排序法 Counting sort
☐	☐	☐	☐			快速排序法 Quick sort
☐	☐	☐	☐			基數排序法 Radix sort (Bucket sort)
☐	☐	☐	☐			堆積 Heap
☐	☐	☐	☐			堆積排序法 Heap sort
☐	☐	☐	☐			大整數/多項式運算
☐	☐	☐	☐			最長共同子串列 Longest Common Subsequence
☐	☐	☐	☐			b. 鏈結串列 list
☐	☐	☐	☐			高次多項式 Polynomial

□	□	□	□	稀疏矩陣運算 Sparse Matrix
□	□	□	□	合併排序法 Merge sort
□	□	□	□	c. 堆疊 stack
□	□	□	□	中序式轉前序式/後序式 Infix to Prefix/Postfix
□	□	□	□	運算式計算 Expression Evaluation
□	□	□	□	深度優先搜尋樹 Depth First Search Tree
□	□	□	□	d. 佇列 queue
□	□	□	□	廣度優先搜尋樹 Breadth First Search Tree
□	□	□	□	e. 圖形 graph
□	□	□	□	Dijkstra 最短路徑 Shortest path
□	□	□	□	Floyd 最短路徑 Shortest path
□	□	□	□	Kruskal 最小生成樹 Minimal spanning tree
□	□	□	□	Prim 最小生成樹 Minimal spanning tree
□	□	□	□	拓樸順序 Topological ordering
□	□	□	□	關鍵路徑 Critical path
□	□	□	□	f. 樹 tree
□	□	□	□	二元樹 binary tree
□	□	□	□	引線二元樹 threaded binary tree
□	□	□	□	二元搜尋樹 binary search tree
□	□	□	□	高度平衡二元樹 AVL tree
□	□	□	□	2-3-4 與紅黑樹 2-3-4 tree/RB tree
□	□	□	□	平衡樹 Balance tree
□	□	□	□	霍夫曼編碼 Huffman coding

h. 其他: _____

4. 請問你學過哪些程式語言 □BASIC, □C, □C++, □Objective-C, □JAVA,
   其它: _____
5. 請問你使用過哪些程式開發環境 □命令列, □DevC++, □Visual Studio, □Eclipse,
   □ XCode, 其它: _____
6. 請問什麼是 C 程式中的全域變數? _____
   有什麼用途? _____
7. 如果下列是設計一個 C 程式可能的步驟, 請根據你掌握與瞭解的狀況勾選

完全可掌握	此步驟常遇見無解問題	此步驟通常跳過無用	沒有此步驟	
□	□	□	□	瞭解程式的要求
□	□	□	□	釐清整個程式的輸入資料與輸出資料
□	□	□	□	設計多個標準的範例資料
□	□	□	□	設計資料在電腦裡的表示方式
□	□	□	□	手動一步一步在紙上完成每一個動作(動作需要夠單純)
□	□	□	□	釐清每一個動作的輸入與輸出

☐ ☐ ☐ ☐ 把每一個動作轉換為 C 程式

☐ ☐ ☐ ☐ 重複出現的動作運用迴圈來完成

☐ ☐ ☐ ☐ 不熟悉或是沒有用過的語法獨立撰寫小程式驗證其表現

☐ ☐ ☐ ☐ 必要時運用原始碼偵錯器來確定單一語法的功能

☐ ☐ ☐ ☐ 設計函式將每一個動作的細節打包起來

☐ ☐ ☐ ☐ 設計比較可能出現錯誤的資料

☐ ☐ ☐ ☐ 根據你的程式處理的邏輯，設計程式特別處理的測試資料 (例如排序時完全反序的數列，有相同元素的序列等)

☐ ☐ ☐ ☐ 撰寫單元測試的程式碼

☐ ☐ ☐ ☐ 測試每一個函式功能是否正確

☐ ☐ ☐ ☐ 測試整個程式的功能

☐ ☐ ☐ ☐ 結果不正確時，分析並且觀察每一個動作的處理過程 (輸入與輸出資料是否如預期)

☐ ☐ ☐ ☐ 找到有問題的程式片段並分析出現問題的資料

偵錯/除錯

☐ ☐ ☐ ☐ 觀察為什麼你寫的程式沒有辦法對指定的輸入產生正確的輸出，分析究竟是哪一個步驟使得程式輸出目前錯誤的結果

☐ ☐ ☐ ☐ 離正確結果差異不太大的話，直接修改程式，再觀察中間處理過程，是否所有資料的改變如同預期

☐ ☐ ☐ ☐ 如果一直改不對，將有問題的程式抽離原來的程式，簡化它的環境，獨立設計測試資料來觀察程式的表現

☐ ☐ ☐ ☐ 必要的時候運用原始碼除錯器，直接觀察程式的細部表現

☐ ☐ ☐ ☐ 搜尋相關功能的程式碼 (Google)

最後這個步驟在做作業的時候當然是不鼓勵的，因為作業的目標是希望你能夠真正掌握某一種運作的方式，並且培養你轉換抽象想法成為程式的能力，如果你由搜尋到的程式著手，很多關鍵步驟你就直接跳過去了，作業就沒有預期的用途了。

8. 請運用 C 的結構 (struct) 語法，定義一個最多能夠存放 10 列 10 行二維整數矩陣的資料型態 Matrix，這種資料型態可以存放例如下列 5 x 3 的矩陣:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 2 \\ 1 & 3 & 4 & 2 & 5 \\ 5 & 4 & 1 & 6 & 2 \end{pmatrix}$$

9. 接上題，請寫一個函式，傳入兩個矩陣，如果維度正確的話，計算兩個矩陣乘積，存入第三個矩陣中，適當設定矩陣的維度，函式回傳此乘積矩陣，如果兩個矩陣不能相乘，乘積矩陣的維度設為 0

10. 在黑板上我們說明一個產生 n 個數字排列的遞迴演算法, 請寫一個 C 函式實作