無師自通的 C 語言程式設計-附大學程式設計先修檢測 (APCS) 試題解析-勘誤表

修訂日期：2023/2/15

|  |  |  |  |
| --- | --- | --- | --- |
| 頁 | 位置(列) | 原始的文字 | 更正或調整後的文字 |
| 43 | 4  6  7  8 | -2  b / 2  b / 2.0  b % 3 | -1  a / 2  a / 2.0  a % 3 |
| 45 | 6 | 或「迴圈結構」 | 或「迴路結構」 |
| 45 | 20 | 「=」 | 「!=」 |
| 46 | 7 | (a>3)  || 0  (b<=2) | (a>3)  || 1  (b<=2) |
| 46 | 11 | a ^ b | (a>3)  ^  (b<2) |
| 53 | 13  14  15  16  17 | (2) D02A16 = 11010000001010102 = 1500528，548710 = 125578 =(B) 的數據  D02A16 + 548710 = 1500528 + 25578 = 1626318。  (3) 548710 = 156F16  D02A16 + 548710 = D02A16 + 156F16 = 5877716。 | (2) D02A16 = 11010000001010102 = 1500528，548710 = 125578  D02A16 + 548710 = 1500528 + 125578 = 1626318=(B) 的數據。  (3) 548710 = 156F16  D02A16 + 548710 = D02A16 + 156F16 = E59916。 |
| 54 | 7 | 4. 若 a, b, c, d, e 均為整數變，下列哪個算式計結果與 a+b\*c-e 計算結果 | 4. 若 a, b, c, d, e 均為整數變，下列哪個算式計算結果與 a+b\*c-e 計算結果算式 |
| 55 | 最後  一列 | 「||」，&「&&」及「||」的運算順序由左而右。 | 「||」，~~&~~「&&」及「||」的運算順序由左而右。 |
|  |  |  |  |
| 74 | 1  5  9 | %d pieces\n  %d pieces\n  %d pieces\n | %d coins\n  %d coins\n  %d coins\n |
| 82 | 12 | • 在「if ()」、「{」 及「}」後面，都不能有「;」。 | • 在「if ()」、「else」、「{」 及「}」後面，都不能有「;」。 |
| 85 | 3  4  5  6  11 | if (num / 1000 == 0)  printf("%d不是**三**位數的整數\n", num);  else  printf("%d為**三**位數的整數\n", num);  1234為**三**位數的整數 | if (num < 1000 || num > 9999)  printf("%d不是四位數的整數\n", num);  else  printf("%d為四位數的整數\n", num);  1234為四位數的整數 |
| 86 | 15 | • 在「if ()」、「{」 及「}」後面，都不能有「;」。 | • 在「if ()」、「else if」、「else」、「{」 及「}」後面，都不能有「;」。 |
| 99 | 10 | if (year % 400 = 0) | if (year % 400 == 0) |
| 102 | 12 | 解 答案：(A) | 解 答案：(D) |
| 113 | 19  20  21  22 | 輸入第1個數值: 8  輸入第1個數值: 6  輸入第1個數值: 2  輸入第1個數值: 10 | 輸入第2個數值: 8  輸入第3個數值: 6  輸入第4個數值: 2  輸入第5個數值: 10 |
| 116 | 5 | num = num / 10; //去掉num的個位數,即num除以10所得的餘數 | num = num / 10; // 取得num除以10的商 |
| 119 | 倒數第11列 | 一列式敘述。 | 一列敘述。 |
| 132 | 4 | i <=50 | i <=5 |
| 142 | 6 | (1) 程式第 7~13 列 | (1) 程式第 3~13 列 |
| 143 | 10  11  12  13 | (3) 變數 a 代表第 1 個數值，變數 b 代表第 2 個數值也代表後續  的每一個數值。因此，下一個數值 b 是前面兩個數值之和，  因此 (a) 的答案應填「b=a+b;」，a+b 的值就成為求下一個數  值的 b。 | (3) 變數 a 代表第 1 個數值，變數 b 代表第 2 個數值，也代表後續的每一個數值。由於每一個數值等於是前面兩個數值之和， 因此 (a) 的答案應填「b=a+b;」，「a+b」 的值就成為計算下一個數值時的「b」。 |
| 143 | 14  15  16  17  18  19 | (4) 在求每個數值時，程式第 6 列先將 b 存入變數 temp，然後程  式第 8 列將 temp 指定給 a。第 6 列及第8列的目的，是將 a值變成前一次的 b 值，b 值就成為求下一個數值的a。輸出每一個數值，照理 (b) 的答案應填「b」，但 b 在 (a) 已改變，因此，需輸出前一次的 b 值，即 a 值。故 (b) 的答案，應填「a」。 | (4) 在求每個數值時，程式第 6 列先將 b 存入變數 temp，然後程式第 8 列將 temp 指定給 a。第 6 列及第8列的目的，是將前一次的「b」值指定給「a」，當作計算下一個數值時的「a」。變數「b」代表後續的每一個數值，程式第9列是要輸出後續的每一個數值，故(b)的答案，應填「b」。 |
| 150 | 19 | for (int j=0; j<N; =j+1) { | for (int j=0; j<N; j=j+1) { |
| 151 | 3 | 並沒有在程式第 4 列與第 5 | 並沒有在程式第 3 列與第 4 |
| 163 | 14 | 解：sqrt(36, 0.5) | 解：sqrt(36) |
| 164 | 5 | printf("ax^2+bx+c=0的兩個根，分別為%f及%f\n",r1,r2); | printf("ax^2+bx+c=0的兩個根，分別為%lf及%lf\n",r1,r2); |
| 164 | 13 | 1.0x^2-1.0x-6.0=0的兩個根，分別為3.000000及-2.000000 | ax^2+bx+c=0的兩個根，分別為3.000000及-2.000000 |
| 164 | 16  17 | • b 2 - 4ab >= 0，才會得到實數根；否則得到的兩個根分都是「-1.#IND00」。 | • b 2 - 4ac >= 0，才會得到實數根；否則得到的兩個根都是「-1.#IND00」。 |
| 168 | 2  3  4 | 與數學運算有關的常用內建數學函式，是宣告在  「ctype.h」標頭檔中。常用的內建數學函式，請  參考「表 6-8」至「表 6-12」。  使用內建數學函式之前， | 與字元運算有關的常用內建字元函式，是宣告在「ctype.h」標頭檔中。常用的內建字元函式，請  參考「表 6-8」至「表 6-12」。  使用內建字元函式之前， |
| 176 | 倒數第4列 | // 宣告有4個元素的一維字元陣列變數score | // 宣告有4個元素的一維整數陣列變數score |
| 177 | 倒數第9列 | // 宣告有5個元素的一維字元陣列變數word， | // 宣告有5個元素的一維字元陣列變數name， |
| 181 | 6 | int movelength; | int movelength=1; |
| 184 | 14,  15,  16 | 若第i個字元與第(i+1)個字元不同時為大寫或小寫，則(str[i]-91) \* (str[i +1]-91) < 0，代表這一段連續大寫或小寫字元到索引 i 為止。 | 若第j個字元與第 (j+1) 個字元不同時為大寫或小寫，則 (str[j]-91) \* (str[j+1]-91) < 0，代表這一段連續大寫或小寫字元到索引j 為止。 |
| 185 | 12,  13 | int uppernum=0; // 0:連續k個字元不是全部大寫 1:全部大寫  int lowernum=0; // 0:連續k個字元不是全部小寫 1:全部小寫 | int uppernum=0; // 紀錄連續大寫字元的長度  int lowernum=0; // 紀錄連續小寫字元的長度 |
| 213 | 4 | 子，必須。 | 子。 |
| 219 | 1 | 常用的內建字串函式(五) | 常用的內建字串函式(六) |
| 222 | 倒數第7列 | 不必自行設定隨機亂數組別的 | 必須自行設定隨機亂數組別的 |
| 230 | 13 | for (i=1 ; i<=4 ; =i+1) { | for (i=1 ; i<=4 ; i=i+1) { |
| 232 | 4,  9,  22 | for (int i=0; i<n; i+1) {  for (int i=0; i<=n/2; i+1)  (2) 程式第 9~10 列 | for (int i=0; i<n; i=i+1) {  for (int i=0; i<=n/2; i=i+1)  (2) 程式第 8~9 列 |
| 228 | 倒數第10列 | charstr[20]={'H', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', '!', '\0'}; | char str[20]={'H', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', '!', '\0'}; |
| 231 | 倒數第2列 | (2) 程式第 3~7 列的迴圈變數 i 從 0 開始，若 I <= (n-2)，則執行 | (2) 程式第3~7列的迴圈變數i從0開始，若i <= (n-2)，則執行 |
| 254 | 2 | 並不能完全無法滿足使用者的所有需求 | 並無法完全滿足使用者的所有需求 |
| 286 | 6 | for (int j=i; j<=n; j+1) | for (int j=i; j<=n; j=j+1) |
| 286 | 18  19  20  21  22  23  24 | 從 1 變化到 2k時，且變化規則為 1，2，4，⋯，2k。1，2，  4，⋯，2k 共有 (k+1) 數，故迴圈會執行 (log22k + 1) = (k+1)  次。現在程式第 5 列 for迴圈的迴圈變數k從1變化到 n 時，  且變化規則為 1，2，4，⋯，n。1，2，4，⋯，n 共有 (log2n  +1) 數，故迴圈會執行 (log2n + 1) 次。  由(1)及(2)的說明可知，第5列for迴圈共執行n(n+1) log2n +  1 /2，最後 x 值為 n(n+1) ⌊log2n + 1⌋/2。 | 從 1 變化到 2k(即，1、2、4、⋯及2k)時，共有  (k+1)個數，則迴圈會執行log22k + 1(= k+1)  次。現在程式第5列for迴圈的迴圈變數k從1變化到 n (即，1、2、4、⋯及n)時，共有(log2n+1)個數，故迴圈會執行 (log2n + 1) 次。  由(1)及(2)的說明可知，第5列for迴圈共執行n(n+1)  ⌊log2n + 1⌋/2次，最後 x 值為 n(n+1) ⌊log2n + 1⌋/2。 |
| 289 | 1 | (D) 每一行都執得到 | (D) 每一行都執行得到 |
| 293 | 倒數  第2列 | (3) 執行程式第 20 列時，呼叫先呼叫 f2(10)， | (3) 執行程式第 20 列時，先呼叫 f2(10)， |
| 297 | 12 | return 函式名稱([參數串列]); | return 包含函式名稱([參數串列])的運算式; |
| 302 | 倒數  第5列 | temp 過渡木釘上： | temp 過渡木釘上。 |
| 349 | 最後  一列 | n>=0 | n > 0 |
| 352 | 11 | (1) 程式第 5 列，若 r=0，則表示a整除b，a 和 b 的最大公因數 | (1) 程式第 5 列，若 r=0，則表示b整除a，a 和 b 的最大公因數 |
| 353 | 9 | 1+1+1+1+1+1+1+1+1+1+1+1=10 | 1+1+1+1+1+1+1+1+1+1=10 |
| 354 | 1 | 函數 f 定義如下， | 函數 f 的定義如上， |
| 357 | 倒數  第1列 |  |  |
| 358 | 倒數  第1列 |  |  |
| 364 | 倒數  第6列 | 因此，呼叫 A1(13)，共輸出 13+17=30 個 '\*'。 | 因此，呼叫 A2(13)，共輸出 13+17=30 個 '\*'。 |
| 372 | 7 | 故執行 G(3) 後所回傳的值為 0+a[0]+a[1]+a[2]+a[3]=9。 | 故執行 G(3) 後所回傳的值為 0+a[0]+a[1]+a[2]+a[3]=14。 |
| 374 | 10 | (2) G(a, x) 的回傳值都可寫成 6n，n>=0。 | (2) G(a, x) 的回傳值都可寫成 6n，n>=0。 |
| 388 | 22  23 | • 程式第 16 列「ptr= data[k];」中的「data[k]」，代表二維陣列data[2][3]的第k列資料。 | 程式第 16 列「ptr= data[k];」中的「data[k]」，代表二維陣列data的第k列資料。 |
| 395 | 2  3 | • 程式第 17列「ptr= data[k];」中的「data[k]」，代表二維陣列data[2][3]的第k列資料。 | 程式第 17 列「ptr= data[k];」中的「data[k]」，代表二維陣列data的第k列資料。 |
| 399 | 8 | 會形成遞迴結構，這是不被允引的。 | 會形成遞迴結構，這是不被允許的。 |
| 401 | 倒數  第4列 | 在上例時，「pointer = &normal;」。因此，本例執行 | 在上例宣告「\*pointer = &normal;」，本例執行 |
| 401 | 倒數  第3列 | >age = 36;」後，normal.age 的值內容也變成 36。 | >age = 36;」後，normal.age 的內容也變成 36。 |
| 401 | 倒數  第1列 | 在結構型態有 k 個欄位的情狀 | 在結構型態有 k 個欄位的情況下 |
| 403 | 5 | [注意] 結構指標變數，只能指向相同的結構變數。 | [注意] 結構指標變數，只能指向型態相同的結構變數。 |
| 405 | 倒數  第13列 | 程式第18列「scanf("%s", student[i].no);」， | 程式第 20列「scanf("%s", coding[i].no);」， |
| 405 | 倒數  第11列 | 20「scanf("%d", &student[i].score);」， | 22「scanf("%d", & coding[i].score);」， |
| 425 | 16,  17 | 4  5 | 5  4 |
| 427 | 16 | // 將葉節點陣列leafqueu中的最1個葉節點的索引值往後移一個 | // 將葉節點陣列leafqueue中的最後1個葉節點的索引值往後移一個 |
| 431 | 22 | // 共有(n-1)組的父節點與子節點 | // 共有(n-1)組的父節點與子節點，用來記錄n個節點各自的父子關係 |
| 431 | 28 | // 輸入(n-1)個父節點與子節點 | // 輸入(n-1)組父節點與子節點 |

第48頁內容更正：

表 2-8 位元運算子的功能說明（假設 a=2，b=1）

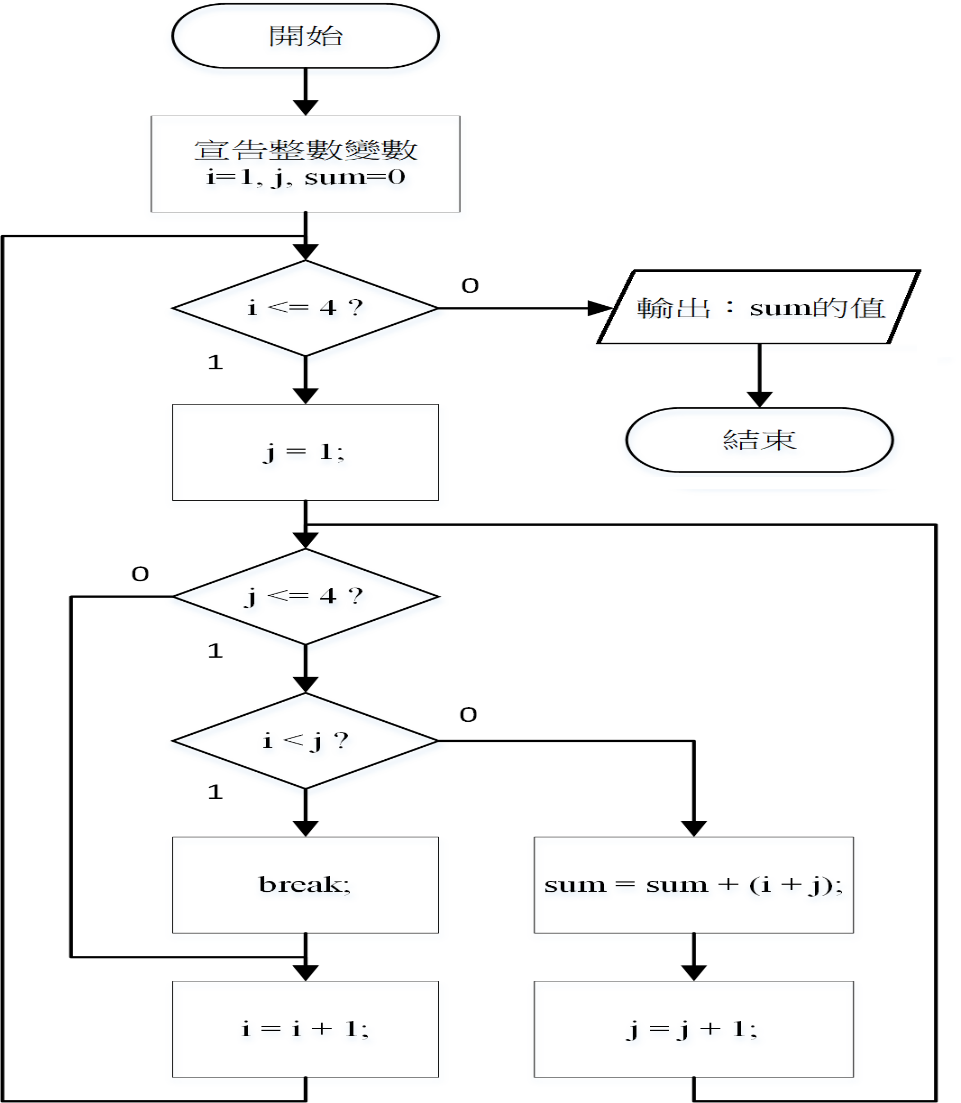
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 運算子 | 運算子類型 | 作用 | 例子 | 結果 | 說明 |
| & | 二元運  算子 | 將兩個整數轉成二進位整數後，對兩個二進位整數的每一個位元值做「&」（且）運算 | a & b | 0 | 1. 若兩個二進位整數對應的位元值，皆為 1，則運算結果為1；否則為 0 2. 將(1)的結果，再換成十進位整數 |
| | | 二元運  算子 | 將兩個整數轉成二進位整數後，對兩個二進位整數的每一個位元值做「|」（或）運算 | a | b | 3 | 1. 若兩個二進位整數對應的位元值皆為 0，則運算結果為0；否則為 1 2. 將(1)的結果，再換成十進位整數 |
| ^ | 二元運  算子 | 將兩個整數轉成二進位整數後，對兩個二進位整數的每一個位元值做「^」（或互斥）運算 | a ^ b | 3 | 1. 若兩個二進位整數對應的位元值，一個為1，另一個為0，則運算結果為1；否則為 0 2. 將(1)的結果，再換成十進位整數 |
| ~ | 一元運  算子 | 將整數轉成二進位整數後，對二進位整數的每一個位元值做「~」（否定）運算 | ~a | -3 | 1. 進位整數的位元值為 0，則運算結果為 1；否則為 0 2. 若最高位元值為 1，表示最後結果為負，則必須使用 2的補數法（即，1 的補數之後 +1），將它轉成十進位整數 |
| << | 二元運  算子 | 將整數轉成二進位整數後，往左移動幾個位元，相當於乘以2的幾次方。 | a << 1 | 4 | 1. 往左移動後，超出儲存範圍的數字捨去，而右邊多出的位元就補上 0 2. 若最高位元值為 1，表示最後結果為負，則必須使用 2的補數法（即，1 的補數之後 +1），將它轉成十進位整數 |
| >> | 二元運  算子 | 將整數轉成二進位整數後，往右移動幾個位元，相當於除以2的幾次方。 | a >> 1 | 1 | 1. 往右移動後，超出儲存範圍的數字捨去，而左邊多出的位元就補上 0 2. 將(1)的結果，再換成十進位整數 |

第100頁的範例10流程圖更正如下：



範例 10 流程圖

第129頁的範例8流程圖更正如下：



範例8 流程圖