b02303042 經濟四 鄭雅文

第一題

(a) **cin >> a >> b >> c;** 改成 **std::cin >> a >> b >> c;**

**cout << min << " is the smallest";**改成**std::cout << min << " is the smallest";**

因為**cout**, **cin** 是定義在namespace std裡面，如果未告訴compiler是哪裡定義的**cin**, **cout**，compiler會不知道是哪個**cin**, **cout**的定義。

(b) 不一定會產生一模一樣的執行結果，因為沒有大括號，else會選擇最近的if的條件的反向，即code的執行順序會如下：

|  |
| --- |
| int min=c;  if(a <= b){  if(a <= c){  min = a;  }  else{  if(b <= c)  min = b;  }  } |

所以(a>b)的情況下處理不到，min永遠為原本指定的c值。

改變前較好，因為程式是正確的；改變後程式錯誤。

(c) 執行結果一模一樣，兩者改變前後比較條件次數也相同，效率一樣好。但是改變後程式碼較精簡，較易讀，這點較好。

(d) 執行結果不會一模一樣，因為 (b <= c) 不是在 (a <= b && a <= c) 不成立的情況下才會執行。表示不論a是否為min，若 (b <= c) 則min都會被assign為b值。

eg. a=10, b=20, c=30，即使min一開始設為a，但仍會做第二次if測試，又(b<=c)成立，所以min會被assign為20。

(e) 執行結果一模一樣，效率較好，因為如果兩數相同，可以少一步assign值到min的步驟。

第二題

(a) 輸出

|  |
| --- |
| 0  5 |

因為and operator只要一個條件不成立，就不會check第二個條件，且&&的結合性是從左到右，所以會先check左邊的條件。if((a > 10) && (b = 1)) 此行因為(a > 10)的條件不成立，所以不會執行第二個條件。

or operator也是有左到右的結合性，但因為第一個條件沒有成立，所以執行第二個條件，b就被assign為5。

(b)

|  |
| --- |
| 0  0 |

因為 == 為比較運算子，檢查左右兩邊的數字值有沒有相等，再return true or false。因為b的值沒有被assign為別的數，所以b始終為0。