中四資訊及通訊科技練習卷：程式編寫與算法基礎

中四（S4）　資訊及通訊科技　｜　時限：75 分鐘　｜　總分：100 分　｜　共 12 題

甲部：運算、迴圈與條件結構（28 分）

第 1 – 4 題：運算次序與整數除法、while 迴圈追蹤、if-elif-else 條件判斷、線性搜尋與二分搜尋比較

1.

某程式使用整數變數 a 及 b，程式中 a 的值為 87，b 的值為 12。

(a) 寫出 Python 運算式 a // b、a % b 及 a / b 的結果，其中 a / b 須準確至小數後兩位。（3 分）
(b) 計算運算式 4 + 3 * 2 ** 3 - 20 // 6 的值，並說明當中的運算次序。（2 分）
(c) 某同學以 0.1 + 0.2 == 0.3 測試浮點數運算，結果為 False。解釋原因。（2 分）

答案：

(a) 87 // 12 = 7（取商的整數部分）；87 % 12 = 3（餘數，因 12 × 7 = 84）；87 / 12 = 7.25（真除法）。（3 分）
(b) 先計算 2 ** 3 = 8（指數優先），再計算 3 × 8 = 24 及 20 // 6 = 3，最後 4 + 24 - 3 = 25。（2 分）
(c) 0.1 及 0.2 在二進制浮點中不能精確表示，儲存值為近似值；兩者相加後與 0.3 的儲存值相差約 5×10^{-17}，故 == 判定為 False。應改用 abs(a - b) 小於 1e-9 的方式比較。（2 分）

2.

以下為一段 Python 程式碼：

```
total = 0;
i = 1;
while i <= 8:
    total = total + i * i
    i = i + 2
```

(a) 完成追蹤表，寫出每次迭代後 i 及 total 的值。（4 分）
(b) 寫出程式執行完畢後 total 的值，以及迴圈體執行的次數。（2 分）
(c) 若把 i = i + 2 改為 i = i + 1，total 的值會變成多少？（1 分）

答案：

(a) 追蹤表：i = 1，total = 1；i = 3，total = 10；i = 5，total = 35；i = 7，total = 84；之後 i = 9，條件 9 不大於 8 不成立，迴圈結束。（4 分）
(b) total = 84（1 + 9 + 25 + 49），迴圈體執行 4 次。（2 分）
(c) 若改為 i = i + 1，會累加 1 至 8 的平方，total = 204。（1 分）

3.

以下程式按分數輸出等級：

```
if score >= 85:
    grade = "A"
elif score >= 70:
    grade = "B"
elif score >= 50:
    grade = "C"
else:
    grade = "F"
```

(a) 分別寫出 score 為 85、84、70 及 49 時的輸出。（4 分）
(b) 若把第一個條件改為 score > 85，score 為 85 時的輸出會是什麼？指出這屬於哪一類錯誤。（2 分）
(c) 說明條件排列次序為何會影響程式結果。（1 分）

答案：

(a) score = 85 → **A**；score = 84 → **B**；score = 70 → **B**；score = 49 → **F**。（4 分）
(b) 條件改為 score 大於 85（不含 85）後，score = 85 不符第一條件，落入第二條件（85 不小於 70）→ 輸出 **B**。程式可執行但結果與規格不符，屬**邏輯錯誤**（logic error）。（2 分）
(c) elif 由上行下行逐一判斷，首個成立的分支即執行並跳離整個結構；若把較寬鬆的條件（如 score 不小於 50）放在前面，高分數會先被歸為 C，後面較嚴格的等級永遠不會被選中。（1 分）

4.

某電話簿已按姓名筆劃由小至大排序，共有 2,000 筆記錄。程式可用線性搜尋或二分搜尋查找記錄。

(a) 寫出線性搜尋在最壞情況及平均情況下的比較次數。（2 分）
(b) 寫出二分搜尋在最壞情況下的比較次數，並列出計算過程。（2 分）
(c) 二分搜尋對資料有何先決條件？若資料未排序，先排序會帶來什麼額外成本？（3 分）

答案：

(a) 線性搜尋最壞情況需 2000 次比較（目標在最後或不存在）；平均情況約 1000.5 次，即 $(n + 1) \div 2$。（2 分）
(b) 二分搜尋每次把搜尋範圍減半，最壞比較次數為 $\text{ceil}(\log_2(n + 1)) = \text{ceil}(\log_2 2001) \approx \text{ceil}(10.97) = 11$ 次。（2 分）
(c) 先決條件是**資料必須已排序**（且能隨機存取）。若資料未排序，須先排序，以合併排序為例約需 $O(n \log n)$ 時間，這成本可能超過二分搜尋節省的比較次數；故只查一次時線性搜尋反而較實際。（3 分）

乙部：排序、函數與資料結構（32 分）

第 5 – 8 題：冒泡排序追蹤、函數參數與預設值、列表統計、字串處理

5.

以下程式以冒泡排序把列表由小至大排列：

```
data = [45, 12, 78, 33, 56, 90]
```

(a) 寫出完成第一趟（first pass）後列表的內容。（4 分）
(b) 若程式不設「已排序即停止」的優化，完成整個排序共需多少次比較？以 n 表示公式。（2 分）
(c) 這個列表最少需要多少趟才完成排序？（1 分）
(d) 若某一趟完全沒有交換，代表什麼？可如何利用這點提早結束？（1 分）

答案：

(a) 第一趟由左至右比較相鄰元素並交換：45 與 12 交換；45 與 78 不換；78 與 33 交換；78 與 56 交換；78 與 90 不換。結果為 [12, 45, 33, 56, 78, 90]，最大值 90 已到最後位置。（4 分）
(b) 比較次數為 (n − 1) + (n − 2) + ⋯ + 1 = n(n − 1) ÷ 2；n = 6 時為 6 × 5 ÷ 2 = 15 次。（2 分）
(c) 最少 3 趟（每趟固定一個元素到最終位置，最後一趟只剩一個元素無須比較）。（1 分）
(d) 某趟完全沒有交換，代表列表已排序；可設 swapped 旗標，該趟結束時若為 False 即 break，最好情況只需 1 趟、n − 1 次比較，時間複雜度由 O(n²) 降至 O(n)。（1 分）

6.

以下函數計算補習班費用，每小時 120 元，單次上限 1,000 元：

```
def lesson_fee(hours, rate=120, cap=1000):
    amount = hours * rate
    if amount > cap:
        amount = cap
    return amount
```

(a) 寫出 lesson_fee(6)、lesson_fee(10) 及 lesson_fee(8.5) 的回傳值。（3 分）
(b) 解釋 rate 及 cap 作為預設參數的好處。（2 分）
(c) 若傳入 hours 為 -3，函數會回傳什麼？指出問題並提出修正方法。（3 分）

答案：

(a) lesson_fee(6) = 6 × 120 = 720；lesson_fee(10) = 1,200，超過上限，回傳 1000；lesson_fee(8.5) = 1,020，同樣被上限截為 1000。（3 分）
(b) 預設參數讓常用值（120 元、1,000 元上限）不必每次傳入，呼叫更簡潔；需要時仍可覆寫（例如 lesson_fee(6, 150)），提高函數重用性而不改函數本體。（2 分）
(c) hours = -3 時回傳 -360，屬不合理的負數費用。應在函數開頭驗證輸入，例如 if hours 小於 0: raise ValueError("hours must be non-negative")，或回傳 0 / 空值並由呼叫者處理。（3 分）

7.

某班 7 位學生的測驗分數如下：

```
marks = [72, 58, 91, 64, 85, 47, 79]
```

(a) 寫出總分、最高分及最低分。（3 分）
(b) 計算平均分，準確至小數後一位。（2 分）
(c) 寫出高於平均分的學生人數。（2 分）
(d) 寫出可計算高於平均分人數的程式片段。（1 分）

答案：

(a) 總分 = 72 + 58 + 91 + 64 + 85 + 47 + 79 = 496；最高分 91；最低分 47。（3 分）
(b) 平均 = 496 ÷ 7 = 70.857…，即 70.9。（2 分）
(c) 高於 70.857 的分數為 72、91、85、79，共 4 人。（2 分）
(d) avg = sum(marks) / len(marks)；count = 0；for m in marks:； if m 大於 avg: count += 1；（或 sum(1 for m in marks if m 大於 avg)）。（1 分）

8.

某程式中字串變數 s 的值為 "Programming"。

(a) 寫出 len(s)、s[0:4] 及 s[-3:] 的值。（3 分）
(b) 寫出 s.count("m") 及 s.count("g") 的值。（2 分）
(c) 寫出 s[::-1] 的結果。（1 分）
(d) 說明如何用切片判斷 s 是否回文（palindrome），並指出 s 是否回文。（2 分）

答案：

(a) len(s) = 11；s[0:4] = Prog（索引 0 至 3）；s[-3:] = ing（最後三個字元）。（3 分）
(b) s.count("m") = 2；s.count("g") = 2。（注意大小寫有別，"Programming" 的首字母 P 為大寫，但 m 及 g 均為小寫，故兩者都數得到。）（2 分）
(c) s[::-1] = gnimmargorP。（1 分）
(d) 判斷 s == s[::-1] 即可；把字串反轉後與原字串比較，相等即回文。本例 gnimmargorP 與 Programming 不同，故不是回文。（2 分）

丙部：算法效率、除錯與搜尋實作（40 分）

第 9 – 12 題：嵌套迴圈的時間複雜度、程式除錯與邏輯錯誤、二分搜尋完整追蹤、綜合應用程式

9.

以下程式計算嵌套迴圈的執行次數：

```
total = 0
for i in range(n):
    for j in range(i):
        total = total + 1
```

(a) 當 n = 120 時，total 的值是多少？（3 分）

(b) 以 n 表示 total 的通式，並說明推導過程。（2 分）

(c) 寫出這段程式的時間複雜度；另有一段 for k in range(3 * n) 的單層迴圈，其時間複雜度為何？（3 分）

(d) 當 n 由 120 增至 1,200，執行次數約增加多少倍？（2 分）

答案：

(a) 外層 i 由 0 至 119，內層執行 i 次，總次數 = 0 + 1 + ⋯ + 119 = 119 × 120 ÷ 2 = **7140**。（3 分）

(b) 通式為 n(n - 1) ÷ 2，即等差數列 0 + 1 + ⋯ + (n - 1) 之和。（2 分）

(c) 嵌套迴圈的時間複雜度為 $O(n^2)$；單層 for k in range(3 * n) 執行 3n 次，為 O(n)（常數因子不計）。（3 分）

(d) n = 1,200 時次數 = 1,199 × 1,200 ÷ 2 = 719400，比值 = 719400 ÷ 7140 ≈ **101** 倍，即 n 增大 10 倍，工作量約增大 100 倍。（2 分）

10.

以下程式目的為計算 2 至 20（包括 20）所有偶數之和，但程式有錯：

```
total = 0
i = 1
while i <= 20:
    if i % 2 == 1:
        total = total + i
    i = i + 1
print(total)
```

(a) 寫出程式實際輸出的值。（2 分）
(b) 寫出正確答案應為多少。（2 分）
(c) 指出錯誤所在並寫出修正後的一行。（3 分）
(d) 若刪去 i = i + 1 一行，程式會出現什麼問題？（3 分）

答案：

(a) 程式把 i % 2 == 1（奇數）的數累加，即 1 + 3 + ⋯ + 19 = 100。（2 分）
(b) 2 至 20 的偶數和 = 2 + 4 + ⋯ + 20 = 110。（2 分）
(c) 錯誤在條件：應為 if i % 2 == 0:（餘 0 才是偶數）。原式 i % 2 == 1 只挑奇數。修正後一行：if i % 2 == 0:。（3 分）
(d) 刪去 i = i + 1 後 i 永遠為 1，條件一直成立，形成無限迴圈（infinite loop），程式不會結束並可能耗盡記憶體或掛起；此為執行時出現的邏輯錯誤，編譯器不會指出。（3 分）

11.

以下列表已按數值由小至大排序，程式以二分搜尋查找 key：

```
ids = [12, 18, 23, 27, 31, 38, 44, 50, 56, 62, 69, 75, 81, 88, 94]
```

(a) 當 key 為 75 時，逐步寫出 low、high、mid 及比較結果，並寫出最終索引。（5 分）
(b) 上述查找共比較了多少次？（2 分）
(c) 當 key 為 70（不存在於列表中）時，寫出最後一次迭代的 low 及 high，並解釋迴圈如何結束。（3 分）

答案：

(a) low = 0，high = 14，mid = 7 → ids[7] = 50，50 小於 75，low = 8；low = 8，high = 14，mid = 11 → ids[11] = 75，相等，找到，索引 11。（5 分）
(b) 共比較 2 次。（2 分）
(c) key = 70：mid = 7（50 小於 70）→ low = 8；mid = 11（75 大於 70）→ high = 10；mid = 9（62 小於 70）→ low = 10；mid = 10（69 小於 70）→ low = 11；此時 low = 11 大於 high = 10，while 條件不成立，迴圈結束並回報找不到，共 4 次比較。（3 分）

12.

某程式的輸入為 8 位學生的分數：

```
scores = [48, 67, 55, 90, 73, 38, 81, 59]
```

(a) 計算平均分，準確至小數後一位。（3 分）
(b) 計算及格（50 分或以上）的人數。（2 分）
(c) 計算高於平均分的人數。（2 分）
(d) 寫出完整程式，輸出平均分（一位小數）、及格人數及高於平均分人數。（3 分）

答案：

(a) 總分 = 48 + 67 + 55 + 90 + 73 + 38 + 81 + 59 = 511；平均 = 511 ÷ 8 = 63.875 → **63.9**。（3 分）
(b) 50 分或以上者為 67、55、90、73、81、59，共 **6** 人。（2 分）
(c) 高於 63.875 者為 67、90、73、81，共 **4** 人。（2 分）
(d) 程式：scores = [...]；avg = sum(scores) / len(scores)；passes = 0；above = 0；for s in scores:； if s 不小於 50: passes += 1； if s 大於 avg: above += 1；print(round(avg, 1), passes, above)，輸出 **63.9 6 4**。（3 分）
