1121 NTOUCSE 程式設計 1C 期中考參考答案

姓名：_____ 系級：_____ 學號：_____ 112/10/31 (二)

考試時間：**13:20 – 16:00**

考試規則：1. **請闔上課本**，**不可**參考任何文件包括小考、作業、實習、或是其它參考資料
2. 你可以在題目卷上直接回答，可以使用鉛筆，但是**請在答案卷上寫下題號，並且註明在題目卷上**
3. 有需要的話，可以使用沒有教過的語法，但是**僅限於 C 語言**
4. 程式撰寫時請寫完整的程式碼，寫… 的分數很低 (程式裡有重複很多遍的敘述本來就是扣分的)
5. **不可**使用電腦、平板、電子紙、智慧手機、手錶、及工程型計算機
6. 請**不要**左顧右盼！請勿討論！請勿交換任何資料！對於題目有任何疑問請舉手發問
7. 如果你提早交卷，請迅速安靜地離開教室，請勿在走廊喧嘩
8. 違反上述考試規則視為不誠實的行為，由學校依學務規章處理
9. 請在**題目卷**及**答案卷**上都**寫下姓名及學號**，交卷時請繳交**題目卷**及**答案卷**

1. 請寫出下面程式片段執行的結果：

(a) [5]
```
01 int x=3, y=6;
02 x += y-->5 && x%2==1;
03 printf("x=%d y=%d", x, y);
```

Sol:

x=4 y=5，第二列程式先執行 y>5 的比較，得到結果 1，然後因為&&是一個 sequence point, 所以會執行 y—的動作使得 y 變數的內容為 5，然後執行 x%2 得到數值 1，比對==1，得到數值 1，此時執行 1&&1 會得到 1，最後執行 x+=1，而得到 4，儲存在變數 x 裡面。

(b) [5]
```
01 int x=3, y=5;
02 x += x/2>1 && (y*=2);
03 printf("x=%d y=%d", x, y);
```

Sol:

x=3 y=5，第二列先執行 x/2，整數除法得到商為 1，比較 1>1 得到 0 的結果，此時&&之後的(y*=2)會被跳過，因為 0&&任何數值都為 0，最後執行 x+=0 的動作，所以發現變數 x 和變數 y 的數值都沒有變。

(c) [5]
```
01 int i, a[] = {1, 3, 5, 7, 2, 4, 6, 0};
02 for (i=2; i<5; i++)
03       printf("%d ", a[a[a[i]]]);
```

Sol:

2 1 4，在迴圈重複三次過程中各個變數儲存的資料有如下的改變：

迴圈控制變數 i : 2 3 4

a[i] : 5 7 2

a[a[i]] : 4 0 5

a[a[a[i]]]: 2 1 4

(d) [5]
```
01 int i, a[] = {1, 3, 5, 7, 2, 4, 6, 0};
02 int j, t, n = sizeof(a)/sizeof(int);
03 for (i=0; i<n-1; i++) {
04       for (j=n-1; j>i; j--)
05             if (a[j-1]<a[j])
06                   t=a[j], a[j]=a[j-1], a[j-1]=t;
07       printf("%d ", a[i]);
08}
```

Sol:

7 6 5 4 3 2 1，這是一個由大排到小的氣泡排序法，外層迴圈重複 n-1 次，每次內層迴圈都把 [n-1,i] 範圍裡面最大的元素移動到最前端，所以列印的時候就會由大到小依序印出 n-1 個元素，最小的一個元素 0 並沒有列印出來。

(e) [5]
```
01 int f(int a[], int i, int j) {
02         if (i>j)
03                 return 0;
04         return f(a, i, j-2)*8 + a[j];
05 }
06
07 int main() {
08         int a[] = {1, 3, 5, 7, 2, 4, 6, 0};
09         printf("%d", f(a, 2, 7));
10         return 0;
11 }
```

Sol:

480，這是一個把陣列第 i 個元素到第 j 個元素裡面間隔一個元素當成一個 8 進位數字，並且運用遞迴函式 f()計算出對應的十進位數值，以上面程式來說，把 a 陣列由第 7 個元素反序到第 2 個元素中每隔一個列出即為 a[3], a[5], a[7]，也就是把 740 當成一個 8 進位數字，遞迴函式 f(a,2,7) 會呼叫 f(a,2,5) 然後把回傳值乘 8 加上 a[7]，也就是 f(a,2,7)=f(a,2,5)*8+a[7]= (f(a,2,3)*8+a[5])*8+a[7]=((f(a,2,1)*8+a[3])*8+a[5])*8+a[7]=((0*8+7)*8+4)*8+0=480

2. 請閱讀下列程式並回答相關問題

```
01 #include <stdio.h>
02
03 int balance = 0;
04 void save(int amount) {
05         balance += amount;
06         printf("%d ", balance);
07 }
08
09 int main() {
10         int i, balance=0;
11         for (i=30; i<100; i+=30)
12                 save(i);
13         printf("balance=%d\n", balance);
14         return 0;
15 }
```

(a) 請問程式中哪個變數是全域變數? [2] 全域變數有什麼特性?[3]

Sol:

第 03 列的 balance 整數變數是全域變數，全域變數從定義的地方開始，可以被整個檔案中所有的函式存取，甚至可以被整個執行程式中所有原始程式檔案中的函式存取，除非變數名稱被區域變數遮蓋起來時沒有辦法用到，但是變數的內容在程式執行過程中是一直存在的。

(b) [5] 請問這個程式會印出什麼數值?

Sol:

30 90 180 balance=0↵

(c) [5] 第 13 列為什麼印不出第 6 列最後一次印出的數值? 該如何修改 main() 函式裡面的變數定義使得第 13 列印出第 6 列最後一次印出的數值?

Sol:

第 13 列印的 balance 變數是第 10 列定義的區域變數，第 6 列印的 balance 變數是第 3 列定義的全域變數，由於第 10 列定義的區域變數的名稱和第 3 列定義的全域變數名稱相同，都是 balance，所以發生遮蓋的效應，在 main()函式裡面只能用到第 10 列定義的區域變數 balance。如果要讓第 13 列印的 balance 變數和第 6 列印的 balance 變數相同，要移掉第 10 列的區域變數 balance 的定義。

(d) [5] 請修改 (c) 的答案，以函式參數及區域變數修改此程式以避免使用全域變數的語法?

Sol:

```
01 void save(int* balance, int amount) {
02        *balance += amount;
03        printf("%d ", *balance);
04 }
05
06 int main() {
07        int i, balance=0;
08        for (i=30; i<100; i+=30)
09              save(&balance, i);
10        printf("balance=%d\n", balance);
11        return 0;
12 }
```

(e) [5] 請問此程式使用全域變數最大的好處為何?

Sol:

使用第三列定義的全域變數 balance 可以使得函式 save()不需要定義參數，可以少打一些字，另外也使得 save()使用到的 balance 變數裡面的內容可以在離開 save()函式之後還能夠保存下來，變成 save()函式的狀態。

(f) [5] 請問為什麼職場中大家都不建議使用全域變數?

Sol:

全域變數因為讓所有的函式裡面都可以直接存取，使得所有的程式藉由這些全域變數耦合在一起，程式的複雜度快速升高，使得程式閱讀、偵錯、以及維護時困難度大增，尤其一個專案程式由多人合作開發時，這些變數的內容有可能在沒有察覺的情況下被別人維護的程式不小心改掉，造成維持軟體正確運作的困難。另外全域變數常常隱藏了呼叫函式時候的資料流，呼叫一個函式執行一些動作，但是由函式的呼叫敘述看不出來是什麼資料被處理到了，使得程式的閱讀困難度和維護成本大大提高。

3. [15] 下列程式中二維整數陣列 int points[10][2]; 存放的是平面上多個點的 x 座標和 y 座標，請完成下列程式運用 stdlib 中的 qsort 函式將 points 陣列中的資料先依照 x 座標由大到小排序，x 座標相等時再依照 y 座標由小至大排序 (在答案卷上請標示程式列號作答，如果寫在題目卷上請在答案卷上註明)

```
01 #include <stdio.h>
02 #include <stdlib.h>
03
04 int compare(const void * a, const void * b) {
05        int *a1 = (int *)a, *b1=(int *)b;
06        return a1[0]==b1[0] ? a1[1]-b1[1] : b1[0]-a1[0];
07 }
```

*a1==*b1　　*(a1+1)-*(b1+1)　　*b1-*b1

```
08
09 int main() {
10      int points[][2] = {{40,17}, {10,19}, {25,35}, {40,21}, {40,15}, {20,11}, {25,16}};
11      int i, n=sizeof(points)/sizeof(int[2]);
12      qsort(___points___, ___n___, ___sizeof(int[2])___, compare);
13      for (i=0; i<n; i++)
14          printf("%d,%d\n", points[i][0], points[i][1]);
15      return 0;
16 }
```

sizeof(int)*2

程式執行所列印的結果如下：

```
40, 15
40, 17
40, 21
25, 16
25, 35
20, 11
10, 19
```

4. [20] 一個以 b 進位表示的 d 位數整數 $N=(n_1n_2n_3...n_d)_b, n_i \in \{0,1,2,...,b-1\}$，如果滿足 $N=n_1^{\,d}+n_2^{\,d}+...+n_d^{\,d}$，我們稱它為自戀數，例如 10 進位整數 $(153)_{10}=1^3+5^3+3^3$、3 進位整數 $(122)_3=1^3+2^3+2^3$、或是 5 進位整數 $(3134)_5=3^4+1^4+3^4+4^4=419$ 都是自戀數，10 進位整數 $(1321)_{10}$ 並不是自戀數因為 $1^4+3^4+2^4+1^4=99\neq1321$。請完成下列程式判斷一個數字是否為自戀數，程式的輸入有多列，每一列是一筆測資，包含一個整數 b 代表進位制 (b<=10)，以及一個 d 位元的數字 (d<100)，是的話請輸出字串 YES，否則輸出字串 NO (在答案卷上請標示程式列號作答，如果寫在題目卷上請在答案卷上註明)

```
01 #include <stdio.h>
02
03 int main() {
04      int b, i, j, d, z, zd, sum, x;
05      char num[100];
06      while (2==scanf("%d%s", &b, num)) {
07          d=0; while (num[d]__!=__0) ___d++___; // 計算有幾位數
08          for (x=sum=i=0; i<d; i++) {
09              z = num[i]-__'0'__;
10              x = x*__b__ + __z__; // 計算 b 進位數字之數值 x
11              for (zd=z,j=1; __j<d__; j++) zd *= z; // 計算出 zd 為每一位數的 d 次方
12              sum += __zd__; //計算每一位數 d 次方的總和
13          }
14          printf(__sum==x__ ? "YES\n" : "NO\n");
15      }
16      return 0;
17 }
```

上面第 11 列是一個計算 z 的 d 次方的迴圈，需要執行 d-1 次，效率不好，可以用下面執行 $\log d$ 次的迴圈取代

```
11              for (zd=1, dp=d; dp>0; dp/=2, z*=__z__)
12                  if (__dp % 2__) zd *= z;
```

其中 z 的 d 次方仍然紀錄在變數 zd 中，迴圈執行的次數運用 dp 來控制，一開始數值設為想要計算的次方數 d，迴圈每執行一次就會除以 2，以 d=13, z=3 為例，目標是計算 $z^d = 3^{13} = 3^{2^0\cdot 1+2^1\cdot 0+2^2\cdot 1+2^3\cdot 1} = (3)^1\cdot(3^2)^0\cdot((3^2)^2)^1\cdot(((3^2)^2)^2)^1$，所以此迴圈就像是把次方數 13 轉為 2 進位，一位數一位數由最低位數開始處理，配合逐步計算出來的權重 3，3^2，$(3^2)^2$，$((3^2)^2)^2$, … 來計算 3^{13}

5. (a) [5] 請解釋為甚麼下列程式列印的結果是 17000000.000000 而不是 17000001.000000？

```
01 float x=17000001;
02 printf("x=%f\n", x);
```

Sol:

浮點數內部的表示方法是科學記號的方法，由正負號、指數、及小數部份組成，標準格式是 IEEE 754 所規範的，又分為 32 位元的單精準度格式以及 64 位元的倍精準度格式，float 為單精準度格式，其中小數部份只有 23 位元，有效位數大約十進位 7～8 位，也就百分誤差大約在 10^{-8}，以 17000001 這個十進位數字來說，絕對誤差約為 2，個位數已經沒有辦法精確表示 17000001 這個數字，最接近而可以表示的數字為 17000000，當程式寫 float x = 17000001; 時，常數 17000001 是一個倍精準浮點數，但是儲存到變數 x 裡面時，只能存放最接近的數字，也就是 17000000，所以列印出來的數字是 17000000.000000

(b) [5] 接上題，請問下列程式片段列印的結果是什麼？迴圈主體共執行了幾次？為什麼？

```
01 float x;
02 int y;
03 for (y=x=17000000; x <17000010; x+=1)
04       if (!y--) break;
05 printf("x=%.0f y=%d\n", x, y);
```

Sol:

x=17000000 y=-1，迴圈共執行了 17000001 次，由題(a)知道 x+=1 這個敘述不會修改變數 x 的數值，這個迴圈因為執行到 y 為 0，break 敘述才結束了迴圈的執行，如果沒有 04 這一列以及 y 變數內容的改變，這個迴圈會是一個無窮迴圈，也就是說這個迴圈目前是由整數變數 y 來控制迴圈的執行次數。