APCS

交通大學資訊工程系 施文岳

Outline

- APCS 程式觀念題
  - C Subset & 練習
- APCS 程式設計實作
  - 範例練習

APCS 程式觀念題

APCS 程式觀念題[1]

- 以思維、解題和設計觀念為主
  - 包含coding tracing, code completion, and code debugging
- 出題以C Subset語法為主
  - Programming Concepts
  - Data types, constant, variable, Global, Local
  - Control structures
  - Loop structures
  - Functions
  - Recursion
  - Array and Structures
- 歷年試題
  - https://apcs.csie.ntnu.edu.tw/index.php/samplequestions/previousexam

1. https://apcs.csie.ntnu.edu.tw/index.php/samplequestions/questionstypes

C與Python的比較 - 1

- 語法風格
  - C用大括號及分號來界定區段和斷句，Python則是利用縮排

C	Python

if (a == b){
  a = a+1;
}
	if a==b:
	  a = a + 1

C與Python的比較 - 2

- 程式語法與定義變數
  - C與Python所提供的工具（ex. for, if, …)基本上意義大致相同，但語法規則不盡相同
  - C語言中，變數需要先定義型態才能使用

C	Python

int a = 10;
int i;
for(i= 0; i < 10; i++){
  a = a + 1;
}
	a = 10
	for i in range(0, 10):
	  a = a + 1

C與Python的比較 - 3

- 輸出輸入
  - C的變數輸出與輸入必須說明資料格式，而python不需要
    - printf & scanf的參數
      - %c: 字元
      - %d: 整數
      - %s: 字串
      - %f: 浮點數
      - …

C	Python
printf(“Hello World!”); char * abc = “Hello World!”; printf(“%s\n”, abc);	print (‘Hello World!’) abc = “Hello World!” print (abc)

C與Python的比較 - 4

- 註解
  - C的註解
    - 單行以//表示
    - 多行以/* … */表示
  - Python的註解
    - 單行以#表示
    - 多行以’’’ … ’’’ 表示

C	Python
// I’m a comment. /*These are comments.*/	# I’m a comment. ’’’These are comments.’’’

基礎練習

19. 下列程式碼是自動計算找零程式的一部分，程式碼中三個主要變數分別為 Total (購買總額)，Paid (實際支付金額)，Change (找零金額)。但是此程式片段有冗餘的程式碼，請找出冗餘程式碼的區塊。

(A) 冗餘程式碼在A區
(B) 冗餘程式碼在B區
(C) 冗餘程式碼在C區
(D) 冗餘程式碼在D區

```
int Total, Paid, Change;
      …
Change = Paid - Total;
printf ("500 : %d pieces\n", (Change-Change%500)/500);
Change = Change % 500;

printf ("100 : %d coins\n", (Change-Change%100)/100);
Change = Change % 100;

// A 區
printf ("50 : %d coins\n", (Change-Change%50)/50);
Change = Change % 50;

// B 區
printf ("10 : %d coins\n", (Change-Change%10)/10);
Change = Change % 10;

// C 區
printf ("5 : %d coins\n", (Change-Change%5)/5);
Change = Change % 5;

// D 區
printf ("1 : %d coins\n", (Change-Change%1)/1);
Change = Change % 1;
```

基礎練習

19. 下列程式碼是自動計算找零程式的一部分，程式碼中三個主要變數分別為 Total (購買總額)，Paid (實際支付金額)，Change (找零金額)。但是此程式片段有冗餘的程式碼，請找出冗餘程式碼的區塊。

(A) 冗餘程式碼在A區
(B) 冗餘程式碼在B區
(C) 冗餘程式碼在C區
(D) 冗餘程式碼在D區

```
int Total, Paid, Change;
    …
Change = Paid - Total;
printf ("500 : %d pieces\n", (Change-Change%500)/500);
Change = Change % 500;
printf ("100 : %d coins\n", (Change-Change%100)/100);
Change = Change % 100;

// A 區
printf ("50 : %d coins\n", (Change-Change%50)/50);
Change = Change % 50;

// B 區
printf ("10 : %d coins\n", (Change-Change%10)/10);
Change = Change % 10;

// C 區
printf ("5 : %d coins\n", (Change-Change%5)/5);
Change = Change % 5;

// D 區
printf ("1 : %d coins\n", (Change-Change%1)/1);
Change = Change % 1;
```

找的錢扣掉面額的餘數後再除以面額

C Subset

資料型態 (Date Types)

- 在C語言內, 資料型態如以下所示
  - 基礎型態
    - 數字: (unsigned) int, float, double, long
    - 字元 (character): char, string (1-D char array)
  - 陣列 array
    - Type name[size];

Index	0	1	2	3	4	5	6	7	8	9
Value	1	3	5	7	9	11	13	15	17	19

int abc[10] = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19}

abc[0] = 1, abc[5] = 11

資料型態 (Date Types)

- 在C語言內, 資料型態如以下所示
  - 基礎型態
    - 數字: int, float, double, long
    - 字元 (character): char, string (1-D char array)
      - C裡面用單引號'a'為字元, 雙引號為字串 "a"
      - char a[3] = "abc"; a[0] = 'a'; a[1] = 'b'; a[2] = 'c'; a[3] = '\0';
  - 陣列 array
    - Type name[row][col];

	0	1	2
0	1	2	3
1	4	5	6

int abc[3][3] = {1, 2, 3, 4, 5, 6}; abc[0][1] = 2, abc[1][1] = 5

資料型態 (Date Types)

- 指標 pointer & 取址 reference
  - Pointer (*): 指向記憶體位置取出值, e.g. type *name;
  - Reference (&): 取出變數的記憶體位址

```c
#include<stdio.h>
int main(){
int a = 10;
int *b = &a;
int **c = &b;
printf("Results: \n");
printf("a is located at %p.\n", &a);
printf("a is %d.\n", a);
printf("b is %p.\n", b);
printf("*b is %d.\n", *b);
printf("b is located at %p.\n", &b);
printf("**c is %d.\n", **c);
printf("*c is %p\n", *c);
printf("c is %p.\n", c);
}
```

```
Results:
a is located at 0x7ffcfd34fa3c.
a is 10.
b is 0x7ffcfd34fa3c.
*b is 10.
b is located at 0x7ffcfd34fa40.
**c is 10.
*c is 0x7ffcfd34fa3c
c is 0x7ffcfd34fa40.
```

Pointer練習題

16. 右列程式片段中，假設 a, a_ptr 和 a_ptrptr 這三個變數都有被正確宣告，且呼叫 G() 函式時的參數為 a_ptr 及 a_ptrptr。G() 函式的兩個參數型態該如何宣告？

1. (a) *int, (b) *int
2. (a) *int, (b) **int
3. (a) int*, (b) int*
4. (a) int*, (b) int**

```
void G (  (a)   a_ptr,  (b)  a_ptrptr) {
   …
}

void main () {
    int a = 1;
    // 加入 a_ptr, a_ptrptr 變數的宣告
   …
    a_ptr = &a;
    a_ptrptr = &a_ptr;
    G (a_ptr, a_ptrptr);
}
```

資料型態 (Date Types)

- 結構 (Structure)
  - 屬於一種複合性的資料結構，在同一個結構下可以包含不同屬性的變數
  - struct struct_name {

```
type name1;
type name2;
...
} variable_name;
```

    - E.g. 一個學生的資訊可能包含著：姓名，年齡，性別，身高，體重

```c
#include<stdio.h>
int main(){
    struct student{
        int  year;
        char *gender;
        int  height;
        int  weight;
        char *name;
    } Tom;
    Tom.name = "Tom";
    Tom.year = 6;
    Tom.gender = "boy";
    Tom.height = 180;
    Tom.weight = 70;
    printf("Hi I'm %s. I am a %d-year-old %s.\n", Tom.name, Tom.year, Tom.gender);
}
```

```
Hi I'm Tom. I am a 6-year-old boy.
```

資料型態 (Date Types)

- typedef 宣告
  - 針對定義好的型態提供別名, e.g. typedef type variable alias_name;

```c
#include<stdio.h>
int main(){
    struct student{
        int  year;
        char *gender;
        int  height;
        int  weight;
        char *name;
    } Tom;
    Tom.name = "Tom";
    Tom.year = 6;
    Tom.gender = "boy";
    Tom.height = 180;
    Tom.weight = 70;
    printf("Hi I'm %s. I am a %d-year-old %s.\n", Tom.name, Tom.year, Tom.gender);
}
```

```c
#include<stdio.h>
int main(){
    struct student{
        int  year;
        char *gender;
        int  height;
        int  weight;
        char *name;
    };

    typedef struct student STUDENT;
    STUDENT Tom;

    Tom.name = "Tom";
    Tom.year = 6;
    Tom.gender = "boy";
    Tom.height = 180;
    Tom.weight = 70;
    printf("Hi I'm %s. I am a %d-year-old %s.\n", Tom.name, Tom.year, Tom.gender);
}
```

資料型態 (Date Types)

- #define 定義
  - 可定義常數或簡易的函式（文字替換）
    - E.g. #define PI 3.1415926, #define AREA(x, y) (x*y)

```c
#include<stdio.h>
#define AREA(x, y) (x*y)
#define PI 3.14159265
int main(){
    printf("The circle area is %f.\n", PI*AREA(5,5));
}
```

```
The circle area is 78.539816.
```

```c
#include<stdio.h>
#define minus(x, y) x-y
int MINUS(x, y){
    return x-y;
}
int main(){
    printf("%d\n", 5*minus(5, 2)-5);
    printf("%d\n", 5*(5-2)-5);
    printf("%d\n", 5*MINUS(5, 2)-5);
}
```

```
18
10
10
```

18 → 5*5-2-5

10 → 5*(5-2)-5

邏輯運算 (Logical Operators)

	C	Python
AND	(a < 5) && (b == 2)	(a < 5) and (b == 2)
OR	(c == 1) || (d > 3)	(c == 1) or (d > 3)
NOT	!(True == False)	not (True == False)

ps 1: “&”和“|”在C裡面也代表一種位元運算 (& (and)和| (or)), 例如 1100&1000 =1000, 1010 | 0101 = 1111
ps 2: “&” 在C裡面也可以拿來取出變數的位址

邏輯運算練習題

14. 假設 x, y, z 為布林(boolean)變數，且 x=TRUE, y=TRUE, z=FALSE。請問下面各布林運算式的真假值依序為何？(TRUE 表真，FALSE 表假)

- `!(y || z) || x`
- `!y || (z || !x)`
- `z || (x && (y || z))`
- `(x || x) && z`

A. T F T F
B. F F T F
C. F T T F
D. T T F T

條件判斷式

- If Statement
  - if(cond. 1){ //只能有一個
    - Expressions1;
  - }
  - else if(cond. 2){ //可以多個
    - Expressions2;
  - }
  - else{ //只能有一個
    - Expressions3;
  - }

- Switch Statement
  - switch(variable){
    - case value1:
      - Expressions1;
      - break;
    - case value2:
      - Expressions2;
      - break;
    - …
    - default:
      - Expressions3;
      - break;
  - }

variable == value1

不加會出事

Keep Going

條件判斷式

- If Statement
  - if(variable == value1){
    - Expressions1;
  - }
  - else if(variable == value2){
    - Expressions2;
  - }
  - else{
    - Expressions3;
  - }

- Switch Statement
  - switch(variable){
    - case value1:
      - Expressions1;
      - break;
    - case value2:
      - Expressions2;
      - break;
    - …
    - default:
      - Expressions3;
      - break;
  - }

If 練習題 1

16. 右側程式執行過後所輸出數值為何？

A. 11
B. 13
C. 15
D. 16

```
void main () {
   int count = 10;
   if (count > 0) {
      count = 11;
   }
   if (count > 10) {
      count = 12;
      if (count % 3 == 4) {
         count = 1;
      }
      else {
         count = 0;
      }
   }
   else if (count > 11) {
      count = 13;
   }
   else {
      count = 14;
   }
   if (count) {
      count = 15;
   }
   else {
      count = 16;
   }

   printf ("%d\n", count);
}
```

If 練習題 1

16. 右側程式執行過後所輸出數值為何？

A. 11
B. 13
C. 15
D. 16

1. Count = 10
2. Count = 11
3. Count = 0
4. Count = 16

```
void main () {
   int count = 10;          <-- 1
   if (count > 0) {
      count = 11;           <-- 2
   }
   if (count > 10) {
      count = 12;
      if (count % 3 == 4) {
         count = 1;
      }
      else {
         count = 0;         <-- 3
      }
   }
   else if (count > 11) {
      count = 13;
   }
   else {
      count = 14;
   }
   if (count) {
      count = 15;
   }
   else {
      count = 16;           <-- 4
   }

   printf ("%d\n", count);
}
```

Switch switches to if 練習題

2. 右側 switch 敘述程式碼可以如何以 if-else 改寫？

```
switch (x) {
     case 10: y = 'a';  break;
     case 20:
     case 30: y = 'b';  break;
     default: y = 'c';
}
```

(A)
```
if(x==10) y='a';
    if(x==20||x==30) y='b';
    y='c';
```
(B)
```
if(x==10) y='a';
    else if (x==20||x==30) y='b';
    else y='c';
```
(C)
```
if(x==10) y='a';
    if(x > =20 && x < =30) y='b';
    y='c';
```
(D)
```
if(x==10) y='a';
    else if(x > =20 && x < =30) y='b';
    else y='c';
```

Switch switches to if 練習題

2. 右側 switch 敘述程式碼可以如何以 if-else 改寫？

```
switch (x) {
     case 10: y = 'a';  break;
     case 20:
     case 30: y = 'b';  break;
     default: y = 'c';
}
```

(A)
```
if(x==10) y='a';
    if(x==20||x==30) y='b';
    y='c';
```
(B)
```
if(x==10) y='a';
    else if (x==20||x==30) y='b';
    else y='c';
```
(C)
```
if(x==10) y='a';
    if(x >=20 && x <=30) y='b';
    y='c';
```
(D)
```
if(x==10) y='a';
    else if(x >=20 && x <=30) y='b';
    else y='c';
```

因為case 20沒有break,
所以當case = 20跟30
做的事是一樣的

迴圈

- 在C中，迴圈有三種類型
  - While
    - while(condition){
      - Expressions;
    - }
  - Do…While
    - do{
      - Expressions;
    - }while(condition);
  - While會先檢查條件是否成立才進行後續動作
  - Do…While會先執行動作再檢查條件是否成立

While練習題 1

22. 右側 **f()** 函式執行後所回傳的值為何？

(A) 1023
(B) 1024
(C) 2047
(D) 2048

```
int f() {
   int p = 2;
   while (p < 2000) {
      p = 2 * p;
   }
   return p;
}
```

While練習題 1

22. 右側 **f()** 函式執行後所回傳的值為何？

(A) 1023
(B) 1024
(C) 2047
(D) 2048

```
int f() {
   int p = 2;
   while (p < 2000) {
      p = 2 * p;
   }
   return p;
}
```

p為2的n次方, 但p要小於2000,
所以一旦超過2000時就會跳出迴圈,
超過2000最小2^n的值為2048

While練習題 2

23. 請問右側程式，執行完後輸出為何？

(A) 241785163922925834941235 2 7
(B) 68921 43
(C) 65537 65539
(D) 134217728 6

```
int i=2, x=3;
int N=65536;

while (i <= N) {
  i = i * i * i;
  x = x + 1;
}
printf ("%d %d \n", i, x);
```

While練習題 2

23. 請問右側程式，執行完後輸出為何？

(A) 241785163922925834941235 2 7
(B) 68921 43
(C) 65537 65539
(D) 134217728 6

```
int i=2, x=3;
int N=65536;

while (i <= N) {
  i = i * i * i;
  x = x + 1;
}
printf ("%d %d \n", i, x);
```

i 為主要造成迴圈運作次數的原因, i 每次迴圈都會是3次方成長,
而65536為2的16次方, 所以3^n要大於16, n = 3, 所以迴圈要跑3次, 而x為3+3=6,
i則是2^(3^3) = 134217728

While練習題 3

23. 右側 **f()** 函式 (a), (b), (c) 處需分別填入哪些數字，方能使得 **f(4)** 輸出 2468 的結果？

(A) **1, 2, 1**
(B) **0, 1, 2**
(C) **0, 2, 1**
(D) **1, 1, 1**

```
int f(int n) {
   int p = 0;
   int i = n;
   while (i >=   (a)   ) {
      p = 10 -   (b)   * i;
      printf ("%d", p);
      i = i -   (c)   ;
   }
}
```

While練習題 3

23. 右側 **f()** 函式 (a), (b), (c) 處需分別填入哪些數字，方能使得 **f(4)** 輸出 2468 的結果？

(A) **1, 2, 1**
(B) **0, 1, 2**
(C) **0, 2, 1**
(D) **1, 1, 1**

這邊的2468是二四六八, 不是兩千四百六十八, 所以要讓p每次從10扣8->6->4->2且迴圈就要執行4次, 因此b*i為8->6->4->2, 所以i 要從4->3->2->1, c為1, a為1, b為2

```
int f(int n) {
   int p = 0;
   int i = n;
   while (i >=    (a)    ) {
      p = 10 -    (b)    * i;
      printf ("%d", p);
      i = i -     (c)    ;
   }
}
```

While 練習題 4

15. 給定右側函式 **F()**，執行 **F()** 時哪一行程式碼可能永遠不會被執行到？

(A) a = a + 5;
(B) a = a + 2;
(C) a = 5;
(D) 每一行都執行得到

```
void F (int a) {
   while (a < 10)
      a = a + 5;
   if (a < 12)
      a = a + 2;
   if (a <= 11)
      a = 5;
}
```

While 練習題 4

15. 給定右側函式 **F()**，執行 **F()** 時哪一行程式碼可能永遠不會被執行到？

(A) a = a + 5;
(B) a = a + 2;
(C) a = 5;
(D) 每一行都執行得到

```
void F (int a) {
   while (a < 10)
      a = a + 5;
   if (a < 12)
      a = a + 2;
   if (a <= 11)
      a = 5;
}
```

a跑完while迴圈後最小為10, 在第一個if判斷成立後, a會變成12, 因此第二個if判斷式永遠都是False

迴圈

- For
  - 首先對控制變數做初始化 -> 檢查判斷式是否成立 -> 執行動作 -> 執行控制變數的運算
  - for(initialization; condition; increment){
    - Expressions;
  - }

Be Careful!

```
#include<stdio.h>
int main(){
    int i = 0;
    for (i = 0; i < 5; i++){
        printf("%d\n", i);
    }
}
```

i = i + 1, i += 1

```
0
1
2
3
4
```

For練習題 1

1. 右側程式正確的輸出應該如下：

```
    *
   ***
  *****
 *******
*********
```

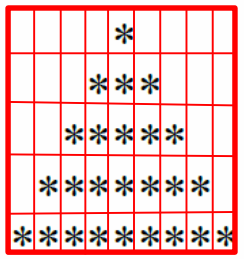
在不修改右側程式之第 4 行及第 7 行程式碼的前提下，最少需修改幾行程式碼以得到正確輸出？

(A) 1
(B) 2
(C) 3
(D) 4

```
1   int k = 4;
2   int m = 1;
3   for (int i=1; i<=5; i=i+1) {
4      for (int j=1; j<=k; j=j+1) {
5         printf (" ");
6      }
7      for (int j=1; j<=m; j=j+1) {
8         printf ("*");
9      }
10     printf ("\n");
11     k = k - 1;
12     m = m + 1;
13  }
```

For練習題 1

1. 右側程式正確的輸出應該如下：



在不修改右側程式之第 4 行及第 7 行程式碼的前提下，最少需修改幾行程式碼以得到正確輸出？

(A) 1
(B) 2
(C) 3
(D) 4

```
1   int k = 4;
2   int m = 1;
3   for (int i=1; i<=5; i=i+1) {
4      for (int j=1; j<=k; j=j+1) {
5         printf (" ");
6      }
7      for (int j=1; j<=m; j=j+1) {
8         printf ("*");
9      }
10     printf ("\n");
11     k = k - 1;
12     m = m + 1;
13  }
```

For練習題 2

1. 給定一個 1x8 的陣列 **A**， **A = {0, 2, 4, 6, 8, 10, 12, 14}**。右側函式 **Search(x)** 真正目的是找到 **A** 之中大於 **x** 的最小值。然而，這個函式有誤。請問下列哪個函式呼叫可測出函式有誤？

(A) Search(-1)
(B) Search(0)
(C) Search(10)
(D) Search(16)

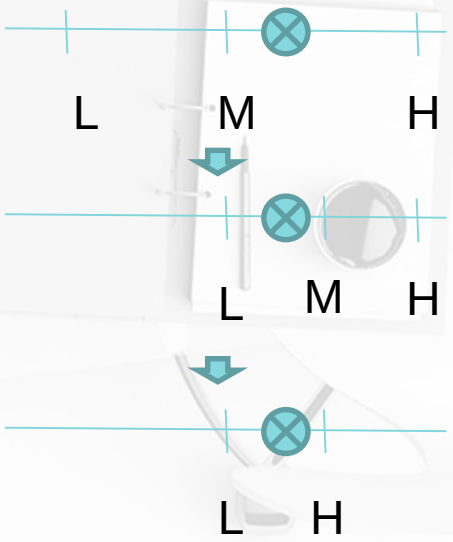
```
int A[8]={0, 2, 4, 6, 8, 10, 12, 14};

int Search (int x) {
   int high = 7;
   int low = 0;
   while (high > low) {
      int mid = (high + low)/2;
      if (A[mid] <= x) {
         low = mid + 1;
      }
      else {
         high = mid;
      }
   }
   return A[high];
}
```

For練習題 2

1. 給定一個 1x8 的陣列 **A**， **A = {0, 2, 4, 6, 8, 10, 12, 14}**。右側函式 **Search(x)** 真正目的是找到 **A** 之中大於 **x** 的最小值。然而，這個函式有誤。請問下列哪個函式呼叫可測出函式有誤？

(A) Search(-1)
(B) Search(0)
(C) Search(10)
(D) Search(16)




```
int A[8]={0, 2, 4, 6, 8, 10, 12, 14};

int Search (int x) {
   int high = 7;
   int low = 0;
   while (high > low) {
      int mid = (high + low)/2;
      if (A[mid] <= x) {
         low = mid + 1;
      }
      else {
         high = mid;
      }
   }
   return A[high];
}
```

For練習題 3

5. 若 **A** 是一個可儲存 n 筆整數的陣列，且資料儲存於 **A[0]~A[n-1]**。經過右側程式碼運算後，以下何者敘述不一定正確？

(A) **p**是A陣列資料中的最大值
(B) **q**是A陣列資料中的最小值
(C) **q< p**
(D) **A[0]<= p**

```
int A[n]={ … };
int p = q = A[0];
for (int i=1; i<n; i=i+1) {
   if (A[i] > p)
      p = A[i];
   if (A[i] < q)
      q = A[i];
}
```

For練習題 4

17. 給定右側函式 **F()**，**F()**執行完所回傳的 **x** 值為何？

A. $n(n+1)\sqrt{\lfloor log_2 n \rfloor}$

B. $\frac{n^2(n+1)}{2}$

C. $\frac{n(n+1)\lfloor log_2 n+1 \rfloor}{2}$

D. $\frac{n(n+1)}{2}$

```
int F (int n) {
   int x = 0;
   for (int i=1; i<=n; i=i+1)
      for (int j=i; j<=n; j=j+1)
         for (int k=1; k<=n; k=k*2)
            x = x + 1;
   return x;
}
```

For練習題 4

17. 給定右側函式 **F()**，**F()**執行完所回傳的 **x** 值為何？

A. $n(n+1)\sqrt{\lfloor log_2 n \rfloor}$

B. $\frac{n^2(n+1)}{2}$

C. $\frac{n(n+1)\lfloor log_2 n+1 \rfloor}{2}$

D. $\frac{n(n+1)}{2}$

```
int F (int n) {
   int x = 0;
   for (int i=1; i<=n; i=i+1)
      for (int j=i; j<=n; j=j+1)
         for (int k=1; k<=n; k=k*2)
            x = x + 1;
   return x;
}
```

執行n(n+1)/2

執行$\log_2 n+1$次

函式宣告

- 函式定義包含三大部分
  - 函式型態
  - 函式本體
  - 函式回傳值
  - type function_name(type1 para1, type2 para2, …){
    - Expressions;
    - return values;
  - }

```
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
```

```c
#include<stdio.h>
int multiplier(int x, int y){
    return x*y;
}
int ninenine(int x){
    int i = 1;
    for(i = 1; i<=9; i++){
        printf("%d * %d = %d\n", x, i, multiplier(x, i));
    }
    return 1;
}
int main(){
    ninenine(2);
}
```

Function練習題

20. 右側程式執行後輸出為何？

(A) 0
(B) 10
(C) 25
(D) 50

```
int G (int B) {
    B = B * B;
    return B;
}

int main () {
    int A=0, m=5;

    A = G(m);
    if (m < 10)
        A = G(m) + A;
    else
        A = G(m);

    printf ("%d \n", A);
    return 0;
}
```

Function練習題

20. 右側程式執行後輸出為何？

(A) 0
(B) 10
(C) 25
(D) 50

```
int G (int B) {
    B = B * B;
    return B;
}

int main () {
    int A=0, m=5;

    A = G(m);
    if (m < 10)
        A = G(m) + A;
    else
        A = G(m);

    printf ("%d \n", A);
    return 0;
}
```

G(5) = 25

A = 50

變數

- 全域變數 (Global Variable)
  - 變數宣告完後，在整份程式中都可以自由呼叫
- 區域變數 (Local Variable)
  - 變數的作用範圍有限，例如在一個函式內宣告的變數，當函式執行完變數也會跟著消失

```c
#include<stdio.h>
int a = 0;
void add(){
    int b = 10;
    printf("add result: %d + %d = %d\n", a, b, a+b);
    a = a + b;
}
int main(){
    int b = 1;
    add();
    printf("main result: %d + %d = %d\n", a, b, a+b);
}
```

Global Variable

Local Variable

Local Variable

```
add result: 0 + 10 = 10
main result: 10 + 1 = 11
```

變數練習題

8. 給定右側程式，其中s有被宣告為全域變數，請問程式執行後輸出為何?

(A) 1,6,7,7,8,8,9
(B) 1,6,7,7,8,1,9
(C) 1,6,7,8,9,9,9
(D) 1,6,7,7,8,9,9

```
int s = 1; // 全域變數

void add (int a) {
   int s = 6;
   for( ; a>=0; a=a-1) {
      printf("%d,", s);
      s++;
      printf("%d,", s);
   }
}
int main () {
   printf("%d,", s);
   add(s);
   printf("%d,", s);
   s = 9;
   printf("%d", s);
   return 0;
}
```

變數練習題

8. 給定右側程式，其中s有被宣告為全域變數，請問程式執行後輸出為何?

(A) 1,6,7,7,8,8,9
(B) 1,6,7,7,8,1,9
(C) 1,6,7,8,9,9,9
(D) 1,6,7,7,8,9,9

1. s = 1
2. s = 6
3. s = 7
4. s = 7
5. s = 8
6. s = 1
7. s = 9

```
int s = 1; // 全域變數

void add (int a) {
   int s = 6;
   for( ; a>=0; a=a-1) {
      printf("%d,", s);
      s++;
      printf("%d,", s);
   }
}
int main () {
   printf("%d,", s);
   add(s);
   printf("%d,", s);
   s = 9;
   printf("%d", s);
   return 0;
}
```

遞迴 (Recursion)

- 在程式語言中，可以透過呼叫自身函式(self-calling)來達到遞迴效果
  - 但函式中一定要有終止條件，否則程式會不斷呼叫自身函式造成記憶體錯誤(segmentation fault)

$$\begin{cases} f_0 = 0 \\ f_1 = 1 \\ f_n = f_{n-1} + f_{n-2} \end{cases}$$

```
The 0-th number of Fibonacci is 0.
The 1-th number of Fibonacci is 1.
The 2-th number of Fibonacci is 1.
The 3-th number of Fibonacci is 2.
The 4-th number of Fibonacci is 3.
The 5-th number of Fibonacci is 5.
The 6-th number of Fibonacci is 8.
The 7-th number of Fibonacci is 13.
The 8-th number of Fibonacci is 21.
The 9-th number of Fibonacci is 34.
The 10-th number of Fibonacci is 55.
```

```c
#include<stdio.h>
int Fibonacci(int n){
    if (n==0){
        return 0;
    }
    else if(n == 1){
        return 1;
    }
    else{
        return Fibonacci(n-2)+Fibonacci(n-1);
    }
}

int main(){
    int x = 0;
    for(x = 0; x <=10; x++){
        printf("The %d-th number of Fibonacci is %d.\n", x, Fibonacci(x));
    }
}
```

遞迴 (Recursion)

遞迴與迴圈的比較

```c
int Fibonacci(int n){
    if (n==0){
        return 0;
    }
    else if(n == 1){
        return 1;
    }
    else{
        return Fibonacci(n-2)+Fibonacci(n-1);
    }
}
```

Recursion Version
0.000123s

```c
int Fibonacci_loop(int n){
    int fn[3] = {0,1,0};
    int i = 0;
    if (n == 0){
        return fn[0];
    }
    else if (n == 1){
        return fn[1];
    }

    for(i=2; i<=n; i++){

        fn[2] = fn[0]+fn[1];
        fn[0] = fn[1];
        fn[1] = fn[2];
    }
    return fn[2];

}
```

Iterative Version
0.00008s

遞迴練習題 1

4. 右側函式兩個回傳式分別該如何撰寫，才能正確計算並回傳兩參數 **a**, **b** 之最大公因數 (Greatest Common Divisor)？

(A) **a**, **GCD(b,r)**
(B) **b**, **GCD(b,r)**
(C) **a**, **GCD(a,r)**
(D) **b**, **GCD(a,r)**

```
int GCD (int a, int b) {
    int r;

    r = a % b;
    if (r == 0)
       return ____________;
    return ________________;
}
```

遞迴練習題 1

4. 右側函式兩個回傳式分別該如何撰寫，才能正確計算並回傳兩參數 **a**, **b** 之最大公因數 (Greatest Common Divisor)？

(A) **a**, **GCD(b,r)**
(B) **b**, **GCD(b,r)**
(C) **a**, **GCD(a,r)**
(D) **b**, **GCD(a,r)**

a b	b
r	b r

```
int GCD (int a, int b) {
    int r;

    r = a % b;
    if (r == 0)
       return _____________;
    return _____________;
}
```

遞迴練習題 2

21. 右側 G() 應為一支遞迴函式，已知當 a 固定為 2，不同的變數 x 值會有不同的回傳值如下表所示。請找出 G() 函式中 (a) 處的計算式該為何？

a 值	x 值	G(a, x) 回傳值
2	0	1
2	1	6
2	2	36
2	3	216
2	4	1296
2	5	7776

```
int G (int a, int x) {
    if (x == 0)
        return 1;
    else
        return ____(a)____ ;
}
```

(A) ((2*a)+2) * G(a, x - 1)
(B) (a+5) * G(a -1, x - 1)
(C) ((3*a) -1) * G(a, x - 1)
(D) (a+6) * G(a, x - 1)

遞迴練習題 2

21. 右側 G() 應為一支遞迴函式，已知當 a 固定為 2，不同的變數 x 值會有不同的回傳值如下表所示。請找出 G() 函式中 (a) 處的計算式該為何？

a 值	x 值	G(a, x) 回傳值
2	0	1
2	1	6
2	2	36
2	3	216
2	4	1296
2	5	7776

```
int G (int a, int x) {
    if (x == 0)
        return 1;
    else
        return ____(a)____ ;
}
```

(A) ((2*a)+2) * G(a, x - 1) → 6
(B) (a+5) * G(a -1, x - 1) → 7
(C) ((3*a) -1) * G(a, x - 1) → 5
(D) (a+6) * G(a, x - 1) → 8

遞迴練習題 3

7. 若以 B(5,2) 呼叫右側 B() 函式，總共會印出幾次 “base case”？

(A) 1
(B) 5
(C) 10
(D) 19

```
int B (int n, int k) {
   if (k == 0 || k == n){
      printf ("base case\n");
      return 1;
   }
   return B(n-1,k-1) + B(n-1,k);
}
```

遞迴練習題 3

7. 若以 B(5,2) 呼叫右側 B() 函式，總共會印出幾次 “base case”？

(A) 1
(B) 5
(C) 10
(D) 19

```
int B (int n, int k) {
   if (k == 0 || k == n){
      printf ("base case\n");
      return 1;
   }
   return B(n-1,k-1) + B(n-1,k);
}
```

算有多少分支:

B(5, 2) -> B(4, 1)+B(4, 2) -> ~~B(3, 0)~~+B(3, 1)+B(3, 1)+B(3, 2) -> ~~B(3, 0)+B(2, 0)~~+B(2, 1)+~~B(2, 0)~~+B(2, 1)+B(2, 1)+~~B(2, 2)~~ -> ~~B(3, 0)+B(2, 0)+B(1, 0)+B(1, 1)+B(2, 0)+B(1, 0)+B(1, 1)+B(1, 0)+B(1, 1)+B(2, 2)~~

APCS 程式設計實作

Before Programing

- 輸出輸入
  - Data type, file, database
  - Format
  - Constraint
- 程式撰寫
  - 了解題目需求
  - 採用/設計合適演算法
  - 資料結構的使用

Q1. 三角形判別

- 輸入格式
  - 輸入一行包含三正整數，數字間以一個空格相隔，且三數字均小於30,001
- 輸出格式
  - 此三正整數，兩字之間以一個空白格最後 一個數字後 不應有空白
  - 第二行輸出三角形類型
    - 無法構成三角形印出”No”
    - 鈍角三角形印出”Obtuse”
    - 直角三角形印出”Right”
    - 銳角三角形印出”Acute”

提示：若a、b、c為三個線段的邊長，且c為最大值，則

若 a+b ≦ c	，三線段無法構成三角形
若 a × a+b × b < c × c	，三線段構成鈍角三角形 (Obtuse triangle)
若 a × a+b × b = c × c	，三線段構成直角三角形 (Right triangle)
若 a × a+b × b > c × c	，三線段構成銳角三角形 (Acute triangle)

Q1. 三角形判別

- 測試範例

範例一：輸入

```
3 4 5
```

範例一：正確輸出

```
3 4 5
Right
```

（說明）a×a + b×b = c×c 成立時為直角三角形。

範例二：輸入

```
101 100 99
```

範例二：正確輸出

```
99 100 101
Acute
```

（說明）邊長排序由小到大輸出，a×a + b×b > c×c 成立時為銳角三角形。

範例三：輸入

```
10 100 10
```

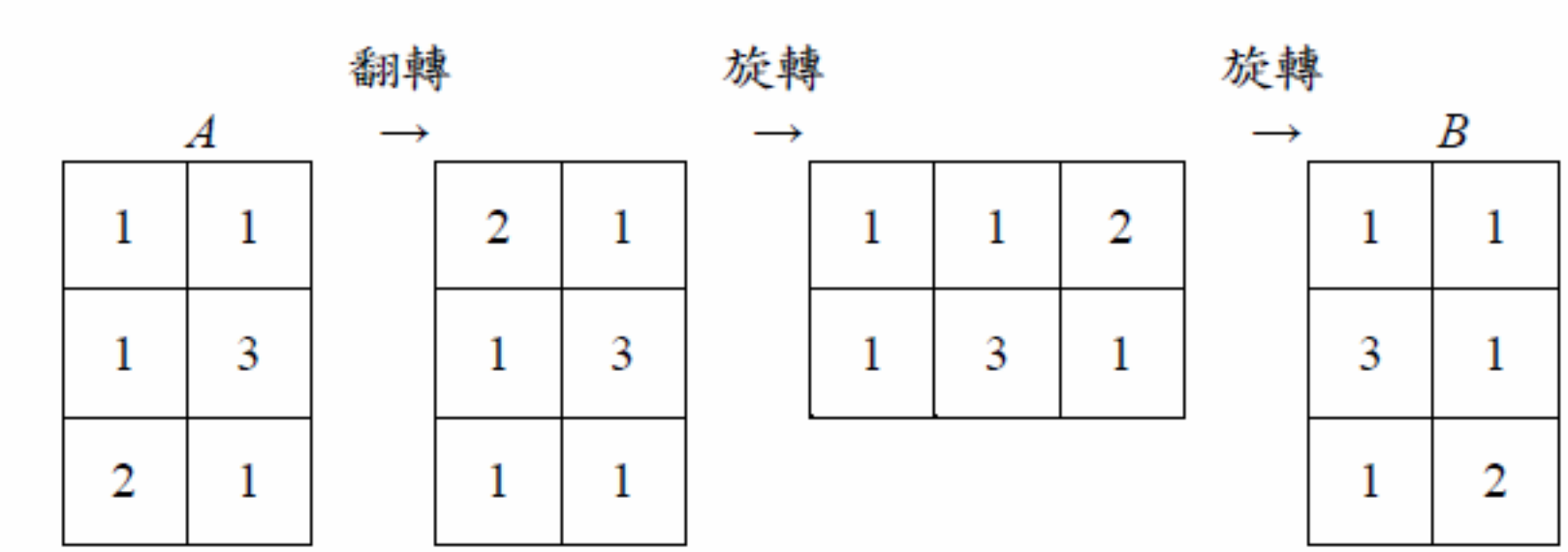
範例三：正確輸出

```
10 10 100
No
```

（說明）由於無法構成三角形，因此第二行須印出「No」。

Q2. 矩陣轉換

- 定義
  - 翻轉：第一列與最後一列交換，第二列與倒數第二列交換，以此類推
  - 旋轉：將矩陣順時針旋轉90度



Q2. 矩陣轉換

- 輸入格式
  - 第一行輸入三個均介於1到10之間的數字，分別為R, M, C
    - R代表陣列的列數
    - M代表一列有幾個數字（行數）
    - C代表要執行幾個指令
  - 第二行到第R+1行則每行輸入M個數字
  - 最後一行輸入要執行的指令，其中包含C個數字
    - 0代表旋轉
    - 1代表翻轉
  - 以上同行數字以空格分隔
- 輸出格式
  - 第一行輸出最後陣列的列數與行數（空格分隔）
  - 第二行開始輸出每一列的數值，行與行間以空格分隔

Q2. 矩陣轉換

範例一：輸入

```
3 2 3
1 1
3 1
1 2
1 0 0
```

範例一：正確輸出

```
3 2
1 1
1 3
2 1
```

（說明）

如圖二所示

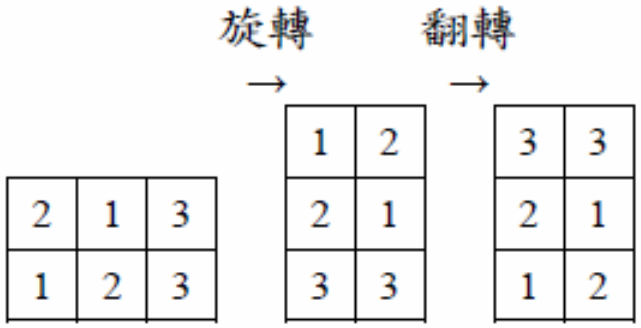
範例二：輸入

```
3 2 2
3 3
2 1
1 2
0 1
```

範例二：正確輸出

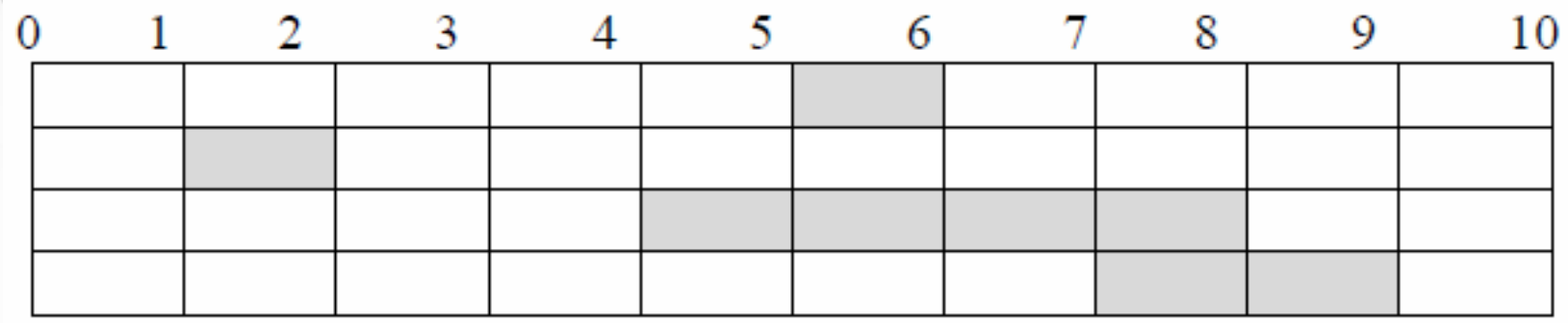
```
2 3
2 1 3
1 2 3
```

（說明）



Q3. 線段覆蓋長度

- 問題定義
  - 給定多個線段，計算出線段的總長度
  - 例如
    - (5, 6), (1, 2), (4, 8), (7, 9)
    - 覆蓋總長度為6



Q3. 線段覆蓋長度

- 輸入格式
  - 第一行為一正整數，代表有幾個線段
  - 接著下面每一行代表一組線段的起始和結尾，中間以空白區隔
- 輸出格式
  - 總長度

範例一：輸入

輸入	說明
5	此測試案例有 5 個線段
160 180	開始端點座標值與結束端點座標
150 200	開始端點座標值與結束端點座標
280 300	開始端點座標值與結束端點座標
300 330	開始端點座標值與結束端點座標
190 210	開始端點座標值與結束端點座標

範例一：輸出

輸出	說明
110	測試案例的結果

範例二：輸入

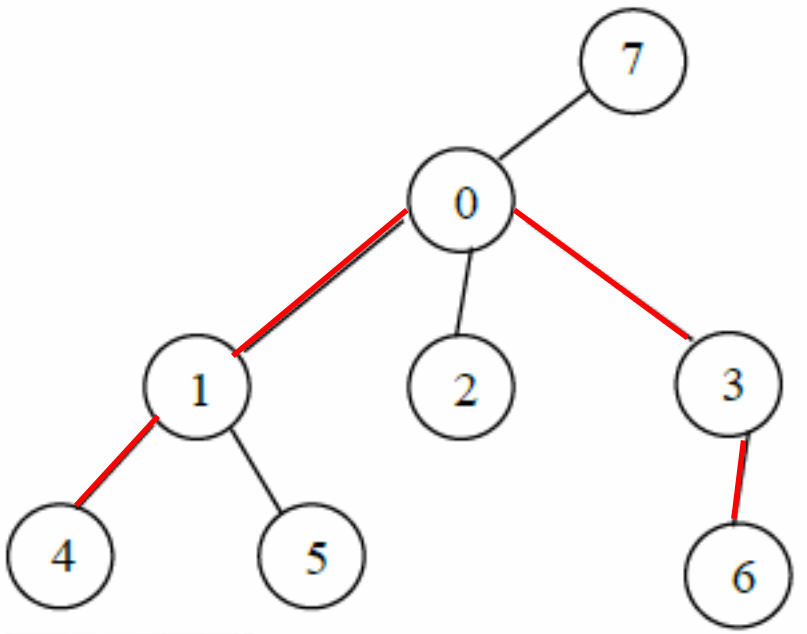
輸入	說明
1	此測試案例有 1 個線段
120 120	開始端點座標值與結束端點座標值

範例二：輸出

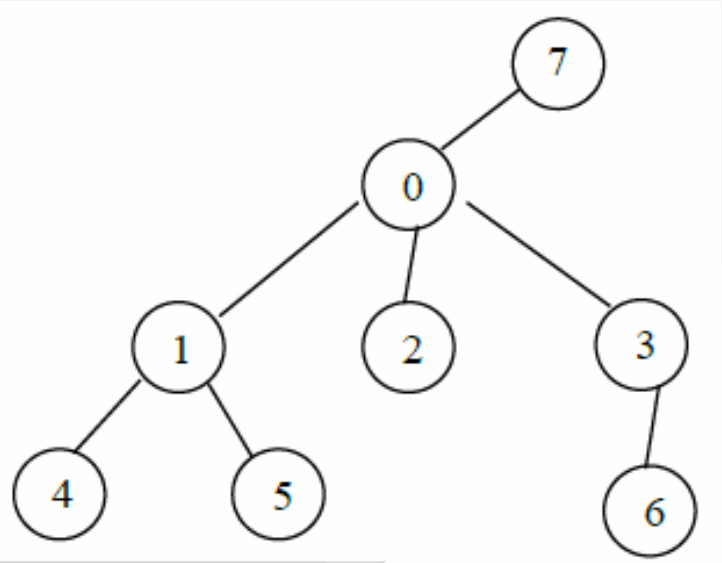
輸出	說明
0	測試案例的結果

Q4. 血緣關係

- 問題定義
  - 找出一家族中，血緣關係最遠的距離為多少
  - 假設家族關係中，只有一個人是祖先，且沒有兩個成員有同樣的小孩



Q4. 血緣關係



- 輸入格式
  - 第一行為一正整數，代表家族中的成員個數，由0~n-1表示每個成員
  - 接下來的n-1行，為親子關係，a b代表b是a的孩子
- 輸出格式
  - 最遠的血緣距離

範例一：輸入

```
8
0 1
0 2
0 3
7 0
1 4
1 5
3 6
```

範例一：正確輸出

```
4
```

（說明）

如題目所附之圖，最遠路徑為 4->1->0->3->6 或 5->1->0->3->6，距離為 4。

範例二：輸入

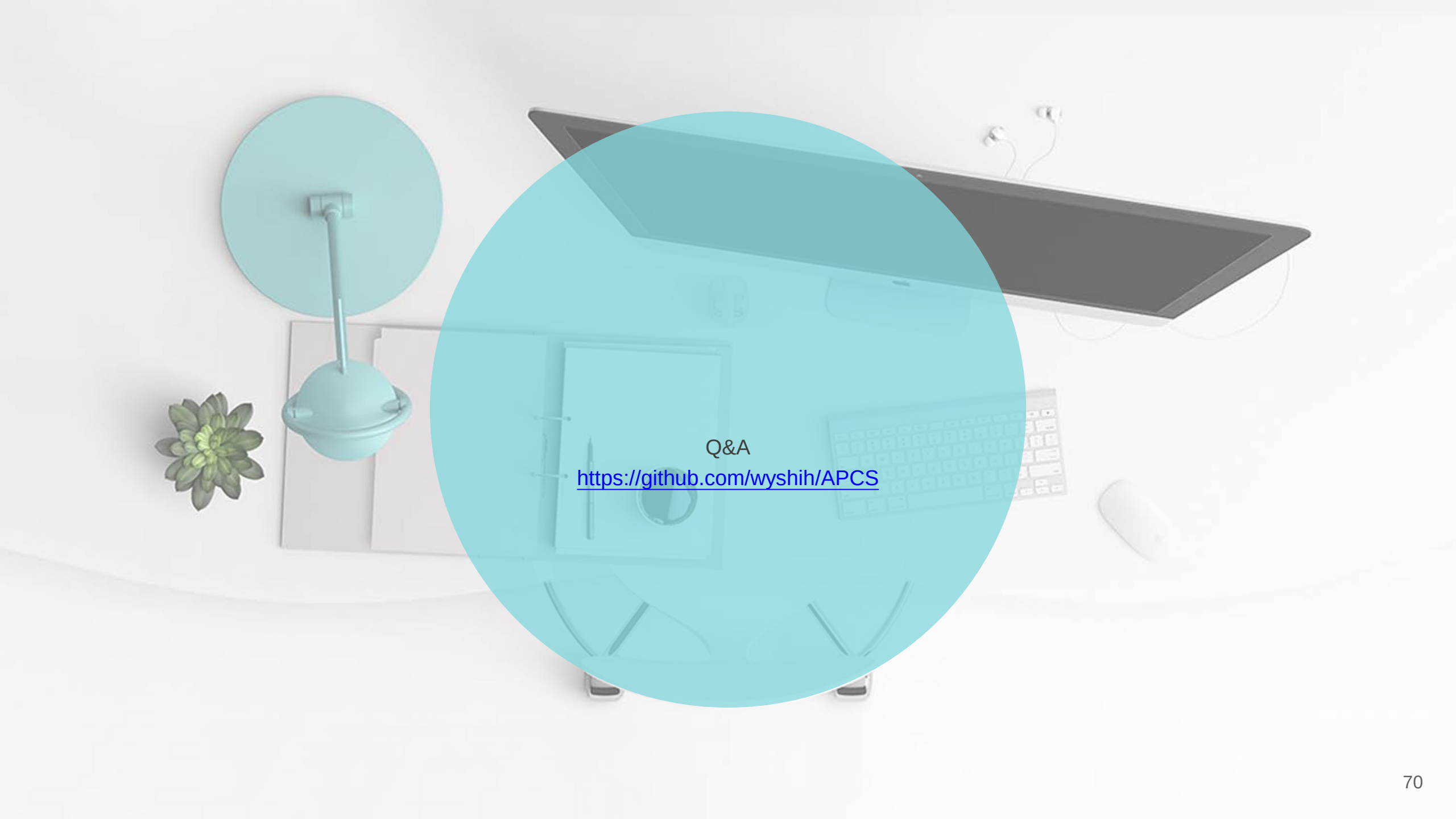
```
4
0 1
0 2
2 3
```

範例二：正確輸出

```
3
```

（說明）

最遠路徑為 1->0->2->3，距離為 3。



Q&A

https://github.com/wyshih/APCS