函數(續) + 遞迴

C語言

Alex Lu @ FCU系程

WHO AM I?

呂銘洋(Alex Lu) @ FCU 資訊二乙

SELF-INTRO

- **自由軟體**愛好者
- linux user
- 興趣
  - **程式語言**（理論/編譯技術）
  - 寫寫程式
  - 沖沖咖啡
  - 聽聽音樂
  - 看看動畫
- 有在玩

今天要講的

- ~~很多廢話~~
- 函數
  - ~~你已經會的~~
    - ~~怎麼用~~
    - ~~用別人寫好的~~
    - ~~原型宣告~~
    - ~~怎麼自訂一個~~
  - 變數
    - 區域跟全域
  - 運作
- 遞迴
- ~~**用人腦執行程式**~~

今天不講的

- 怎麼傳陣列當參數
- 怎麼回傳陣列
- 怎麼傳回兩個/多個值

複習

定義一個函數

架構

```
回傳類型 函數名子（參數類型 參數名稱）{
    做要做的事。
    return 要回傳的東西;
}
```

ex.

```
/*           ↓ 用, 分隔一組參數 */
int add(int x, int y) {
    int ans = x + y;
    return ans;
}
```

練習

寫出加減乘除的版本
分別叫作add, sub, mul, div

```c
int sub(int x, int y) {
    int ans = x - y;
    return ans;
}

int mul(int x, int y) {
    int ans = x * y;
    return ans;
}

int div(int x, int y) {
    int ans = x / y;
    return ans;
}
```

練習

只用printf跟剛剛寫的4個function印出

1 + (2 + 3)

跟

(4 * (3 + 5)) / 2

的結果

```c
int main() {
    int ans1 = add(1, add(2, 3));
    int ans2 = div(mul(4, add(3, 5)), 2);
    printf("%d", ans1);
    printf("%d", ans2);
    system("pause");
    return 0;
}
```

解釋

剛剛發生了什麼事

- 變數
- 參數

ans = ***div(mul(4, add(3, 5)), 2)***;

⇒ div(***mul(4, add(3, 5))***, 2)

⇒ mul(4, ***add(3, 5)***)

⇒ add(3, 5) → ***8***

⇒ mul(4, ***8***) → ***32***

⇒ div(***32***, 2)→ ***16***

⇒ ans = ***16***;

printf也一樣

```
int ans = div(mul(4, add(3, 5)), 2);
printf("%d", ans);
```

所以ans可以省去

```
printf("%d", div(mul(4, add(3, 5)), 2));
```

全域變數

在main外面的變數

```c
int x = 1;      /* ← 全域 */
int main() {
    int y = 2; /* ← 區域 */
    printf("%d %d", x, y);
};
```

會印出什麼？

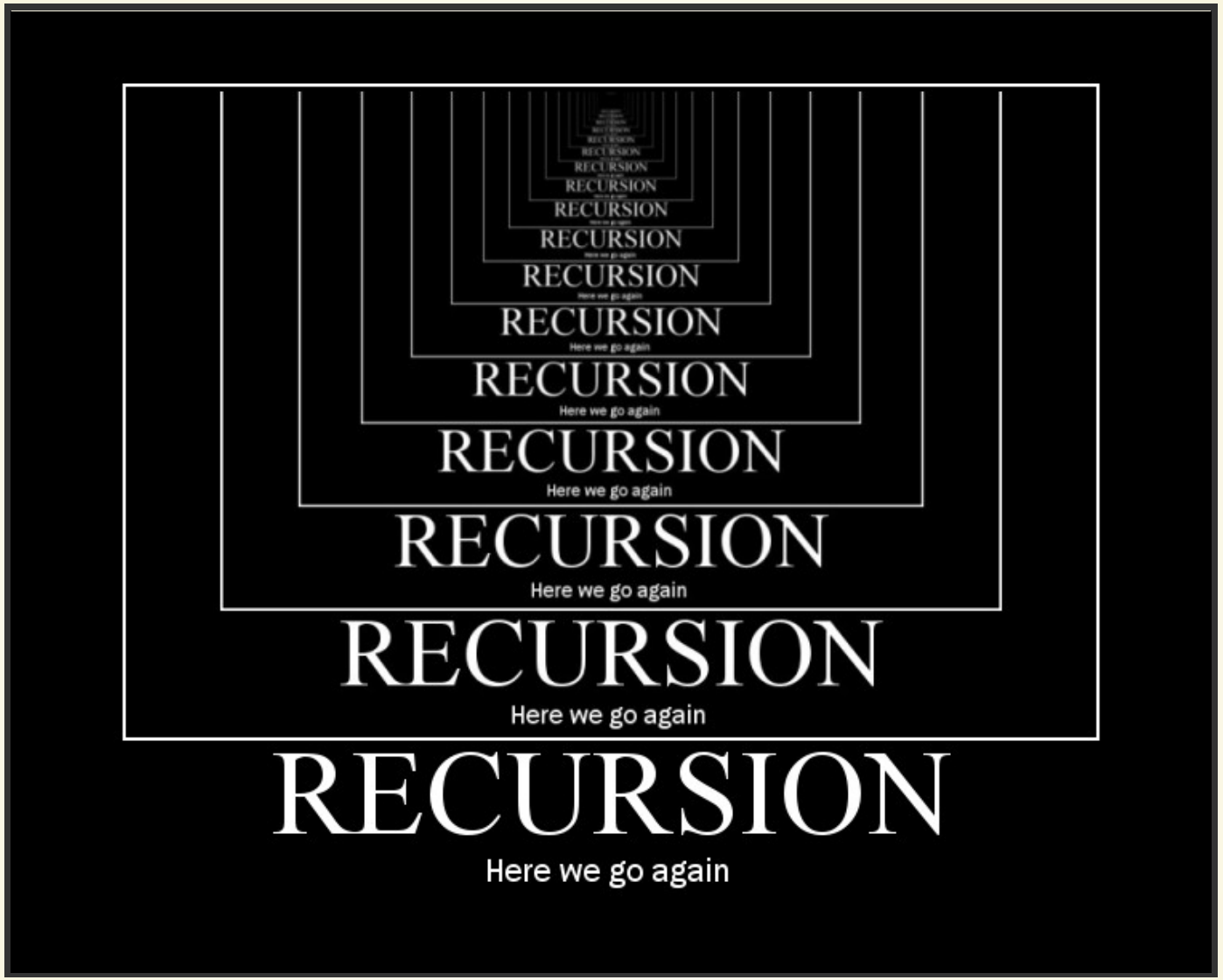
```c
int x = 1;
int f() {
    int x = 2;
    printf("%d", x);
}
```

```c
int x = 1;
int x = 2;
int f() {
    printf("%d", x);
}
```

```c
int x = 1;
int f(int x) {
    printf("%d", x);
}
/* f(1) ⇒ ? */
```

遞迴

RECURSION


RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again

維基百科
自由的百科全書

首頁
分類索引
特色內容
新聞動態
最近更改
隨機條目

條目 討論 台灣正體 汉 漢 閱讀 編輯 檢視歷

遞迴 [編輯]

維基百科，自由的百科全書

遞迴（英語：Recursion），又譯為**遞迴**，在數學與電腦科學中，是指在函式的定義中使用函式自身的方法。遞迴一詞還較常用於描述以自相似方法重複事物的過程。例如，當兩面鏡子相互之間近似平行時，鏡中巢狀的影像是以無限遞迴的形式出現的。也可以理解為自我複製的過程。


Google
遞迴
網頁
圖片
地圖
更多
搜尋工具
約有 588,000 項結果 (搜尋時間：0.18 秒)
您是不是要查： 遞迴

自己裡面包含自己

結果是？

```
void hello(void) {
    printf("hello\n");
    hello();
    return;
}
```

遞迴

- 很像迴圈
- 不如說是比迴圈更強
- 但是通常很吃記憶體

在做的事

- 回答最小的問題　　(base case)
- 把大問題化成小問題(recursive case)

EXAMPLE

階乗

$$n! = 1 \times 2 \times 3 \times 4 \times 5 \times \ldots \times n$$

TRY IT

如果用迴圈的話

```
int fact(int n) {
    int ans = 1;
    int i;
    for(i = 0; i < n; i += 1)
        ans = ans * (n + 1);
    return ans;
}
```

數學上的定義

$$n! = \begin{cases} 1 & \text{if } n = 0, \\ (n-1)! \times n & \text{if } n > 0. \end{cases}$$

計算　　結果

$5! = 5 \times 4!$ ⇒ 120

$4! = 4 \times 3!$ ⇒ 24

$3! = 3 \times 2!$ ⇒ 6

$2! = 2 \times 1!$ ⇒ 2

$1! = 1 \times 0!$ ⇒ 1

$0! = 1$

在c裏面寫成

```c
int fact(int n) {
    if (n == 0)
        return 1;
    else
        return n * fact(n - 1);
}
```

EXAMPLE(MORE)

N次方

$a^n = a \times a \times a \times \ldots \times a$ (n times)

定義

- $a^0 = 1$
- $a^n = a \times a^{n-1}$

用遞迴

```
int pow(int a, int n) {
    if (n == 0)
        return 1;
    else
        return a * pow(a, n - 1);
}
```

EXAMPLE

最大公因數

GCD(54, 24) ⇒ 6

兩種定義

$$\gcd(a, 0) = a$$
$$\gcd(a, b) = \gcd(b, a \bmod b),$$

$$\gcd(a, a) = a$$
$$\gcd(a, b) = \gcd(a - b, b) \quad \text{, if } a > b$$
$$\gcd(a, b) = \gcd(a, b - a) \quad \text{, if } b > a$$

練習

- 用遞迴的方法，計算$1 \times 3 \times 5 \times \ldots \times n$
- 用遞迴的方式，判斷x是奇數還是偶數

```
int odd_mul(int n) {
    if (!n % 2)
        return 0;
    if (n == 1)
        return 1;
    else
        return n * odd_mul(n - 2);
}
```

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

EXAMPLE(TREE)

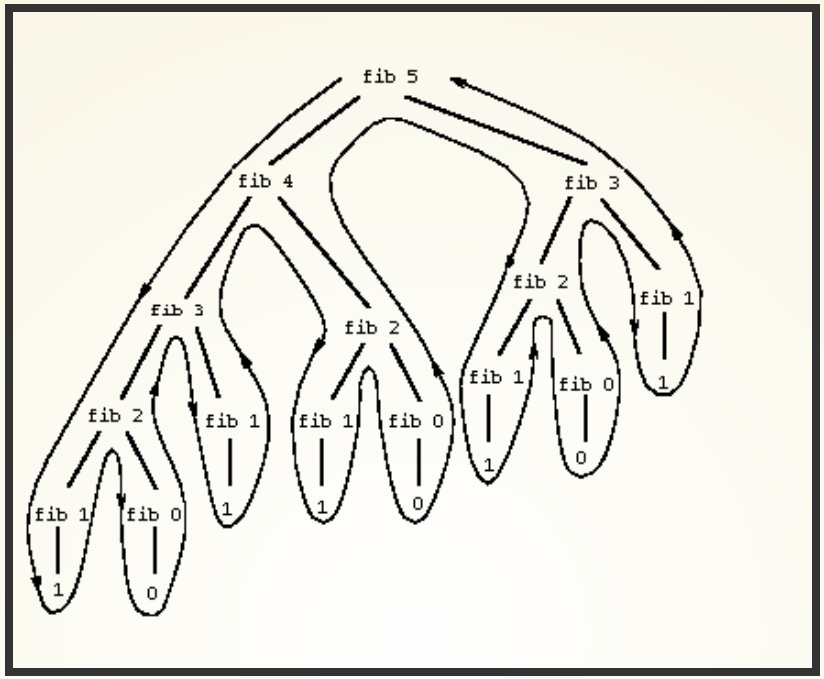
費氏數列

1, 1, 2, 3, 5, 8, 13, 21, ...

定義

- fib(0) = 0
- fib(1) = 1
- fib(n) = fib(n - 1) + fib(n - 2)

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


fib 5
fib 4
fib 3
fib 3
fib 2
fib 2
fib 1
fib 2
fib 1
fib 0
fib 1
fib 1
fib 0
fib 1
1
0
1
1
0
1
0
1

小進階

利用剛剛寫的fact跟pow

三、請設計一遞迴程式，計算下列數學公式

$$f(x,n)=\sum_{k=1}^{n}\frac{x^k}{k!}=\frac{x^1}{1!}+\frac{x^2}{2!}+\frac{x^3}{3!}+\ldots+\frac{x^n}{n!}$$

註：1) 數列相加、次方、以及階層計算均請用遞迴函式完成。

```
int f(int x, int n) {
    if (n == 1)
        return pow(1, 1) / fact(1);
    else
        return pow(x, n) / fact(n) + f(x, n - 1);
}
```

EXAMPLE(STRING)

算字串長度

len("abcde") ⇒ 5

字串

在字元陣列中連續儲存的字元，結尾是'\0'

```
/*       ↓ 如果後面有設定初值的話，這裡可以省略，[]會自動填上初值的個數 */
char abc[] = {'a', 'b', 'c', '\0'};
/*    ↕ 兩種寫法都一樣    */
char abc[] = "abc";
```

len(S) = 到'\0'之前的字元個數

```
/* 在main外面 */
char A = "abcde";
/* 呼叫的時候 */
    printf("%d", len(0));
```

```
int len(int st) {
    if (A[st] == '\0')
        return 0;
    else
        return 1 + len(st + 1);
}
```

EXAMPLE(STRING)

反轉字串

"abc" ⇒ "cba"

A: 要反轉的字串
B: 轉換完的字串

```
void reverse_str(int st, int en) {
    if (st > en) {
        return;
    }
    else if (st == en) {
        B[st] = A[st];
    }
    else {
        B[st] = A[en];
        B[en] = A[st];
        reverse_str(st + 1, en - 1);
    }
}
```

小進階

判斷字串是不是回文

P("abcde") ⇒ F

P("abcba") ⇒ T

```
/* A = "abcde"  */
is_palindromic(0, 4) /* ⇒  0 */
/* A = "abcba"  */
is_palindromic(0, 4) /* ⇒  1 */
/* A = "abba"   */
is_palindromic(0, 3) /* ⇒  1 */
```

```
int is_palindromic(int st, int en) {
    if (st == en) {
        return 1;
    }
    else if (st > en) {
        return 0;
    }
    else {
        if (A[st] == A[en])
            return check(st + 1, en - 1);
        else
            return 0;
    }
}
```

EXAMPLE(SEARCH)

二分搜尋

bs([1, 3, 16, 20, 22, 70, 99], 21)

⇒ bs([22, 70, 99], 21)

⇒ bs([22], 21)

⇒ 'not-found

bs([1, 3, 16, 20, 22, 35, 99], 35)

⇒ bs([22, 35, 99], 35)

⇒ found

TRY IT

寫一個遞迴函數，輸入陣列的頭尾索引跟要搜尋的數字
找到的話回傳索引，沒找到的話回傳-1

```
/* int A[] = {1, 3, 5, 7, 9} */
bs(0, 4, 7); /* ⇒  3  */
/* int A[] = {1, 3, 5, 7, 9} */
bs(0, 4, 4); /* ⇒  -1 */
```

```
int bs(int st, int en, int ta) {
    int mid = (st + en) / 2;
    if (st >= en) {
        return -1;
    }
    else {
        if (A[mid] == ta)
            return mid;
        else if (A[mid] > ta)
            return bs(st, mid - 1, ta);
        else
            return bs(st, mid - 1, ta);
    }
}
```

小結

- 其實還有很多話題
  - 尾遞迴
  - 資料結構的遞迴
  - 離散數學的遞迴
  - …

END

THANKS :)

工商時間

- web security
  - 11/22 (五) 16:00~18:00
  - 資電205
  - **jyny**