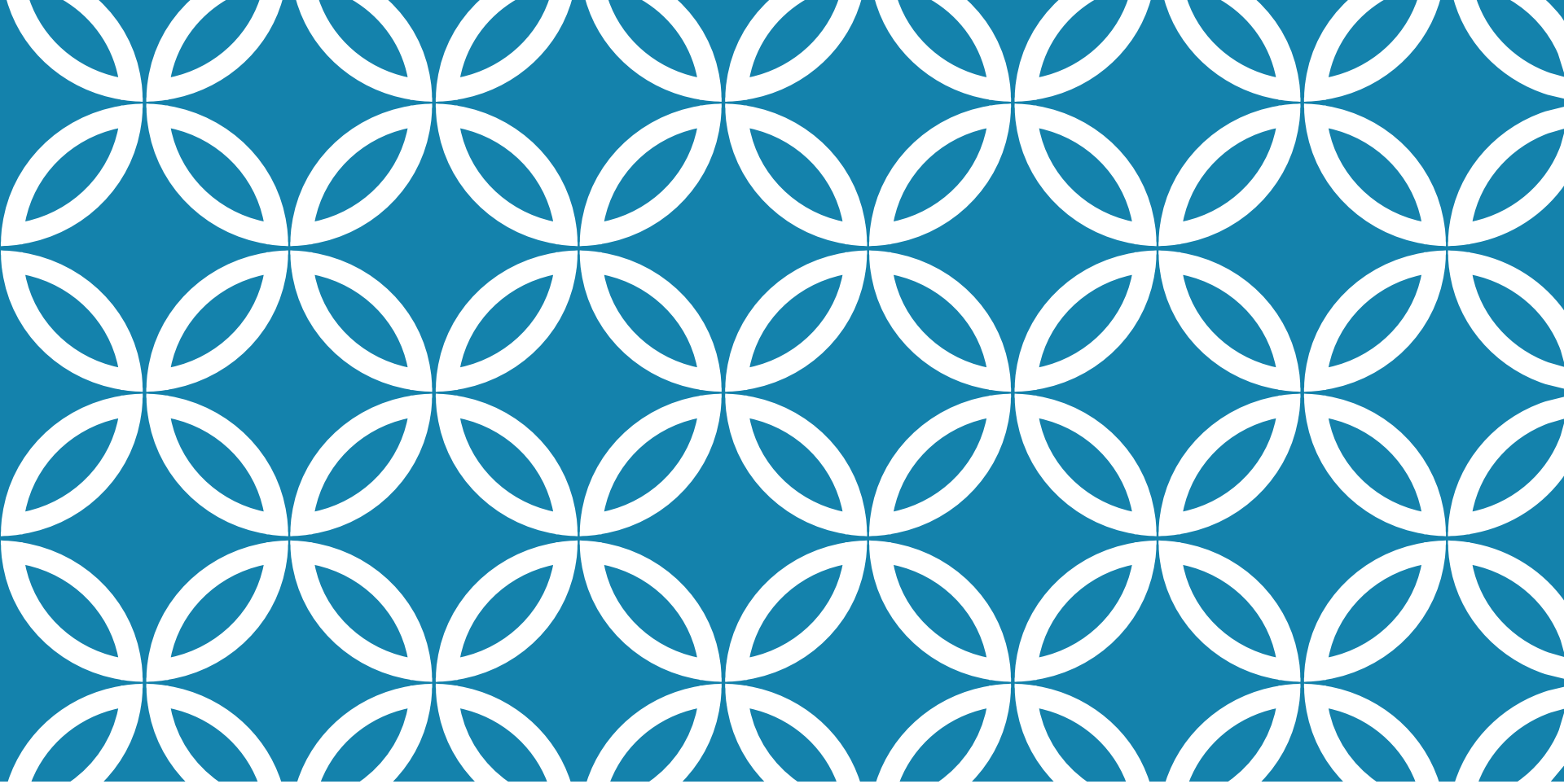


時間複雜度
費氏數列 FIBONACCI

資料結構
鍾宜玲

費氏數列 FIBONACCI

0, 1, 1, 2, 3, 5, 8, 13, ...

即 $F_0 = 0$、$F_1 = 1$，且當 n>=2，$F_n = F_{n-1} + F_{n-2}$

F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}
0	1	1	2	3	5	8	13	21	34	55	89

設計一個程式，執行時輸入n (0<=n<=40)，計算並輸出 Fibonacci 數列的第 n 項。

(第0項是0，第1項是1，...)

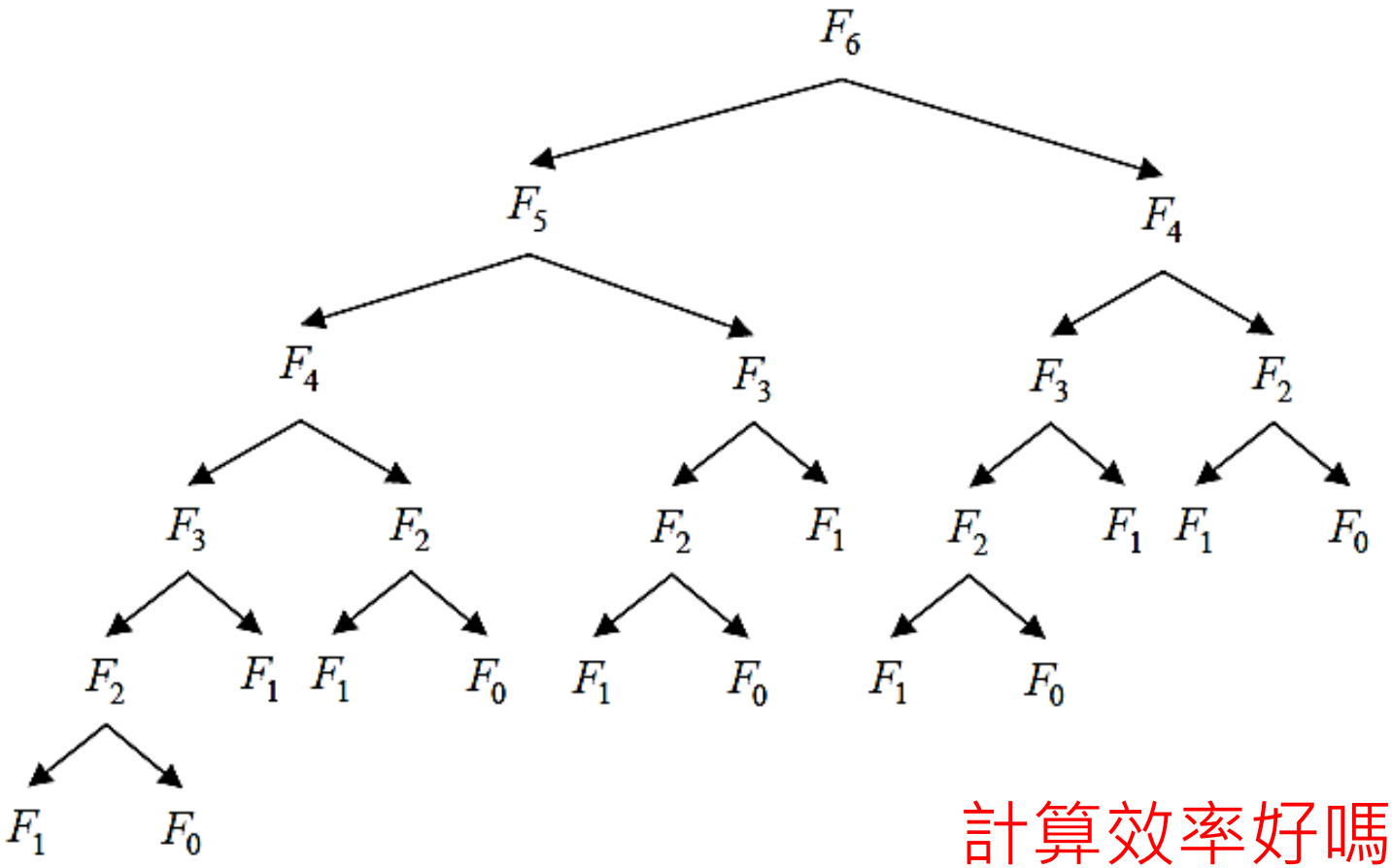
遞迴程式

在數學上，**費波那契數列**是以遞迴的方法來定義：

- $F_0 = 0$
- $F_1 = 1$
- $F_n = F_{n-1} + F_{n-2}$ (n≧2)

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

計算第6項的呼叫過程



計算效率好嗎？

遞迴程式的時間複雜度

- 遞迴程式看起來簡單，但是並不實用
- 嚴重的重覆計算
- 時間複雜度為指數等級

$$O(\varphi^n), \varphi = \frac{1+\sqrt{5}}{2} \cong 1.618$$

$$\varphi^n < 2^n$$

- 因此，時間複雜度可以用 $O(2^n)$ 表示。

使用迴圈

```
long fib(int n)
{
    int i;
    long pre=0, curr=1, next;
    for(i=2; i<=n; i++){
        next = pre + curr;
        pre = curr;
        curr = next;
    }
    return curr;
}
```

```
long fib(int n)
{
    int i;
    long a=0, b=1;
    for(i=1; i<=n; i++){
        a=a+b;
        b=a-b;
    }
    return a;
}
```

迴圈程式之時間複雜度為 $O(n)$

迴圈與遞迴的比較

n	迴圈執行次數	遞迴程式 呼叫次數
1	1	0
2	2	2
3	3	4
4	4	8
5	5	14
6	6	24
7	7	40
8	8	66