



# DATA STRUCTURE

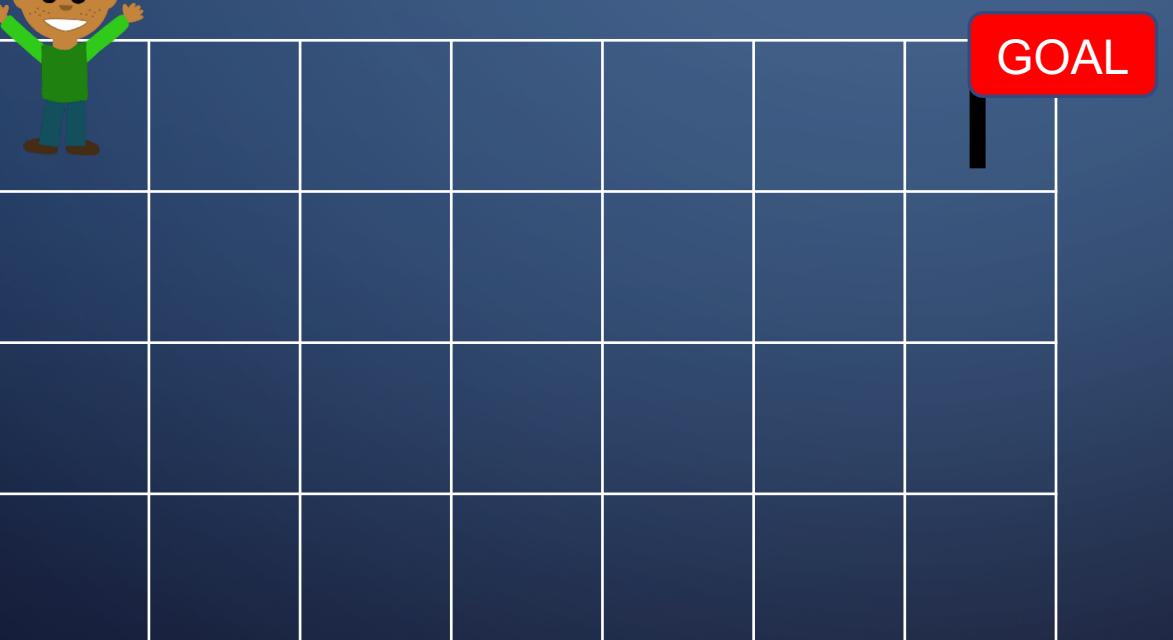
## 01. 基本程式設計(I)

# 實習課的目的

- 訓練邏輯思考的程式腦！
- 寫程式就跟堆積木一樣！

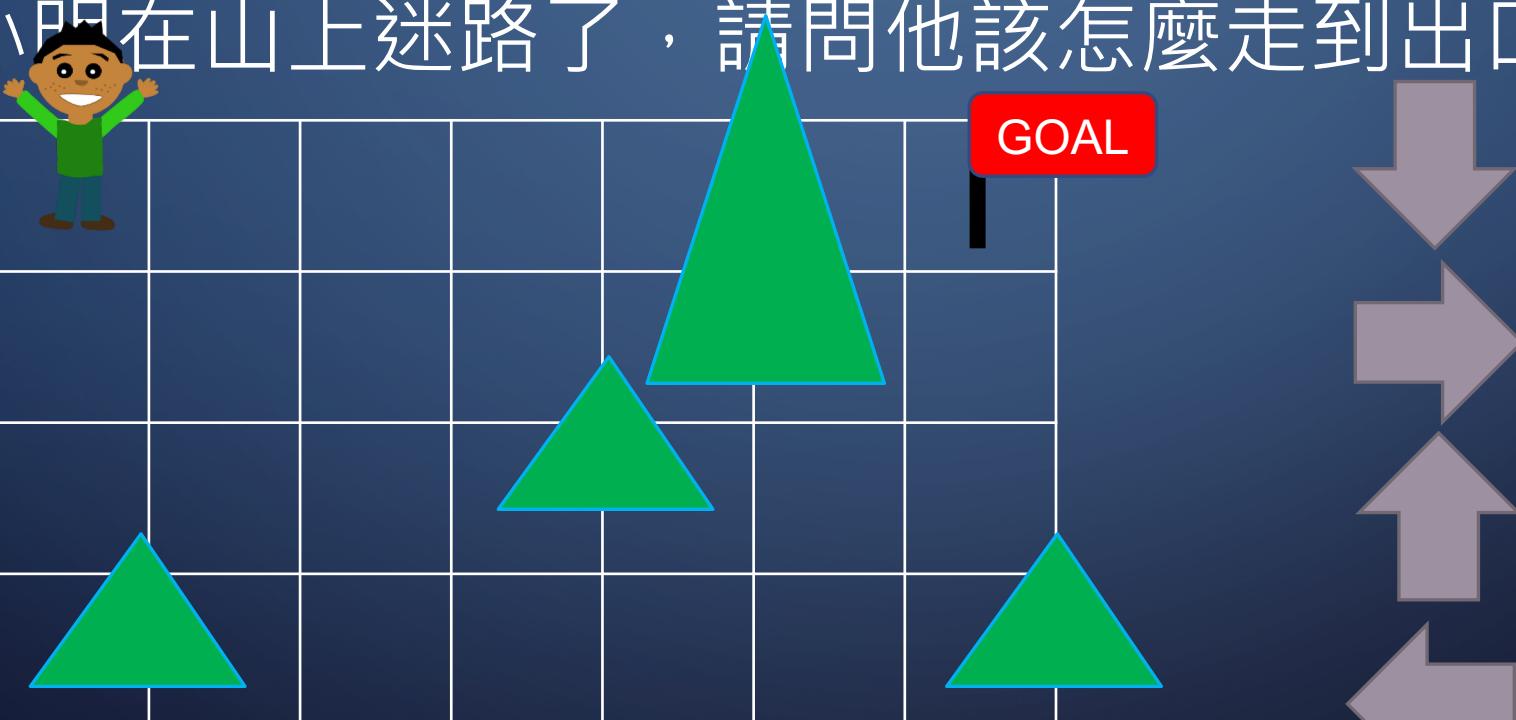
# 範例

- 小明又在山上迷路了，請問他該怎麼走到出口？



# 範例

- 小明在山上迷路了，請問他該怎麼走到出口？



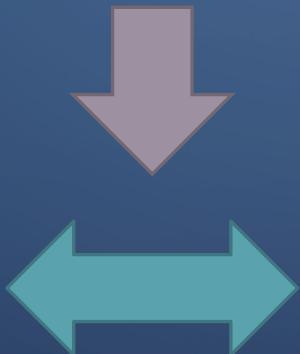
# 寫程式也是一樣

- 利用可用的單純指令(or函數)組合出想要的結果
- 指令的數目很少，但可以組合出無限多總可能
- 結構化程式設計
- 物件導向程式設計

# 結構化程式設計

- 基於循序、選擇、迴圈三個要件組成

循序> 往下走



選擇> 分支

迴圈> 重複直到某個條件

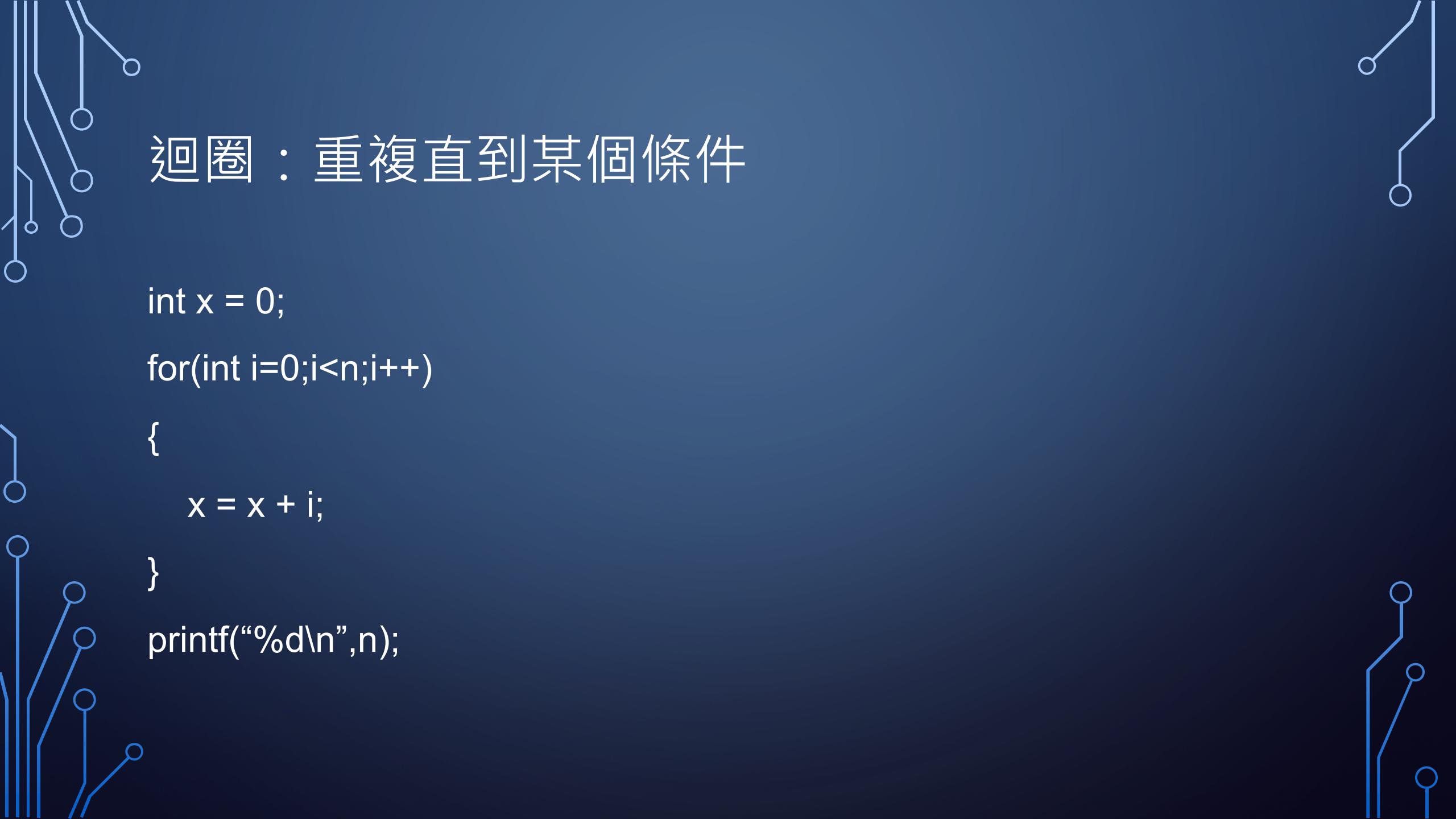
# 範例

- 計算 $1+2+\dots+n$ 的總和
- 小明：我知道！ $n(1+n)/2$ ！



循序：一行一行由上往下做

```
int x = 0;  
x = x + 1;  
x = x + 2;  
x = x + 3;  
...  
x = x + n;  
printf("%d\n",n);
```



## 迴圈：重複直到某個條件

```
int x = 0;  
  
for(int i=0;i<n;i++)  
{  
    x = x + i;  
}  
printf("%d\n",n);
```

# FOR 迴圈 VS WHILE 迴圈

```
for(int i=0;i<n;i++)  
{  
    ...  
}
```

```
i=0;  
while(i<n)  
{  
    ...  
    i++;  
}
```

- 迴圈內本身就包含判斷式 ( IF - ELSE ) !

# 程式訓練：可以用這三種結構組合把問題解決

- 所有問題都可以用這三種結構把問題解決嗎？
  - a.k.a 所有演算法都可以用這三種結構完成嗎？
- Yes, but...
  - 不保證執行時間
  - 資料結構/演算法課之後就會探討每個程式/每個演算法的執行時間

# PRACTICE : 聖誕樹

- 練習使用IF-ELSE and FOR-LOOP
- 繪製出下列聖誕樹
- 比較並記錄7層、11層、21層、41層、51層的計算時間

```
請輸入層數：7
*
 ***
 *****
 ******
 *****
 ****
 *****
 ****
 ****
 ****
 ****
 ****
 ****
 ****
 ****
```



# 計算時間使用TIME.H

```
#include <time.h>  
  
clock_t t1;  
  
t1 = clock();
```

t1會取得目前系統的時間(單位是clock = 跳表)

CLOCKS\_PER\_SEC

系統常數,將跳表數轉成秒數

```
#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
  
int main()  
{  
    int level = 0;  
    printf("請輸入層數 : ");  
    scanf_s("%d", &level);  
    clock_t t1, t2;  
    t1 = clock();  
    // 這邊補上繪製聖誕樹的code  
  
    // 到此為止  
    t2 = clock();  
  
    printf("%lf\n", (t2 - t1) /  
    (double)(CLOCKS_PER_SEC));  
  
    system("pause");  
    return 0;  
}
```

# HINT

- Case

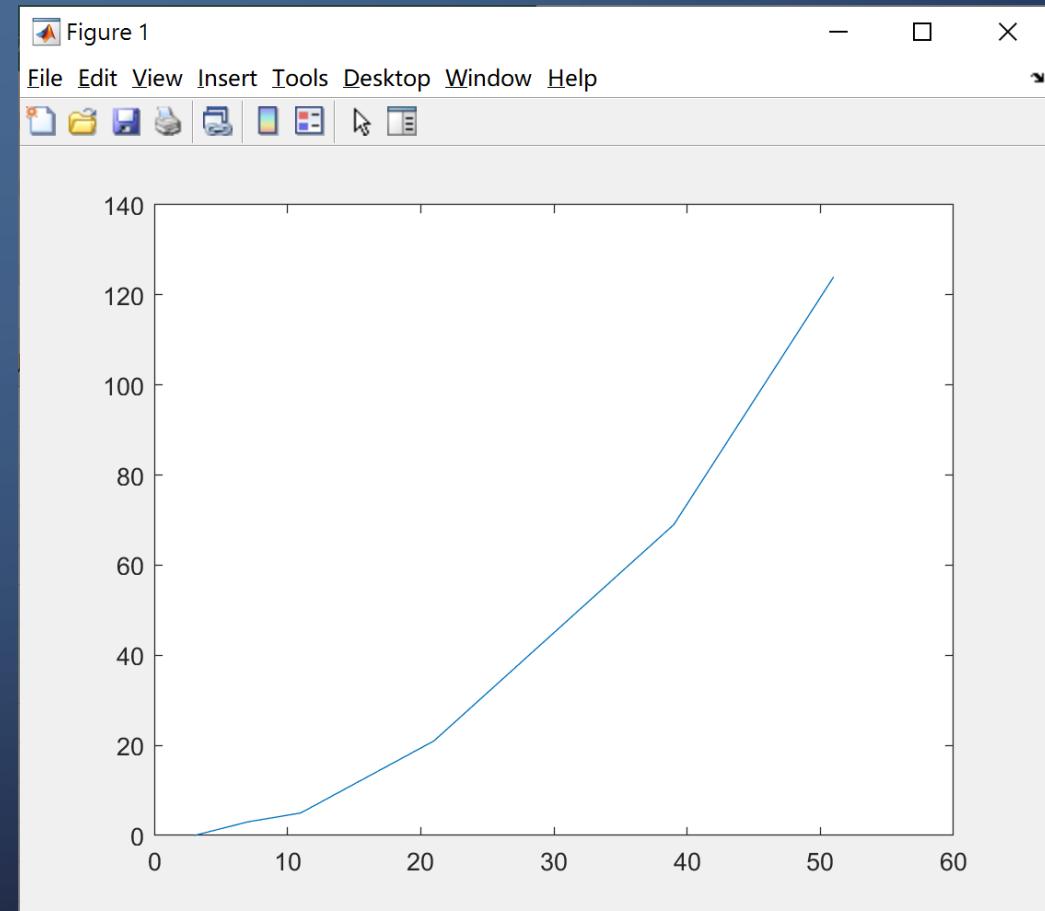
- 第一行先畫個 $n$ 個空格，畫了1個星星
- 第三行先畫個 $n-1$ 個空格，畫了3個星星
- ...
- 所以...
- 我需要兩個for loop!
  - 裡面那層for loop要做兩件事：畫空格、畫星星
  - 裡面那層for loop的次數會隨著層數增加
  - 第一行一共要輸出 $n+1$ 個字元，第二層一共要輸出 $(n-1)+3$ 個字元，...

請輸入層數 : 7

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

# 結果

- 執行時間大致上會成一個指數上升



# 實作：指數計算機

- 實做兩種指數計算機的算法，並比較花費時間
- 第一種：循規蹈矩
  - ```
for(int i=0;i<n;i++)  
    result = result * a;
```
- 第二種：快速幕算法

# 快速幕算法

- 啟發： $x$ 的平方 =  $x$ 的指數部分 \* 2
  - $3^2 = 3^{1*2}$  ,  $8^2 = (2^3)^2 = 2^{3*2} = 64$  ,  $81^2 = (3^4)^2 = 3^{4*2} = 6561$
  - 換句話說，剛剛連乘的時候，有很大一部分其實可以不用重複計算....？
  - Ex: 我要算3的8次方，本來我要 $3*3*3*3*3*3*3*3$  乘8-1次，但其實可以...
  - $3*3*3*3$  乘3次，然後平方乘1次！？
  - 再往下思考，前面的 $3*3*3*3=3$ 的4次方也可以看成...
  - $3*3=3^2$  然後平方
  - 所以原來的8-1次乘法就會變成3\*3 平方 平方 只需要3次乘法!!!

# 快速幕算法

- 觀察指數部分
  - 平方=指數\*2 相對於2進位制也就是往前推進一位(把指數用2進位表示)
  - $2^{100} = (2^{10})\text{的平方} = (2^1\text{的平方})\text{的平方}$
- 而 $2^{101}$ 呢 ?
  - $2^{101} = (2^{10})\text{的平方再}*2$
- $2^{1011}$ 呢 ?
  - $2^{1011} = (2^{101})\text{的平方再}*2$

# 快速冪算法

計算 $a$ 的 $b$ 次方

1. 將次方數(冪) $b$ 轉成二進位 $c$
2. 預設輸出 $r=1$
3. 從 $c$ 的第一位(必為1)開始，一位一位往後讀取直到結束
4. 如果是1，則 $r=r \cdot r \cdot a$   
    如果是0，則 $r=r \cdot r$
5. 輸出 $r$

## 實作

- 請實作上述兩種幕次計算法，並比較所需的乘法次數和花費的時間
- 計算 $2$ 的5次方、10次方、15次方、20次方、25次方、30次方分別所需的乘法次數和花費時間
- Ex:  $2$ 的5次方用方法1需要4次乘法，方法2需要3次乘法
- 列表並將橫軸：次方數-縱軸：時間以EXCEL繪製出圖形
  - 如果不會用excel的話，就放成像這樣的表格也可以

| 次方數 | 次數 | 時間    |
|-----|----|-------|
| 5   | 7  | 0.002 |
| 10  | 11 | 0.004 |
| 15  | 21 | 0.016 |
| 20  | 31 | 0.029 |
| 25  | 41 | 0.07  |
| 30  | 51 | 0.105 |