

# 資料結構報告

林聖紘

Oct. 22 / 2024

# Ackermann Function

1 解題說明	2
2 演算法設計與設計	3
3 效能分析	5
4 測試與過程	7

## 解題說明

$$A(m, n) = \begin{cases} n + 1 & \text{若 } m=0 \\ A(m-1, 1) & \text{若 } m>0 \text{ 且 } n=0 \\ A(m-1, A(m, n-1)) & \text{若 } m>0 \text{ 且 } n>0 \end{cases}$$

如果  $m$  為 0, 則返回  $n + 1$ 。

如果  $m$  大於 0 且  $n$  為 0, 則遞迴調用  $\text{ack}(m - 1, 1)$ 。

如果  $m$  和  $n$  都大於 0, 則遞迴調用  $\text{ack}(m - 1, \text{ack}(m, n - 1))$ 。

## 演算法設計與設計

兩者都有使用 <chrono> 標頭檔, 用來計算效能以及時間

遞迴方式

```
int ack(int m, int n) {  
    cnt++;  
    if (m == 0) {  
        return n + 1;  
    }  
    else if (m > 0 && n == 0) {  
        return ack(m - 1, 1);  
    }  
    else if (m > 0 && n > 0) {  
        return ack(m - 1, ack(m, n - 1));  
    }  
    return -1;  
}
```

## 演算法設計與設計

### 非遞迴方式

```
stack<tuple<int, int, int>> s;  
s.push(make_tuple(m, n, 0));  
int result = 0;
```

(使用堆疊模擬遞迴)

```
while (!s.empty()) {  
    auto curr = s.top();  
    s.pop();  
    int m_curr = get<0>(curr);  
    int n_curr = get<1>(curr);  
    int r = get<2>(curr);  
  
    if (r == 1) {  
        n_curr = result;  
    }  
  
    if (m_curr == 0) {  
        result = n_curr + 1;  
    }  
    else if (n_curr == 0) {  
        s.push(make_tuple(m_curr - 1, 1, 0));  
    }  
    else {  
        s.push(make_tuple(m_curr - 1, 0, 1));  
        s.push(make_tuple(m_curr, n_curr - 1, 0));  
    }  
}
```

## 效能分析

遞迴:



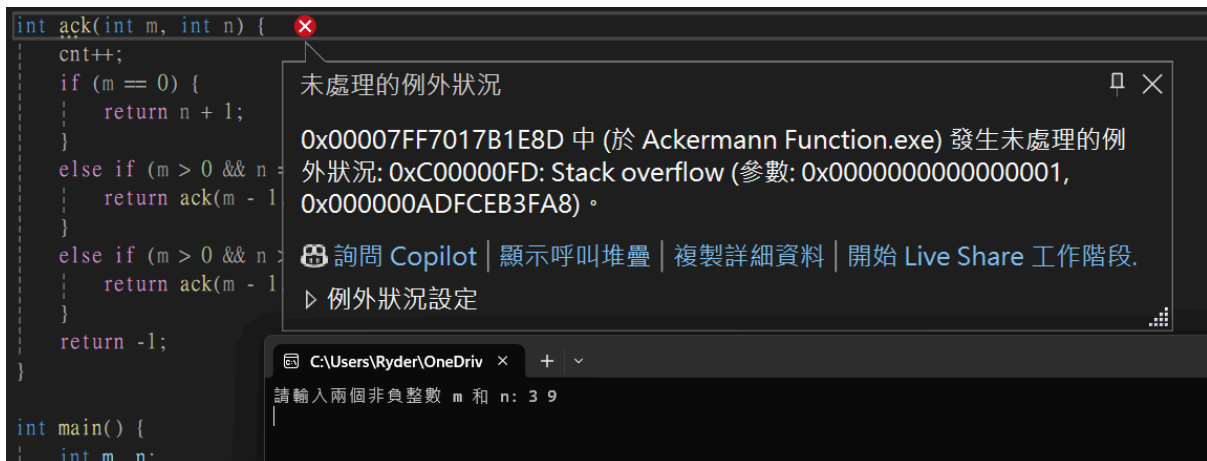
Microsoft Visual Studio

請輸入兩個非負整數 m 和 n: 3 8  
Ackermann(3, 8) = 2045  
函數執行的次數: 2785999  
執行時間: 0.0125932 秒

C:\Users\Ryder\OneDrive\Desktop\資料結構\Ackermann Function\x64\Debug\Ackermann Function.exe (流程 44824) 已結束，代碼為 0 (0x0)。

若要在偵錯停止時自動關閉主控台，請啟用 [工具] -> [選項] -> [偵錯] -> [偵錯停止時，自動關閉主控台]。  
按任意鍵關閉此視窗]

最高只能(3,8)，到(3,9)會跑不動(如下)



int ack(int m, int n) {  
 cnt++;  
 if (m == 0) {  
 return n + 1;  
 }  
 else if (m > 0 && n == 0) {  
 return ack(m - 1, n);  
 }  
 else if (m > 0 && n > 0) {  
 return ack(m - 1, ack(m, n - 1));  
 }  
 return -1;  
}  
  
int main() {  
 int m, n;  
 ...  
}

未處理的例外狀況

0x00007FF7017B1E8D 中 (於 Ackermann Function.exe) 發生未處理的例外狀況: 0xC00000FD: Stack overflow (參數: 0x0000000000000001, 0x0000000ADFCB3FA8)。


詢問 Copilot | 顯示呼叫堆疊 | 複製詳細資料 | 開始 Live Share 工作階段。  
▷ 例外狀況設定

C:\Users\Ryder\OneDrive\Desktop\資料結構\Ackermann Function\x64\Debug\Ackermann Function.exe (流程 44824) 已結束，代碼為 0 (0x0)。

請輸入兩個非負整數 m 和 n: 3 9

## 效能分析

非遞迴:



```
Microsoft Visual Studio
請輸入非負整數 m: 3
請輸入非負整數 n: 13
Ackermann(3, 13) = 65533
函數執行的次數: 1
執行時間: 782.926 秒

C:\Users\Ryder\OneDrive\Desktop\資料結構\Ackermann非遞迴\x64\Debug\Ackermann非遞迴.exe (流程 34268) 已結束，代碼為 0 (0x0)。
若要在偵錯停止時自動關閉主控台，請啟用 [工具] -> [選項] -> [偵錯] -> [偵錯停止時，自動關閉主控台]。
按任意鍵關閉此視窗
```

最高只能(3,13), 耗時約13分鐘

分析結果:

非遞迴的效能比遞迴好

## 測試與過程

程式將提示你輸入兩個整數值  $m$  和  $n$ 。

隨後，程式將計算阿克曼函數  $A(m,n)$  的值，並顯示計算結果以及函數的總調用次數。這樣可以幫助你了解在計算過程中函數被調用的頻率，特別是對於較大的輸入值。