**Hw1**

**題目**

Ackermann's function A(m,n) is defined as fllows: A(m,n) = {[n+1, if m = 0],[A(m-1, 1), if n = 0],[A(m-1, A(m,n-1)), otherwise]} This function is studied because it grows very fast for small values of m and n. Write a recursive function for computing this function. Then write a nonrecursive algorithm for computing Ackermann's function.

**說明**

使用遞迴完成功能，依照題目要求撰寫，以下為該題虛擬碼。

A(int m, int n){

if(m == 0){return n+1;

}else if(n == 0){return A(m,n);

}else{return A(m-1,A(m,n-1));}  
}

**效能分析**

遞迴的深度與次數受m和n影響，尤其是m。

當m = 0，時間複雜度為O(1)。

當m = 1，時間複雜度為O(n)。

當m = 2，時間複雜度為O(n^2)。

當m的值越大，所需的效能會越龐大。

**執行結果**

一張含有 文字, 字型, 螢幕擷取畫面, 黑色 的圖片

自動產生的描述

**範例計算(已排版)**

m = 2, n = 3。

A(2,3) = A(1,A(2,2)) = A(1,7) = 9

A(2,2) = A(1,A(2,1)) = A(1,5) = 7

A(2,1) = A(1,A(2,0)) = A(1,3) = 5

A(1,7) = A(0,A(1,6)) = A(0,8) = 9

A(1,6) = A(0,A(1,5)) = A(0,7) = 8

A(1,5) = A(0,A(1,4)) = A(0,6) = 7

A(1,4) = A(0,A(1,3)) = A(0,5) = 6

A(1,3) = A(0,A(1,2)) = A(0,4) = 5

A(1,2) = A(0,A(1,1)) = A(0,3) = 4

A(1,1) = A(0,A(1,0)) = A(0,2) = 3

**心得**

Ackermann’s function是一個會快速增長且難以計算的函數，不易優化。

即便使用C或C++撰寫，也容易出現效能瓶頸。