

Ackermann 函數

解題說明：

當 m 為 0 時，回傳 $n + 1$ 。

當 m 大於 0 且 n 為 0 時，調用 Ackermann 函數，將 m 減 1， n 設為 1。通用情況下，它計算 $A(m-1, A(m, n-1))$ 。

Algorithm Design & Programming:

```
#include <iostream>
using namespace std;
int ackermann(int m, int n) {
    // 當 m 為 0 時，返回 n + 1
    if (m == 0) {
        return n + 1;
    }
    // 當 m 大於 0 且 n 為 0 時，調用Ackermann函數，m 減 1，n 為 1
    if (n == 0) {
        return ackermann(m - 1, 1);
    }
    // 計算 A(m-1, A(m, n-1))
    return ackermann(m - 1, ackermann(m, n - 1));
}

int main() {
    int m, n;
    cout << "請輸入 m 和 n 的值: ";
    cin >> m >> n;

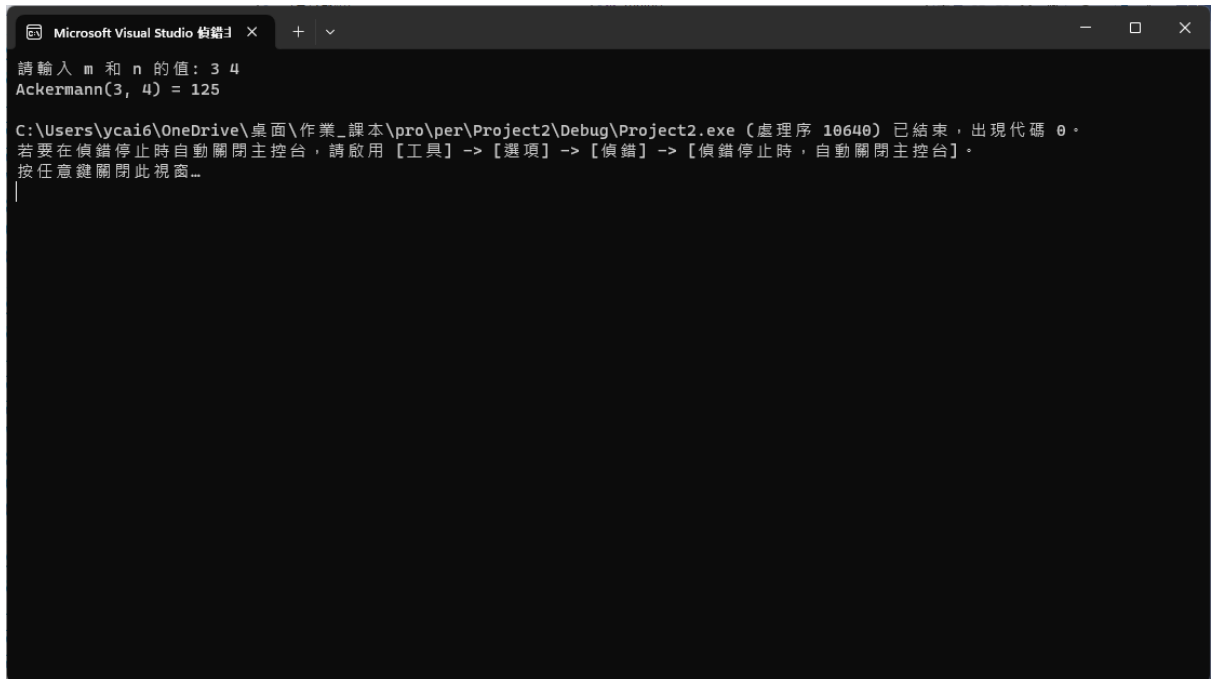
    int result = ackermann(m, n);
    cout << "Ackermann(" << m << ", " << n << ") = " << result << endl;

    return 0;
}
```

效能分析:

較大的 m 和 n 值使 Ackermann 函數增長非常快，並可能導致遞迴深度過大或計算時間過長

測試與驗證:



```
Microsoft Visual Studio 偵錯器 x + -
請輸入 m 和 n 的值: 3 4
Ackermann(3, 4) = 125

C:\Users\ycai6\OneDrive\桌面\作業_課本\pro\per\Project2\Debug\Project2.exe (處理序 10640) 已結束，出現代碼 0。
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...
```

冪集:

解題說明:

輸入集合的元素個數 n 以及這 n 個元素，遞迴函數 `computePowerset` 被呼叫處理包含或不包含當前元素的情況，如果已處理完集合中的所有元素，當前子集合會加入到冪集。

最後程式輸出冪集中的每個子集。

Algorithm Design & Programming:

```
#include <iostream>
#include <vector>

using namespace std;

// 計算集合 S 的冪集
void computePowerset(const vector<char>& S, vector<char>& currentSubset, int
index, vector<vector<char>>& powerset) {
    // 如果已處理完 S 中所有元素，把當前子集加入冪集
    if (index == S.size()) {

        powerset.push_back(currentSubset); return;
    }

    // 處理包含當前元素的情況
    currentSubset.push_back(S[index]);
    computePowerset(S, currentSubset, index + 1, powerset);

    // 處理不包含當前元素的情況
    currentSubset.pop_back();
    computePowerset(S, currentSubset, index + 1, powerset);
}

int main() {
    vector<char> S; int n;

    cout << "請輸入集合的元素個數 n: "; cin >> n;

    cout << "請依次輸入集合的元素: "; for (int i = 0; i < n; ++i) {
        char element; cin >> element;
        S.push_back(element);
    }

    vector<vector<char>> powerset; vector<char> currentSubset;

    // 初始化一個空子集，然後計算冪集
    computePowerset(S, currentSubset, 0, powerset);

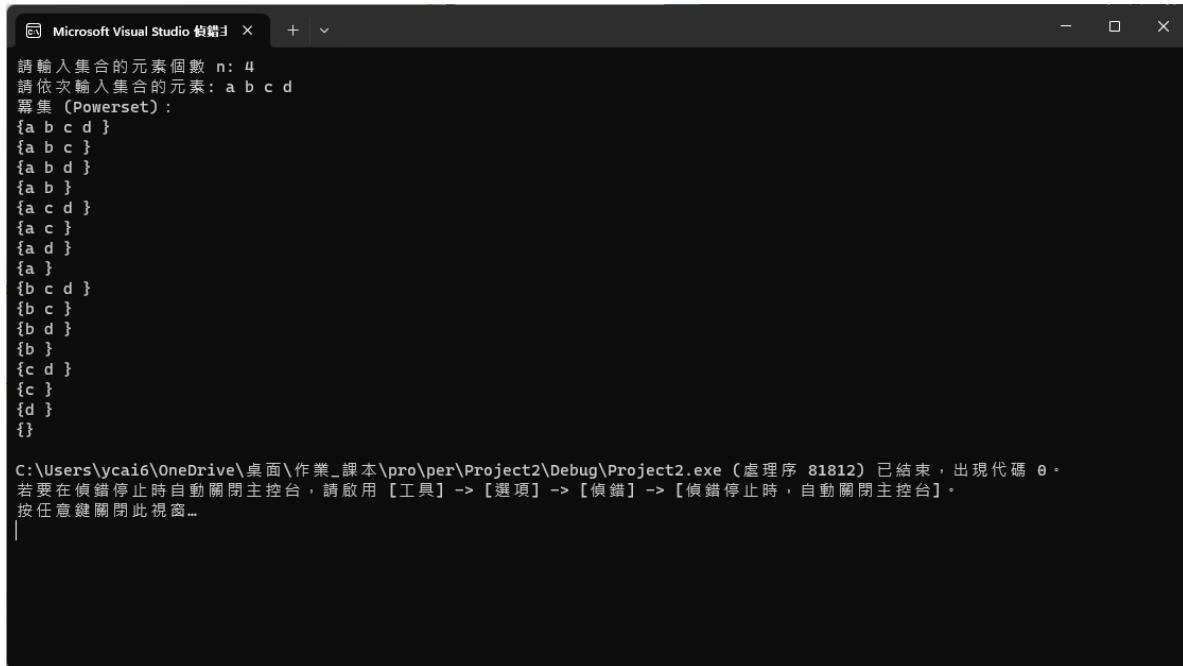
    cout << "冪集 (Powerset) : " << endl;
    for (const vector<char>& subset : powerset) {
        cout << "{";
        for (char element : subset) {
            std::cout << element << " ";
        }
        cout << "}" << endl;
    }
}
```

```
        return 0;
    }
}
```

效能分析:

遞迴函數的時間複雜度是 $O(2^n)$, 對於大型集合這種方法可能不太實用。

測試與驗證:



```
Microsoft Visual Studio 偵錯器 x + v
請輸入集合的元素個數 n: 4
請依次輸入集合的元素: a b c d
冪集 (Powerset) :
{a b c d }
{a b c }
{a b d }
{a b }
{a c d }
{a c }
{a d }
{a }
{b c d }
{b c }
{b d }
{b }
{c d }
{c }
{d }
{}
C:\Users\ycai6\OneDrive\桌面\作業_課本\pro\per\Project2\Debug\Project2.exe (處理序 81812) 已結束，出現代碼 0。
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...
```