11-3 測試 Vec 類別

目前我們的 Vec 類別如下：

[vec.h]

```
#ifndef VEC_H_INCLUDED
#define VEC_H_INCLUDED

class Vec {
private:
    int *m_data;
    int m_size;
    int m_capacity;
public:
    Vec();
    ~Vec();
    int size();
    int capacity();
    void push_back(int val);
    void pop_back();
};

Vec::Vec()
{
    m_data = nullptr;
    m_size = 0;
    m_capacity = 0;
}

Vec::~Vec()
{
    if(m_data!=nullptr)
    {
        delete [] m_data;
    }
}

int Vec::size()
{
    return m_size;
}

int Vec::capacity()
{
    return m_capacity;
}

void Vec::push_back(int val)
{
    // 如果空間不夠用
    if(m_size==m_capacity)
    {
        // 每次成長為原來的 2 倍大小
        int new_capacity = m_capacity*2;

        if(new_capacity==0)
            new_capacity = 1;

        // 配置一塊新的空間
        int *new_data = new int[new_capacity];
        // 把資料搬到新的空間
        for(int i=0; i<m_size; i++)
        {
            new_data[i] = m_data[i];
        }
```

```
        // 釋放舊的空間，改用新空間
        if(m_data!=nullptr)
            delete [] m_data;

        m_data = new_data;
        m_capacity = new_capacity;
    }

    m_data[m_size] = val;
    m_size++;
}

void Vec::pop_back()
{
    m_size--;
}

#endif // VEC_H_INCLUDED
```

在主程式中引入其標頭檔來使用看看。

```
#include <iostream>
#include "vec.h"

using namespace std;

int main()
{
    Vec a;

    a.push_back(1);
    cout << "Cap:" << a.capacity() << " Size:" << a.size() << endl;
    a.push_back(3);
    cout << "Cap:" << a.capacity() << " Size:" << a.size() << endl;
    a.push_back(5);
    cout << "Cap:" << a.capacity() << " Size:" << a.size() << endl;
    a.push_back(7);
    cout << "Cap:" << a.capacity() << " Size:" << a.size() << endl;
    a.push_back(9);
    cout << "Cap:" << a.capacity() << " Size:" << a.size() << endl;

    return 0;
}
```

注意看 capacity 和 size 的變化。

```
Cap:1 Size:1
Cap:2 Size:2
Cap:4 Size:3
Cap:4 Size:4
Cap:8 Size:5
```

Revision #2
★Created 8 June 2026 00:42:00 by huihui
Updated 8 June 2026 00:50:38 by huihui