11-4 重載 [] 運算子

現在我們還缺一個重要的功能，那就是存取 Vec 裡的值。

試著執行這段程式看看。

[main.cpp]

```
#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;

    for(int i=0; i<a.size(); i++)
    {
        cout << a[i] << " ";
    }
    cout << endl;

    return 0;
}
```

我們會得到這個錯誤訊息。

> error: no match for 'operator[]' (operand types are 'Vec' and 'int')|

因為在第 23 行中的`cout << a[i]`，編譯器不知道`Vec []`要怎麼處理。

`[]`是一個運算子，每種資料型別的`[]`運算可能有所不同，我們必須自己實作 Vec 的`[]`運算子運算。

在 Vec 宣告的尾端加入這行重載`[]`運算子的成員函數。

```
    ...
    void pop_back();
    int& operator[] (int index); // <== 加入這行
};
```

在函數實作部分加入。

```
int& Vec::operator[] (int index)
{
    return m_data[index];
}
```

這段程式碼是在做什麼呢? 我們把它拆解成四個部分來看。

- `int&`（回傳型態）：注意那個 & 符號，它代表「參照」（Reference）。意思是它不只傳回那個數字的值，而是直接把那塊記憶體的「真身」給你。這樣你才能寫出像 a[0] = 10; 這樣的程式碼，直接修改裡面的數值。
- `Vec::`（所屬類別）：這代表這個功能是屬於 Vec 這個類別。
- `operator[]`（函數名稱）：這是 C++ 的特殊關鍵字。組合起來就是告訴編譯器：「我要重新定義中括號 `[]` 的功能」。
- `(int index)`（參數）：中括號裡面填的數字（索引值）。例如 a[3]，這個 index 就是 3。

現在可以執行這段程式來驗證了。

[main.cpp]

```
#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;

    for(int i=0; i<a.size(); i++)
    {
        cout << a[i] << " ";  // 1 3 5 7 9
```

```
        a[i] = a[i]*2; // 把值修改為 2 倍
    }
    cout << endl;

    for(int i=0; i<a.size(); i++)
    {
        cout << a[i] << " ";  // 2 6 10 14 18
    }
    cout << endl;

    return 0;
}
```

Cap:1 Size:1 Cap:2 Size:2 Cap:4 Size:3 Cap:4 Size:4 Cap:8 Size:5 1 3 5 7 9 2 6 10 14 18

Revision #2
★Created 2026-06-08 00:50:50 UTC by huihui
Updated 2026-06-08 01:08:25 UTC by huihui