赋值运算符和下标运算符重载程序实例

- 【例2-19】整型动态数组类的实现。在下面的整型数组类IntArray中，定义了赋值运算符重载函数和下标运算符重载函数。

```
// IntArray.h
class IntArray
{
public:
    IntArray(int = 10);
    ~IntArray();
    IntArray& operator=(const IntArray &); //重载赋值运算符
    int& operator[] (int); //重载下标操作符
    void DisplayArrayInfo();
private:
     int m_size;              // 数组大小
     int *m_ptr;              // 指向数组第一个元素的指针
};
```

```cpp
// IntArray.cpp
#include "IntArray.h"
#include <assert.h>
#include <iostream>
using namespace std;
//定义构造函数
IntArray::IntArray(int arraySize)
{
    m_size = (arraySize > 0 ? arraySize:10);
    m_ptr = new int[m_size];
    assert(m_ptr != 0);
    for (int i = 0; i < m_size; i++)
        m_ptr[ i ] = 0;
}
//定义析构函数
IntArray::~IntArray()
{
    delete [] m_ptr;
}
```

```cpp
//定义赋值运算符重载函数
IntArray&
    IntArray::operator=(const IntArray &right)
{
     if (&right != this) // 检测自赋值
     {
         if(m_size != right.m_size)
         {
             delete [] m_ptr;
             m_size = right.m_size;
             m_ptr = new int[m_size];
             assert(m_ptr != 0);
         }
        for (int i = 0; i <m_size; i++)
             m_ptr[i] = right.m_ptr[i];
     }
     return *this;
}
```

```
//定义下标运算符重载函数
int& IntArray::operator[] (int subscript)
{
    assert(0 <= subscript && subscript < m_size); //下标越界，退出程序
    return m_ptr[subscript];
}
void IntArray::DisplayArrayInfo()
{
    for(int i=0;i<m_size;i++)
        cout<<m_ptr[i]<<' ';
    cout<<endl;
}
```

```cpp
// testIntArray.cpp
#include <iostream>
#include "IntArray.h"
using namespace std;
int main()
{
    IntArray arrayA;
    IntArray arrayB(5);
    for (int i=0;i<5;i++)
        arrayB[i]=i+1;        //调用下标运算符函数
    cout<<"数组arrayA为："<<endl;
    arrayA.DisplayArrayInfo();
    cout<<"数组arrayB为："<<endl;
    arrayB.DisplayArrayInfo();
    arrayA=arrayB;          //调用赋值运算符函数
    cout<<"执行arrayA=arrayB后，数组arrayA为："<<endl;
    arrayA.DisplayArrayInfo();
    return 0;
}
```

运行结果为：

数组arrayA为：

0 0 0 0 0 0 0 0 0 0

数组arrayB为：

1 2 3 4 5

执行arrayA=arrayB后，

数组arrayA为：

1 2 3 4 5

- 提示：“arrayB[i]=i+1;”则相当于“arrayB.m_ptr[i]=i+1;”。重载了下标运算符“[]”,可以直接把数组对象当做数组名使用，非常直观易懂，而实际上还是要调用成员函数，为对象的指针成员指向内存的内容操作。