程序设计实习

郭炜　微博 http://weibo.com/guoweiofpku
http://blog.sina.com.cn/u/3266490431
刘家瑛 微博 http://weibo.com/pkuliujiaying

运算符重载实例：可变长整型数组

(教材P215)

```
int main() {  //要编写可变长整型数组类，使之能如下使用：
   CArray a; //开始里的数组是空的
   for( int i = 0;i < 5;++i)
            a.push_back(i);
   CArray a2,a3;
   a2 = a;
   for( int i = 0; i < a.length(); ++i )
            cout << a2[i] << " " ;
   a2 = a3; //a2是空的
   for( int i = 0; i < a2.length(); ++i ) //a2.length()返回0
            cout << a2[i] << " ";
   cout << endl;
   a[3] = 100;
   CArray a4(a);
   for( int i = 0; i < a4.length(); ++i )
            cout << a4[i] << " ";
   return 0;
}
```

要用动态分配的内存来存放数组元素，需要一个指针成员变量

要重载“=”

要重载“[ ]”

要自己写复制构造函数

程序输出结果是：

0 1 2 3 4
0 1 2 100 4

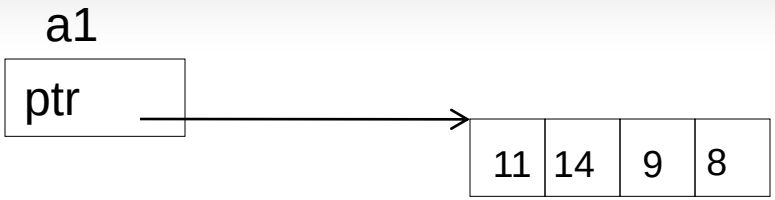
要做哪些事情？

```
class  CArray  {
    int size; //数组元素的个数
    int *ptr; //指向动态分配的数组
public:
   CArray(int s = 0);     //s代表数组元素的个数
   CArray(CArray & a);
   ~CArray();
   void push_back(int v); //用于在数组尾部添加一个元素v
   CArray & operator=( const CArray & a);
    //用于数组对象间的赋值
   int length() { return size; } //返回数组元素个数
   _____  CArray::operator[](int i)  //返回值是什么类型？
   {//用以支持根据下标访问数组元素，
    // 如n = a[i] 和a[i] = 4; 这样的语句
            return ptr[i];
    }
};
```

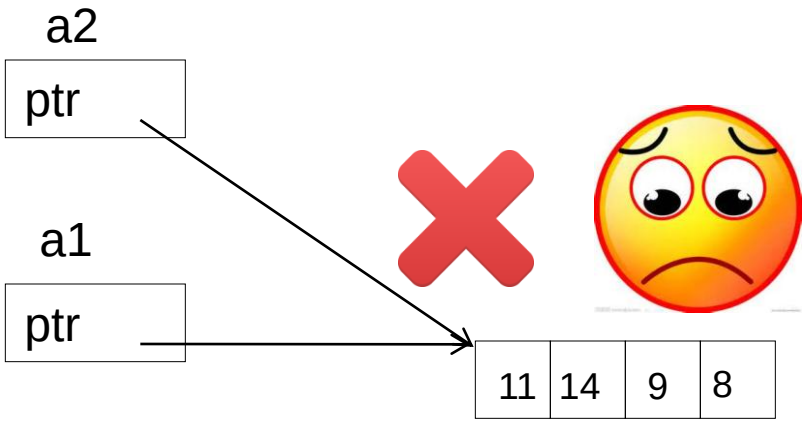
int &

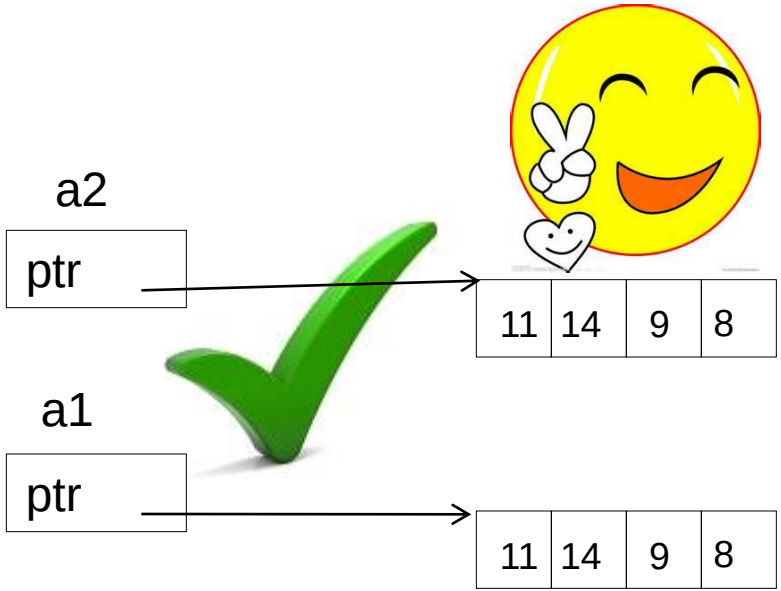
```
class  CArray   {
    int size; //数组元素的个数
    int *ptr; //指向动态分配的数组
public:
   CArray(int s = 0);     //s代表数组元素的个数
   CArray(CArray & a);
   ~CArray();
   void push_back(int v); //用于在数组尾部添加一个元素v
   CArray & operator=( const CArray & a);
    //用于数组对象间的赋值
   int length() { return size; } //返回数组元素个数
   int & CArray::operator[](int i)  //返回值为int不行!不支持a[i] = 4
   {//用以支持根据下标访问数组元素，
    // 如n = a[i] 和a[i] = 4; 这样的语句
            return ptr[i];
   }
};
```

```
CArray::CArray(int s):size(s)
{
        if( s == 0)
                ptr = NULL;
        else
                ptr = new int[s];
}
CArray::CArray(CArray & a)   {
        if( !a.ptr) {
                ptr = NULL;
                size = 0;
                return;
        }
        ptr = new int[a.size];
        memcpy( ptr, a.ptr, sizeof(int ) * a.size);
        size = a.size;
}
```



CArray a2(a1);





```
CArray::~CArray()
{
        if( ptr) delete [] ptr;
}
CArray & CArray::operator=( const CArray & a)
{ //赋值号的作用是使“=”左边对象里存放的数组，大小和内容都和右边的对象一样
  if( ptr == a.ptr) //防止a=a这样的赋值导致出错
        return * this;
  if( a.ptr == NULL) {  //如果a里面的数组是空的
        if( ptr )  delete [] ptr;
        ptr = NULL;
        size = 0;
        return * this;
  }
```

```
        if( size < a.size) { //如果原有空间够大，就不用分配新的空间
            if(ptr)
                delete [] ptr;
            ptr = new int[a.size];
        }
        memcpy( ptr,a.ptr,sizeof(int)*a.size);
        size = a.size;
        return * this;
} // CArray & CArray::operator=( const CArray & a)
```

```
void CArray::push_back(int v)
{  //在数组尾部添加一个元素
  if( ptr) {
            int * tmpPtr = new int[size+1]; //重新分配空间
            memcpy(tmpPtr,ptr,sizeof(int)*size); //拷贝原数组内容
            delete [] ptr;
            ptr = tmpPtr;
  }
  else  //数组本来是空的
          ptr = new int[1];
  ptr[size++] = v; //加入新的数组元素
}
```