继承与多态程序实例
– 虚函数

【例3-10】编写程序：定义一个员工类Employee，包含以下成员：

两个私有成员变量，分别为char型指针变量m_name（姓名）和m_no（员工号）；用于初始化成员变量的构造函数Employee(char *name, char *no)；用于销毁m_name和m_no所指内存空间的析构函数~Employee()；用于输出员工信息的Display()函数。

以Employee类作为基类，派生出领导类Leader，新增成员：

一个私有成员变量，char型指针变量m_posdes（职位描述）；用于初始化成员变量的构造函数Leader(char *name, char *no, char *posdes)；用于销毁m_posdes所指内存空间的析构函数~Leader()；用于输出领导信息的Display()函数。

例如，执行下面程序，可以在屏幕上输出：“张三 1011 软件开发部部门经理”。

```
int main()
{
    Employee *pe = new Leader("张三", "1011", "软件开发部部门经理");
    pe->Display();
    delete pe;
    return 0;
}
```

```
// Employee.h
#ifndef _EMPLOYEE_H
#define _EMPLOYEE_H
class Employee
{
protected:
    char *m_name;       // 姓名
    char *m_no;         // 员工号
public:
    Employee(char *name, char *no);          // 构造函数
    virtual ~Employee();                     // 析构函数
    virtual void Display();                  // 输出员工信息
};
#endif      // end of _EMPLOYEE_H
```

```cpp
// Employee.cpp
#include "Employee.h"
#include <iostream>
using namespace std;
Employee::Employee(char *name, char *no)
{
    m_name = new char[strlen(name)+1];
    m_no = new char[strlen(no)+1];
    strcpy(m_name, name);
    strcpy(m_no, no);
    cout<<"Employee构造函数被调用！"<<endl;
}
```

```cpp
Employee::~Employee()
{
    delete []m_name;
    delete []m_no;
    cout<<"Employee析构函数被调用！"<<endl;
}
void Employee::Display()
{
    cout<<"姓名："<<m_name<<endl
        <<"员工号："<<m_no<<endl;
}
```

```
// Leader.h
#ifndef _LEADER_H
#define _LEADER_H
#include "Employee.h"
class Leader : public Employee
{
private:
    char *m_posdes;     // 职位描述
public:
    Leader(char *name, char *no, char *posdes);
    ~Leader();
    void Display();
};
#endif       // end of _LEADER_H
```

```cpp
// Leader.cpp
#include "Leader.h"
#include <iostream>
using namespace std;
Leader::Leader(char *name, char *no, char *posdes)
    : Employee(name, no)
{
    m_posdes = new char[strlen(posdes)+1];
    strcpy(m_posdes, posdes);
    cout<<"Leader构造函数被调用！"<<endl;
}
```

```cpp
Leader::~Leader()
{
    delete []m_posdes;
    cout<<"Leader析构函数被调用！"<<endl;
}
void Leader::Display()
{
    cout<<"姓名："<<m_name<<endl
        <<"员工号："<<m_no<<endl
        <<"职位描述："<<m_posdes<<endl;
}
```

```cpp
// main.cpp
#include "Leader.h"
int main()
{
    Employee *pe = new Leader("张三",
"1011", "软件开发部部门经理");
    pe->Display();
    delete pe;
    return 0;
}
```

运行结果：

```
Employee构造函数被调用！
Leader构造函数被调用！
姓名：张三
员工号：1011
职位描述：软件开发部部门经理
Leader析构函数被调用！
Employee析构函数被调用！
```