***类间关联***

1．贼(Thief)偷窃行人，每个行人都有随身带的钱，贼偷窃一个行人，他的钱就增加相应的钱包里的钱数。 请给出Thief类和Walker类。

2. 警察局有多名警察，每个警察抓获一名贼，警察局的声望就增加1点，该警察的奖金就增加100元，贼的金钱减为0。

请实现相关的警察局、警察、贼的类。

用例可为：警察局S有警察p1,p2,p3,贼有t1,t2,t3,t4, p1抓获t2,t3, p2抓获t4, p3没抓获任何贼。t1的初始金钱为500，

t2的初始金钱为800，t3的初始金钱为300，t4的初始金钱为1000，S的初始声望为100，警察的初始奖金为0。

输出最终S的声望，每个警察的奖金数。

3．在2的基础上，再加上1中的类，试一试。

4．用简单**双向关联**和**关联类**的形式分别实现男人(Man)和女人(Woman)间的**一对一关系**。

一个未婚男人可以和一个未婚女人结婚；

一个已婚男人可以和其妻子离婚；

一个未婚女人可以和一个未婚男人结婚；

一个已婚女人可以和其丈夫离婚；

一个已婚男人可以“知道”其妻子；

一个已婚女人可以“知道”其丈夫；

5.现有类A和类B

class A

{

public:

A(int num):mData(num){ }

~A( ) { }

int GetData( ) const { return mData; }

void SetData(int data) { mData = data; }

private:

int mData;

};

class B

{

public:

B(int num=0):pa(new A(num)) { }

~B( ) {delete pa;}

B(const B& rhs)

{

pa=new A(\*rhs.pa);

}

B& ***operator=***(const B& rhs)

{

if ( this!=&rhs )

{

delete pa;

pa=new A(\*rhs.pa);

}

return \*this;

}

A\* ***operator->***( ) const {return pa;}

***~~void~~*** GetData() const { return pa->GetData();}***//错误，返回值应为int***

void SetData(int data) { pa->SetData(data); }

private:

A\* pa;

};

***a）***现需要以***引用计数的方法***，重新实现类B，要求类A不得做任何修改。

***b）***请在**a）的基础上**，以**Copy On Write的方式修改类B的实现**，使得B类对象可以访问B类的成员SetData(int);(即可以修改Ｂ类对象中pa指针指向的A类对象的数据成员。也就是说，使用B类对象时，对于以只读方式访问A类的成员，使用引用计数；对于以写方式访问A类的成员，要先进行深赋值，然后再写数据)。

6.若程序中需要频繁地用new创建、用delete销毁A类对象，请在A类中通过

重载operator new和operator delete，实时统计创建A类对象的个数、销毁的个数、

累计分配的字节数、还在使用的字节数。

若还可能使用new[],delete[],如何也能完成上边的统计工作。

7.通过分析二元运算符的交换律，以及左操作数的限制，理解

为什么重载+，一般用自由函数形式？

为什么重载+=，一般用成员函数形式？

为什么重载=，必须用成员函数形式？

为什么重载<<，必须用自由函数形式？

8.实现分页器类(Paginate),不用实现输入指定页的部分。

如：

对于如下主函数：

int main()

{

Paginate pager(13);

for(int i=1;i<=13;++i)

{

//i当前页，13总页数

pager.setPage(i).show();

}

cout<<"start move...."<<endl;

pager.setPage(5).show();

pager.next().show();

pager.prev().show();

//直接翻5页

pager.nextN().show();

pager.next().show();

pager.prevN().show();

return 0;

}

其输出为：(+表示是当前页)

上页 1+ 2 3 4 5 … 13 下页

上页 1 2+ 3 4 5 … 13 下页

上页 1 2 3+ 4 5 … 13 下页

上页 1 2 3 4+ 5 … 13 下页

上页 1 2 3 4 5+ … 13 下页

上页 1 … 6+ 7 8 9 10 … 13 下页

上页 1 … 6 7+ 8 9 10 … 13 下页

上页 1 … 6 7 8+ 9 10 … 13 下页

上页 1 … 6 7 8 9+ 10 … 13 下页

上页 1 … 6 7 8 9 10+ … 13 下页

上页 1 … 9 10 11+ 12 13 下页

上页 1 … 9 10 11 12+ 13 下页

上页 1 … 9 10 11 12 13+ 下页

start move....

上页 1 2 3 4 5+ … 13 下页

上页 1 … 6+ 7 8 9 10 … 13 下页

上页 1 2 3 4 5+ … 13 下页

上页 1 … 6 7 8 9 10+ … 13 下页

上页 1 … 9 10 11+ 12 13 下页

上页 1 … 6+ 7 8 9 10 … 13 下页