题目描述

现在你是某宝的一名工程师，被要求实现APP中的部分功能，包括创建新用户，创建新商品，用户浏览商品，用户购买商品等等，这些功能需要通过三个类
(Merchandise类，MerchandiseList类，User类)来实现。具体要求如下：

- Merchandise类
  - 你需要在自己定义的Merchandise类中至少实现以下接口

```
// 构造函数，初始化一个Merchandise类的对象
Merchandise::Merchandise(const char* name, int price);

// 成员函数，返回商品名
const char* Merchandise::GetMerchandiseName();

// 成员函数，返回商品价格
int Merchandise::GetMerchandisePrice();
```

  - 调用示例

```
Merchandise mer_pen = Merchandise("pen", 2); // 创建一个Merchandise对象，名为pen，价格为2元

Merchandise mer_apple = Merchandise("apple", 3) //  创建一个Merchandise对象，名为apple，价格为3元

mer_pen.GetMerchandiseName(); //返回商品名

mer_pen.GetMerchandisePrice(); //返回商品价格
```

- MerchandiseList类
  - 你需要在自己定义的MerchandiseList类中至少实现以下接口

```
// 构造函数，初始化一个MerchandiseList类的对象。
MerchandiseList::MerchandiseList();

// 成员函数，往MerchandiseList对象中增加一条记录(每条记录包括一个Merchandise指针，和一个数值)
void MerchandiseList::AddRecord(Merchandise *merchandise, int value);

// 成员函数，修改商品merchandise所在的那条记录中的value值为new_value，若商品不在列表中，不进行操作直接返回。
void MerchandiseList::ModifyRecord(Merchandise *merchandise, int
new_value);
```

```
// 成员函数，查找merchandise所代表商品的记录，并返回这条记录中的value值(若未找到记录则返回-1)
int MerchandiseList::FindRecord(Merchandise *merchandise);

// 成员函数，删除merchandise所代表商品的记录，成功删除后返回true，如果不存在该记录，则不做操作并返回false
bool MerchandiseList::DeleteRecord(Merchandise* merchandise);

/*
提示：
（1）对于一个MerchandiseList对象，它所保存的记录最多只有20条。
（2）若AddRecord(merchandise, value0)的参数merchandise已经在之前的记录中出现，则直接更新记录中的value值：
    new_value = old_value + value0
（3）判断两个Merchandise对象是否是代表同一种商品的依据是其名字（name）是否相同，名字相同视为同一种商品。
    如对象A的名为"pen"，B名为"pencil"，C名为"pen"，则A和C都是同一种商品。
（4）在MerchandiseList中，一个商品merchandise对应一个value，value可以认为是该商品的数量，不同的应用场景value会有不同的具体含义（在下面User类的设计中就会看到）。
（5）merchandise和value的对应关系可以通过多种方法实现，例如，定义一个结构，使用两个队列等，或者使用map结构。
（6）可以自己实现一些接口便于代码实现。
*/
```

- 调用示例

```
MerchandiseList merchandise_list; // 创建一个merchandise_list对象

merchandise_list.AddRecord(&mer_pen, 8) // 在merchandise_list中添加一条记录 (&mer_pen, 8)

merchandise_list.ModifyRecord(&mer_pen, 7) // 修改merchandise_list中的项（&mer_pen, 8）为 (&mer_pen, 7)

merchandise_list.FindRecord(&mer_pen) // 查找&mer_pen所在的项，并返回7

merchandise_list.DeleteRecord(&mer_pen) // 成功删除mer_pen，并返回true

merchandise_list.DeleteRecord(&mer_apple) // 因为mer_apple不在merchandise_list中，删除失败并返回false
```

- User类
  - 你需要在自己定义的User类中至少实现以下接口

```
// 构造函数，初始化一个User类的对象
User::User(const char* user_name);

// 成员函数，浏览商品(使用上面定义好的MerchandiseList类记录浏览情况)
```

```
void User::BrowseMerchandise(Merchandise* merchandise);
/*
注意：
出现在浏览记录MerchandiseList的商品，value代表浏览次数，如：第一次浏览value为1，第二次再浏览value为2。
调用BrowseMerchandise()一次代表浏览对应商品一次。
*/

// 成员函数，返回浏览记录
MerchandiseList* User::GetBrowseHistory();

// 成员函数，购买商品(使用上面定义好的MerchandiseList类记录购买情况，number代表购买的数量)
void User::BuyMerchandise(Merchandise *merchandise, int number);
/*
注意：
出现在购买记录MerchandiseList的商品，value代表购买数量，如：购买3个商品value为3
*/

// 成员函数，返回购买记录
MerchandiseList* User::GetBuyHistory();
/*
注意：
（1）请注意浏览商品记录和购买商品记录的区别
（2）购买记录只包含购买数量大于等于1的商品
*/
```

- 调用示例

```
User user_Tom = User("Tom"); // 创建一个User对象，用户名为Tom

user_Tom.BrowseMerchandise(&mer_pen); // 浏览商品pen一次

user_Tom.GetBrowseHistory(); // 返回用户Tom的浏览记录

user_Tom.BuyMerchandise(&mer_apple, 6); // 购买6个apple

user_Tom.GetBuyHistory(); // 返回用户Tom的购买记录

user_Tom.BuyMerchandise(&mer_apple, 2) // 再次购买2个苹果

user_Tom.GetBuyHistory(); // 返回再次购买后的购买记录
```

Tips:

- 除了上述要求实现的接口外，可以适当地设计自定义的类接口以方便代码实现。
- 某些类的成员变量需要考虑如何恰当地初始化。

- 通常将类的定义写在头文件.h中，将成员函数的实现写在源文件.cpp中，并注意成员访问控制修饰符的使用(public, private等)。

注意⚠️

- 请正确处理头文件和实现文件之间的关系，文件、函数的命名严格按照给定要求，注意大小写。
- 将6个文件(merchandise.cpp, merchandise.h, merchandise_list.cpp, merchandise_list.h, user.cpp, user.h)打包成ZIP压缩包上传(ZIP包中不要包含中间文件夹或者其他文件)。
- 请不要在你提交的代码中包含main函数。