**Object-Oriented Programming II Spring 2017**

**CIS 9410**

# Programming Assignment III

For this assignment you are to implement 2 classes that will be used for implementing an inventory management system. One class, **Inventory**, is used to model an inventory of objects associated with a warehouse or other outlet. Associated with an inventory is a collection of inventory items. These items are modeled by the class **InventoryItem**, the second class that you are to implement. The specification of the 2 classes are given below.

**InventoryItem** Class

The **InventoryItem** class is used to model items that can be part of an inventory. The items are characterized by a unique item identification number, an item name, a unit price, a quantity on hand, as well as the total value of that item in the inventory, which is computed as the product of the quantity on hand and the unit price.

The **InventoryItem** class consists of the following functions:

*Constructors*:

**InventoryItem();**

The default constructor instantiates an **InventoryItem** object in which all of the numeric data members have a value of 0, and all of the **String** members have an empty (null) string value.

**InventoryItem(int idno, String nme, double uprc, int qty);**

This initialization constructor instantiates an **InventoryItem** object with an identification numberf of **idno**, an item name of **nme**, a unit price of **uprc**, and a quantity on hand value of **qty**.

*Accessor Functions*:

**int getID() const;**

**String getName() const;**

**double getUnitPrice() const;**

**int getQuantity() const;**

**double getValue() const;**

The value of an item in an inventory is computed as the product of the item’s unit price and quantity of that item in the inventory.

**bool operator==(InventoryItem itm);**

Returns *true* if the corresponding data members of the two operands are equivalent, otherwise returns *false*.

*Mutator/Modifier Functions*:

**void setID(int idno);**

**void setName(String nme);**

**void setUnitPrice(double uprc);**

**void setQuantity(int qty);**

*Input/Output Functions*:

**void Read();**

**void Display();**

**Inventory** Class

The **Inventory** class is used to model an inventory, which consists of some number of inventory items. Each inventory is associated with some “owner,” which, for example, could be a warehouse or retail outlet. The items in an inventory must have unique identification numbers, and no restriction should be placed on the number of items in an inventory.

The **Inventory** class consists of the following functions:

*Constructors/Destructor*:

**Inventory();**

This constructor instantiates an **Inventory** object in which there are no inventory items, and the value of the owner is a null **String** value.

**Inventory(String ownr);**

This initialization constructor instantiates an **Inventory** object in which there are no inventory items, and for which the owner is specified by **ownr**.

**~Inventory();** (If necessary)

*Accessor Functions*:

**String getOwner() const;**

**int getItemCount() const;**

This function returns the count of the total number of *unique* items in the inventory, not the total quantity of the items.

**double Value() const;**

This function returns the total/aggregate value of all of the items in the inventory.

**InventoryItem Find(int idno);**

This function returns the **InventoryItem** object having the identification number specified by **idno**. If there is no item in the inventory with the specified identification number, then the function should return the *globally-defined* “null” constant **InventoryItem** object **NULL\_ITEM**.

**void Display();**

This function displays the name of the owner of the inventory, along with the data associated with each of the items in the inventory. The displayed data should be organized in a user-friendly way.

*Mutator/Modifier Functions*:

**void setOwner(String ownr);**

**bool Insert(InventoryItem itm);**

This function inserts into the inventory the inventory item specified by **itm** if there is not item in the inventory with the same identification number. The items in the inventory should be maintained in identification number order. If the item is successfully inserted, then the function returns *true*. If there is already an item with the same identification number, the item is not inserted, and the function returns *false*.

**bool Remove(int idno);**

If there is an item in the inventory with the specified identification number, then that item is removed, and the function returns *true*. If there is no item in the inventory with the specified identification number, then the function returns *false*.

The **Inventory** and **InventoryItem** classes should be specified in the file *inventory.h* and, where appropriate, their member functions should be defined in the file *inventory.cpp*. In addition, your code should incorporate the files *mystring.h* and *mystring.cpp* for the **String** class.

Where appropriate you may specify additional class that can used for implementing the required classes.

In addition to the classes specified above, you should create a small application to test your code to ensure that it conforms to the specification. I will also supply my own test code.

Due Date: March 30, 2017