CS 201: Problem Solving & Programming II

Lab # 8

The warm up

Remember we worked with an item class in midterm exam. We will add a friend function in it and do some operator overloading to make life easy!

So here’s the diagram of the item class, once again. Don’t worry about the last function, it will be covered later in exercise 1.

|  |
| --- |
| **Item** |
| - m\_name : string  - m\_price : double |
| +Item()  +Item(name: string& const, price : double)  +getPrice() : double  +getName(): string  +print(Item& item) : void friend;  **+ operator<<(out : ostream&, item: Item&) : ostream& friend** |
| **Note: Creating a friend function**  We can declare a function as friend using the keyword *friend* before the function declaration. For example:  class Item {  public:  friend void print();  } |

**The task**

1. Define an Item class.
2. Define the friend function ***print()*** to print the attributes of item class. For example, if you declare a item object with name “Apple” and price “1.30” calling the print() will output:

Item: Apple, price: $1.30

1. Create a main.cpp file and create an object of Item and call its print() function. It should print the object as stated above.

Move to the next section if you are done.

Exercise 1

Now instead of using the print function you have to use operator overloading to overload the << operator to achieve the same output.

**Overloading an operator**

To overload the functionality of an operator, define a method where you have to specify what the operator will do when called. For example, another friend function of Item class can be defined to overload the << operator:

friend ostream& operator<<(ostream& out, Item& item) {

// the operation that the << operator will do …

}

**The task**

1. Add the overloading function to Item class.
2. Define the function to print the same output as print function did.
3. Call the << operator to print a item object like this:

Item item(string(“apple”, 1.30);

cout << item;

It should print in the console:

Item: Apple, price: $1.30.

Exercise 2

**Some more tasks using operator overloading**

Suppose we have to read the items from a file. To read a value from a file, generally we use >> operator like this:

double price;

Ifstream fin(“filename.ext”);

fin >> price;

Now that we know overloading we can exploit it to read one object at a time like this:

Item item;

fin >> item;

So instead of reading the item name and price in two separate statements, we can read the whole object at once and the code looks nice too. **In this exercise you have to write that overloading function to achieve this task.** The function declaration can be as follows (it can be defined in main.cpp and doesn’t have to be a friend) -

**void operator>>(istream& in, Item& item);**

Also you have to define two more functions in the Item class:

**void setName(string name);**

**void setPrice(double price);**

A pseudo-code of the overloading function is given as a hint:

**void operator>>(istream& in, Item& item) {**

**// read the first line of the file using the input stream**

**string name;**

**getline(in, name);**

**//read the price next**

**. . .**

**//set the name in item object**

**item.setName(name);**

**. . .**

**}**

Once you complete the function, you can use it to read an object from the file using the statement:

fin >> item;

A sample file “**items.txt**” is given to read from.