Iterators and Friends

Due Tuesday, May 05 at 8 a.m.

CSE 1325 - Spring 2020 - Homework #11 - 1

Assignment Overview

This assignment is somewhat different, in that the Full Credit, Bonus, and Extreme Bonus levels are unrelated. It focuses on topics that you can expect to see on Exam #3.

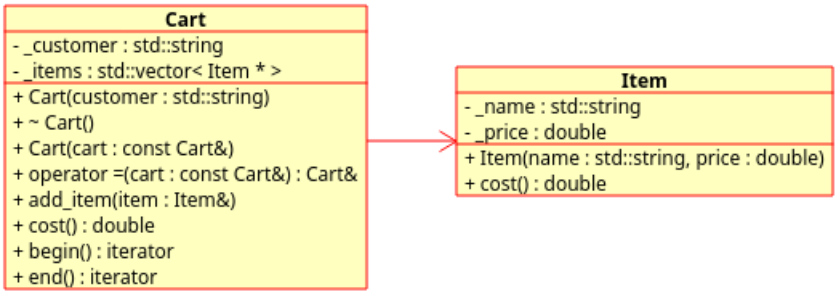
Full Credit is a simple shopping cart problem. It exercises custom iterators, string manipulation, rethrowing an (improved!) exception, and the Rule of 3.

Bonus is a relatively short main program that collects words from a file, then prints out all words with a specified frequency. It exercises the std::map container, std::pair, and the std::find_if algorithm.

Extreme Bonus is student-defined!

Full Credit

Implement the Item and (shopping) Cart classes shown in the class diagram per the following details.



Item

- **Constructor** - This is a basic "data bag" class, so just initialize the attributes from the parameters.
- **cost()** - This is a getter for _price. (We don't need a getter for _name, since we're overloading the streaming in operator.)
- **operator<<** - Stream out a dollar sign, the _price in fixed notation with precision 2 (e.g., $12.95), and the name of the item. It's fine to have spaces between the $ and number so that the decimal points align. That how I wrote the suggested solution.
- **operator>> - This is a bit of a challenge. Read a newline-terminated string. The *last* whitespace-separated word in that string is the price - convert it to a double and store as _price. The *rest* of the words are the name of the item, so store the rest of the string in _name.**
    - If the last word on the input line throws an exception when passed to std::stod, catch that exception and throw another. Include as the parameter a brief error message and the offending line of input, with the last word marked in some way.
    - You may use any string manipulations you like, e.g., find_last_of and find_last_not_of, or use a regular expression, or parse into a vector and reassemble all but the last, or use

getline with a delimiter, or search one char at a time - the objective is for you to practice some Lecture 23 string manipulation. For best benefit, try it several different ways!)

Cart

- **Constructor** - Just store the customer's name in _customer.
- **Destructor** - Delete all of the items in _items from the heap.
- **Copy Constructor** - Do a "deep copy", that is, clone each item on the heap, and set the destination's pointer to that while the source's pointer continues to point to the original.
- **Copy assignment operator / operator=** - Same as the copy constructor, but remember that if `this==&cart` , do nothing! Thats the `cart1=cart1;` case.
- **Iteration** - Define iterator, const_iterator, begin(), and end() just as we showed you in Lecture 22, so that you can iterate over a cart object and get back each item from the vector.
- **add_item(item)** - Clone the item onto the heap, and push a pointer to it onto _items.
- **cost()** - Add up the cost of each item in _items, and return the total.
- **Note:** You many NOT define operator<< for Cart. The point of this problem is to iterate over the Cart object in main().

Main

- Instance a Cart with any customer name you like, and an empty Item (e.g., "" and 0 parameter).
- Then loop, reading in new items from std::cin *using Item's operator>>* until end of file, pushing each item onto the Cart instance. If an exception occurs, print e.what() to cerr and try again.
- Once the end of file is reached, *use Cart's iteration capability that you defined above* to print out a table of each item in the cart, followed by the total cost (ignoring tax).

Add, commit, and push all files.

```
ricegf@pluto:~/dev/cpp/202001/P11/full_credit$ make
g++ --std=c++17 -c main.cpp -o main.o
g++ --std=c++17 -c cart.cpp -o cart.o
g++ --std=c++17 -c item.cpp -o item.o
g++ --std=c++17 main.o cart.o item.o -o cart
ricegf@pluto:~/dev/cpp/202001/P11/full_credit$ ./cart
Enter product names and price (e.g., "English peas 0.79"):
Rome apples 1.49
Del Monte Diced Tomatoes 2.29
Bananas .89
Special of the Day free
Invalid price: 'Special of the Day' ==> 'free'

Register Receipt

$    1.49  Rome apples
$    2.29  Del Monte Diced Tomatoes
$    0.89  Bananas
--------------------
$    4.67  Total Cost
ricegf@pluto:~/dev/cpp/202001/P11/full_credit$
```

Bonus

Write a main function that accepts a filename as its sole, mandatory parameter. Properly report an error and exit if the parameter is missing or isn't a valid filename.

Otherwise, open the file and load every whitespace-separated word in it as a key for a std::map, with the value being the number of occurrances of that word. For example, if "the" occurs 6 times in the document, the key for that pair is "the" and the value is 6.

One the file has been read, use the map to print out every word in alphabetical order along with the number of occurances.

Finally, enter a loop, accepting an int representing the number of occurances. If the int is 0 (an invalid number of occurances), exit the program successfully. Otherwise, use std::find_if to print out every word with that number of occurrances.

Add, commit, and push all files.

```
ricegf@pluto:~/dev/cpp/202001/P11/bonus$ make -j
g++ --std=c++17 -c main.cpp -o main.o
g++ --std=c++17 main.o -o words
ricegf@pluto:~/dev/cpp/202001/P11/bonus$ ./words dr_seuss.txt
BUMP!           1
I               3
Sally           1
all             3
and             4
at              1
ball            1
bit             1
cold            3
could           1
day             1
did             3
do              2
go              1
had             1
house           2
how             1
in              2
it              2
like            1
little          1
not             3
nothing         1
one             1
out             1
play            2
said            1
sat             4
shine           1
sit             4
so              3
something       2
sun             1
that            1
the             3
then            1
there           1
there,          1
to              5
too             3
two             1
was             2
we              8
went            1
wet             3
wish            1
with            1
==================================
List words with which frequency? 5
to              5
==================================
List words with which frequency? 4
and             4
sat             4
sit             4
==================================
List words with which frequency? 6
==================================
List words with which frequency?
```

Extreme Bonus

Go for broke! Demonstrate *any* major C++ capability that we did **not** cover this semester, e.g., you might

- Demonstrate the priority queue algorithm from the standard library.
- Use a functor to create a map that that automatically sorts (and thus iterates) in reverse order.
- Demonstrate move semantics or smart pointers.
- Demonstrate a true gtkmm table.
- Create a custom gtkmm widget.

What did you always want to explore? Explore it for extreme bonus points!