Jeremy Prater

CS 162

Assignment 1 Design Document

Class for listItem

- String name
- String unit (pound, oz, can, etc...)
- Double unit price
- Double unit quantity

Constructor takes unit price, name, unit, and quantity

Functions needed

- Get name
- Get unit
- Get unit price
- Get quantity
- Get total price

Class for groceryList

- listItem ** currentList
    - Dynamically allocated pointer list
    - Contents contain pointers to class objects of (listItem*)
    - Pointers are allocated with new/delete
- Uint32 itemCount - Start size of listItem 4
    - Upon adding nth element, reallocate size to 2n

Functions needed:

- CreateList
- AddItem
    - Function takes (name, unit, qty, unit price)
- RemoveItem
- DisplayItems

<u>Testing</u>

The test suite involves the following cases:

- Remove all items from an empty groceryList
    - Check for null pointer access
- Get total price from empty list

        - Check for null pointer access
    - Add 6 items
        - Check for array resize on each item
    - Remove out of index list item
        - Check that count does not go down
    - Remove item 1
        - Verify that Tomatoes are not on final list
    - Print list
        - Verify contents
    - Remove all items from list
        - Verify count is zero
    - Add one item
        - Verify count is one
    - Delete class and exit.

The program was verified to have no memory leaks by using the following command.

```
> valgrind ./groceryList

==10452== Memcheck, a memory error detector

==10452== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et
al.

==10452== Using Valgrind-3.11.0 and LibVEX; rerun with -h for
copyright info

==10452== Command: ./groceryList

==10452==

==10452==

==10452== HEAP SUMMARY:

==10452==     in use at exit: 1,768,281 bytes in 334 blocks

==10452==   total heap usage: 353 allocs, 19 frees, 1,777,645 bytes
allocated

==10452==

==10452== LEAK SUMMARY:

==10452==    definitely lost: 0 bytes in 0 blocks

==10452==    indirectly lost: 0 bytes in 0 blocks

==10452==      possibly lost: 0 bytes in 0 blocks
```

```
==10452==    still reachable: 1,768,281 bytes in 334 blocks

==10452==         suppressed: 0 bytes in 0 blocks

==10452== Rerun with --leak-check=full to see details of leaked memory

==10452==

==10452== For counts of detected and suppressed errors, rerun with: -v

==10452== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from
0)
```

<u>Reflection</u>

I ran into a particularly difficult bug which took me around 3 hours to triage. I had to use gdb and set a memory watch point on memory and try to find the root cause of why it was being modified.

The root cause was that I was only decrementing the list count inside of an ‘if’ block that checked if the list item to delete was not the last one. The purpose of the ‘if’ block is to join the ‘tail’ of list item pointers back on to the rest of the list, which prevents memory fragmentation holes since the list is just an indexed list.