Functionality				
Component	Description	Point penalty	Actual	Comments
Fails to compile	The program fails to compile. In this case it is difficult to evaluate the functionality so no other functional components are considered.	-75		
Program crashes or hangs	The program crashes or gets into an infinite loop. In this case it is difficult to evaluate the functionality so no other functional components are considered.	-50		
Compiles with warnings	The program compiles with warnings but it runs without crashing or hanging.	-5	-5	
Major memory leaks	More than one memory leak. Usually due to missing or improperly implemented destructor.	-10		
Minor memory leak	Just one (or two, if related) small memory leaks. Usually due to forgetting to delete a temporarily allocated item.	-5		
Major output mismatch	There are several missing lines of output, they are out of order, or they are plain wrong.	-10		
Minor output mismatch	Missing one or two lines, pervasive punctuation, spelling or capitalization errors.	-5		
Trivial output mismatch	A single trivial error in punctuation, spelling or capitalization errors.	-2		
Missing feature	A feature listed in the requirements is not implemented. This will likely also cause a major output mismatch. This penalty is "per feature".	-5		
			-5	
Functionality total:	95			
Design				
Component	Description	Points	Actual	
Public data members	You should not have any public data members in your classes. Use getters/setters to read/write these members.	-10	-10	
Poor identifier names	Make sure your classes, methods, variables, etc. are named with sensible names.	-5		
Poor cohesion/coupling	Your program design should make sure that each method or function just does one thing and that each class is well encapsulated.	-10		
Repeated code	You shouldn't find yourself repeating the same lines of code over and over again. If you do then you need to refactor to a helper function/method. It is common for people to repeat code in the copy constructor and assignment operator. Try to avoid this.	-5		
Overly complex control structure	Try to use simple and straight-forward control statements (if, for, while, etc.)	-5		

Too clever code	Your code is overly clever which makes it hard to understand and even harder to debug.	-5		
Using char[] for strings	It is a design flaw to use fixed size character arrays for strings. This can very easily lead to character buffer overflow problems that you will only discover at runtime with specific data.	-5		
Missing copy constructor and/or assignment operato	Any class that needs a destructor also needs a copy constructor and assignment operator. The reason is because you (the class author) don't have control over how the client code will use your class. Without a copy constructor you are susceptible to problems if the client code making copies of your class and destroying the copy. The same reasoning applies to the assignment operator.	-5	-5	
Break statements	While the C++ language allows for break statements in loops, it is better design practice to avoid this. In nearly all cases a break can be avoided.	-5		
Multiple return statements	Each method in your classes should have no more than one return statement. There is a exception to this when there is a simple if/return at the very beginning of the method. It is not technically illegal to have multiple returns but it is better design to have a single one.	-5	-5	
Embedded linked list functionality	It is best design to have the linked list implemented as a separate class rather than embedded into the inventory class. This supports better cohesion and modularity.	-10	-10	
Changing the case of the data	Your code ends up converting the item to lower case while it puts it in the inventory. This is not correct. The program shouldn't change the capitalization. You should do the case conversion only for comparisons.	-5		
			-30	
Design Total:	70			
Days late	0			
Overall Grade	82.5			