Assignment 2		Student: Austin Major	
Max Value	**Earned Value**	**Check Marks**	**Instructor Comments**
Submission (10)			
5	5	Source/header filenames correct and named using UpperCamelCase	
5	5	Comment lines identify filename, author, student ID, assignment #	
Programming Style (30)			
5	5	Exhibits continuity of style throughout project in accordance with the Linux Kernel Coding Style and any additional requirements included in this course's style guide • Use of open/close parenthesis (), no parentheses between function name and opening parenthesis and a single space between operand and opening parenthesis of expression • Use of open/close curly braces {}, opening curly brace for functions always on newline, everywhere else always at end of line with a single space preceding curly brace • Use of open/close brackets [] • Use of indentation, one level of indentation after each opening curly brace until closing curly brace • Use of blanks lines to separate functions, sections of code (multiple blank lines condensed to a single line) • Spacing around operands (*if-else, while, do-while, for, switch*) • Spacing around binary and unary operators	
5	5	Identifier names • Function and variable names are meaningful and are named using lowerCamelCase, short names (one char in len) are allowed for simple counter variables • Classes are named using UpperCamelCase • A constant variable is used instead of an explicit value when the value is subject to change or used repeatedly	
5	5	Scope • Variables declared with minimal scope (declared in the innermost block required and no global declarations)	
5	5	Flow control • Exhibits good use of flow control and loop statements (*if-else, while, do-while, for, switch*)	
5	5	Classes and functions • Classes demonstrate OOP principle of data encapsulation • Classes include default constructor and overloaded constructor	

		allowing member initialization via the initialization section • Functions perform specific tasks (black-box style) • Constants for a class are either defined in the private section of the class or are limited in scope to the class implementation itself	
5	5	Commentary • Exhibits good use of commentary throughout header and source files, comments are meaningful • Functions are commented • Logical groups of statements are commented	
Accuracy (40)			
20	18	Source files compile without syntax errors (use –Wall flag) and include sufficient logic to produce expected outcomes • The main() includes a loop for inputting individual tickets where each ticket is initialized using the overloaded ticket constructor (passing input values as arguments) and added to a list of tickets via the push_back() function of the vector class • The main() includes a separate loop for processing the list of tickets where each ticket's values are outputted and gross/net bushels are accumulated • The main() includes the outputting of the accumulated totals after processing the list of tickets	Did not initialize gross/net bushel variables before using (accumulating) yielding unpredictable results for output.
10	10	Program runs without causing a run-time error using the "happy path", only valid values (conversion errors are tolerated for this assignment) and exhibits sufficient logic to produce expected outcomes	
10	10	Program's input and output are user-friendly, easy to understand and use and exist in the same file as the main() function. A specialized class should not perform input and output.	
Test Cases for Input (10)			
5	5	Input allows an empty ticket number (or a special ticket number) to indicate no more tickets	
5	5	Input prompts for ticket information using same requirements as assignment 1	
Test Cases for Output (10)			
5	4	Output shows ticket information (receipt) for each ticket using same requirements as assignment 1	Using foreign material dockage value instead of foreign material percentage for the percentage.
5	5	Output shows summary of total gross and net bushels for all tickets	
100	97.00	**Total**	Well done overall.