Wichita State University

CS 311: Object Oriented Programming
Spring 2018 – Assignment 2

For this assignment, we will improve upon the program implemented in assignment 1 by introducing constructor(s) for the *Ticket* class as well as allowing a list of tickets using the STL class *vector*. In addition, any issues raised in the feedback you received for assignment 1 must addressed (fixed) as part of this assignment.

Use the following criteria for writing your program:

- ✓ Modify the program named *WheatHarvest* (source file named WheatHarvest.cpp) to:
  - Declare an object variable of type *Ticket* using the *Ticket* class constructor allowing arguments to initialize its member variables using a class constructor initialization section
  - Declare a list of *Ticket* objects using the STL *vector* class. After initializing a single Ticket object, add the object to the vector list. This list is dynamic and may contain 0 or more tickets
  - Output the list of tickets while accumulating totals for gross and net bushels of each ticket
  - Output total gross and net bushels of all tickets

- ✓ Modify class named *Ticket* (header file named Ticket.h, source file named Ticket.cpp) to:
  - Include a no-argument constructor
  - Include a constructor allowing arguments for initializing its member variables and then initialize the member variables using an initialization section
  - Remove any unused member variables
  - Remove any unused mutator/accessor functions that set/get member variables unnecessarily

To test the changes, repeatedly prompt for the ticket information (ticket number, gross and tare weights, moisture level, and foreign material) until user decides there are no more tickets (allow for user case in which no ticket is provided). For each ticket provided, initialize an object of type *Ticket* using the constructor allowing arguments for initializing member variables, then add the ticket to the list of tickets. Next (in a separate loop from entering tickets), process the list of tickets, outputting the ticket information in the same manner as defined in assignment 1 while also accumulating totals for each ticket's gross and net bushels. Finally, output the total gross and net bushels of all tickets.

Example Input:

```
    Ticket number 1 (alphanumeric): 101300A
    Gross weight (lbs) 33180
    Tare weight (lbs): 10780
    Moisture level (%): 14.0
    Foreign material (%): 0.75

    Ticket number 2 (alphanumeric): 101400A
    Gross weight (lbs) 24150
    Tare weight (lbs): 10780
    Moisture level (%): 11.5
    Foreign material (%): 1.25

    Ticket number 3 (alphanumeric): <nothing entered, user just presses Enter>
```

Example Output (based on input):

```
    Ticket 101300A:
          33180   Gross Weight
          10780   Tare Weight
          22400   Net Weight

         373.33   Gross Bushels
           7.47   Moisture Level (14%)
           2.80   Foreign Material (0.75%)
         363.07   Net Bushels

    Ticket 101400A:
          24150   Gross Weight
          10780   Tare Weight
          13370   Net Weight

         222.83   Gross Bushels
           0.00   Moisture Level (11.5%)
           2.79   Foreign Material (1.25%)
         220.05   Net Bushels

    Wheat Harvest Summary Totals
         596.17   Gross Bushels
         583.11   Net Bushels
```

Requirements and Submission:

- Due date: Saturday, January 20, 11:59pm
- Include the following header information (comment lines) at the beginning of each source/header file (before the #include directives) submitted. Notice each comment line begins with two slashes. C++ recognizes these as the start of single line comment. If you prefer, you may use block commenting.

```
// File Name: WheatHarvest.cpp          Such as: // File Name: WheatHarvest.cpp
// Author: Firstname Lastname                    // Author: Joe Shobe
// Student ID: ********                          // Student ID: xxxxxxxx
// Assignment Number: ##                         // Assignment Number: 2
```

- Grades will be based on:
  - ✓ User-friendly interactive input, output, and error messages (when applicable)
  - ✓ Programming style: Proper use of control statements, standard C++ techniques and naming conventions, and the conciseness and readability of the code
  - ✓ Accuracy: Does the program calculate and output correct data
  - ✓ Appropriate and descriptive comments
- Use of solutions found on the web, in the books, or from other students is not permitted. It must be your own work. Under no circumstances are you to share your work with another student.
- Review the section *Submitting Assignments* in the Linux Server PDF for more information on how to turn in your assignment.
- No submissions will be accepted by Blackboard, e-mail, CD, DVD, or other independent storage or electronic media, unless approved by the instructor.