1.

Write a grading program for a class with the following grading policies:

a. There are two quizzes, each graded on the basis of 10 points.

b. There is one midterm exam and one final exam, each graded on the basis of 100 points.

c. The final exam counts for 50% of the grade, the midterm counts for 25%, and the two quizzes together count for a total of 25%.(Do not forget to normalize the quiz scores. They should be converted to a percentage before they are averaged in.)

Any grade of 90 or more is an A, any grade of 80 or more(but less than 90) is a B, any grade of 70 or more (but less than 80) is a C, any grade of 60 or more (but less than 70) is a D, and any grade below 60 is an F. The program will read in the student's scores and output the student's average numeric score for the entire course and final letter grade. Define and use a structure for the student record.

```
Data for the test run:
1  7  10  90 95
2  9  8   90 80
3  7  8   70 80
4  5  8   50 70
5  4  0   40 35
```

2.

Write the definition for a class named GasPump to be used to model a pump at an automobile service station. Before you go further with this programming exercise, write down the behavior you expect from a gas pump from the point of view of the purchaser.

The following are listed things a gas pump might be expected to do. If your list differs, and you think your list is as good or better than these, then consult your instructor. You and your instructor should jointly decide what behavior you are to implement. Then implement and test the agreed upon design for a gas pump class.

a. A display of the amount dispensed

b. A display of the amount charged for the amount dispensed.

c. A display of the cost per gallon, liter, or other unit of volume that is used where you reside.

d. Before use, the gas pump must reset the amount dispensed and amount charged to zero.

e. Once started, a gas pump continues to dispense fuel, keep track of the amount dispensed, and compute the charge for the amount dispensed until stopped.

f. A stop dispensing control of some kind is needed.

Implement the behavior of the gas pump as declarations of member functions of the gaspump class, then write implementations of these member functions. You will have to decide if there is data the gas pump has to keep track of that the user of the pump should not have access to. If so, make these private member variables.

3.

Your Community Supported Agriculture (CSA) farm delivers a box of fresh fruits and vegetables to your house once a week. For this Programming Project, define the class BoxOfProduce that contains exactly three bundles of fruits or vegetables. You can represent the fruits or vegetables as an array of type string. Add accessor and mutator functions to get or set the fruits or vegetables stored in the array. Also write an output function that displays the complete contents of the box on the console.

Next write a main function that creates a BoxOfProduce with three items randomly selected from this list:

Broccoli
Tomato
Kiwi
Kale
Tomatillo

This list should be stored in a text file that is read in by your program. For now you can assume that the list contains exactly five types of fruits or vegetables.
Don't worry if your program randomly selects duplicate produce for the three items.
Next, the main function should display the contents of the box and allow the user to

substitute any one of the five possible fruits or vegetables for any of the fruits or vegetables selected for the box. After the user is done with substitutions output the final contents of the box to be delivered.

4.

Redefine the implementation of the class Day described in Programming Project 1 (or do the definition for the first time, but do the implementation as described here). This time the day is implemented as three member variables of type char that store the first three letters in the name of the day. Embed your definition in a test program.

5.

Write a program that outputs a histogram of grades for an assignment given to a class of students. The program should input each student's grade as an integer and store the grade in a vector. Grades should be entered until the user enters -1 for a grade. The program should then scan through the vector and compute the histogram. In computing the histogram the minimum value of a grade is 0 but your program should determine the maximum value entered by the user. Output the histogram to the console. See Programming Project 5.7 for information on how to compute a histogram.

6.

First, complete the HW3-3 Modify the main function with a loop so that an arbitrary number of BoxOfProduce objects are created and substitutions are allowed for each box. Add a menu so the user can decide when to stop creating boxes.

You would like to throw in a free recipe flyer for salsa verde if the box contains tomatillos. However, there are only 5 recipe flyers. Add a static member variable to the BoxOfProduce class that counts the number of recipe flyers remaining and initialize it to 5. Also add a member variable that indicates whether or not the box contains a recipe flyer and modify the output function to also print "salsa verde recipe" if the box contains a recipe flyer. Finally, add logic inside the class so that if the box contains at least one order of tomatillos then it automatically gets a recipe flyer until all of the recipe flyers are gone. Note that a box should only get one recipe flyer even if there are multiple orders of tomatillos.

Test your class by creating boxes with tomatillos from your menu until all of the flyers are gone.