***CIS 311 - Assignment 5***

Your next job, working at CyberTech Manufacturing is to design a new sales reporting system. You will need to use the ideas of generics and LINQ in order to complete this assignment. There is an unsorted file called commiss.txt that has the following format:

John Fredricks 102 70.00 90.00 104.00 305.00 408.00

Jane Williams 104 89.00 105.00 70.00 400.00 207.00

Each line in the file consists of an employee’s first name, last name, employee ID number, Monday sales amount, Tuesday sales amount, Wednesday sales amount, Thursday sales amount and Friday sales amount. You will build a class (it doesn’t have to be a properly designed class, e.g. don’t worry about using getters/setters, just do what was shown in the latest chapter) which will store the following information on an employee:

strFirstName As String

strLastName As String

intID As Integer

sngMonSales As Single

sngTueSales As Single

sngWedSales As Single

sngThuSales As Single

sngFriSales As Single

sngTotalSales As Single

You may only store this information on each employee – nothing more… You will then use an appropriate generics data container to hold all of the employee instances that you create as you are reading the records in from the text file. These instances should be stored in employee ID order. You will then produce the following report (only a portion is shown here):

Cybertech Manufacturing

\*\*\* Sales Report by ID \*\*\*

-----------------------------

Name ID Mon Sales Tue Sales Wed Sales Thu Sales Fri Sales Total Sales

----------------- --- --------- --------- --------- --------- --------- -----------

Fredricks, John 102 $70.00 $90.00 $104.00 $305.00 $408.00 $977.00

Howard, Scott 103 $100.00 $100.00 $100.00 $100.00 $100.00 $500.00

.

.

.

Verski, Frank 416 $102.00 $105.00 $107.00 $202.00 $148.00 $664.00

Using LINQ, turn around and produce the same report again, but this time use LINQ to send the data out in last name order:

Cybertech Manufacturing

\*\*\* Sales Report by Name \*\*\*

-----------------------------

Name ID Mon Sales Tue Sales Wed Sales Thu Sales Fri Sales Total Sales

----------------- --- --------- --------- --------- --------- --------- -----------

Ashen, Craig 176 $202.00 $102.00 $104.00 $105.00 $207.00 $720.00

Bethere, Al 346 $89.00 $105.00 $70.00 $400.00 $207.00 $871.00

.

.

.

Zeusche, Judy 347 $103.00 $203.00 $203.00 $205.00 $306.00 $1,020.00

Once you’ve printed the sorted, formatted data from your generics container of class instances, you will use LINQ to tabulate the following statistics from your generics container – again, I have not shown all of the actual output, just enough to help out understand what needs to be done and a few samples to make sure you’re getting the correct values:

-------------------------------------------------------------------------

Sales Value Statistics

-------------------------------------------------------------------------

Mon Tue Wed Thu Fri Total

Low XXXXXXX XXXXXXX XXXXXXX XXXXXXX XXXXXXX $500.00

Ave $136.10 XXXXXXX XXXXXXX XXXXXXX XXXXXXX XXXXXXX

High XXXXXXX $300.00 XXXXXXX XXXXXXX XXXXXXX XXXXXXX

-------------------------------------------------------------------------

Total $4,083.00 $4,848.00 $5,091.00 $6,381.00 $6,378.00 $26,781.00

There are XX employees who sold less than the average total.

There are XX employees who sold above the average total.

The names of the above average selling employees in alphabetical order are:

XXXXXXXXXX XXXXXXXXXX

XXXXXXXXXX XXXXXXXXXX

All of these values are calculated using LINQ. You must derive them that way; nothing else will count – so no arrays, no iterating over the container counting or adding things up in variables – it must be a LINQ query! You may develop the application as a console application (make sure you trap bad input path/filenames). You can just write the output to the console – you do not need to send it to an output file. Do take the time to line things up and format properly or you will lose points for not doing so. Make sure that your application prompts the user for the input filename at the beginning and then pauses before it terminates so that the output can be viewed.

Complete your assignment and place your entire solution in a zip file, which you will upload to Canvas. Turn in a cover sheet, your program source code and screenshots of your program’s execution stapled together in that order in class.