COMP 2614 – .NET Windows Application Development in C#

Assignment Three

Assignment Date: Week 3
Due Date: Week 5

Assignment Objective

The purpose of this assignment is to read and parse text file data, store this data in a data structure and print it to the console. You will also design a solution for a typical business problem.

Note that this is the last console-based assignment—all remaining assignments will be Windows Forms projects.

Your Task

You're a developer for a company that sells computer hardware. One of the vendors sends their invoice data in a custom formatted text file.

Design a solution that imports this file and stores it in an appropriate data structure. Generate output to the console that emulates the screenshot in this document.

Details

Write a console application that:

1) reads a file containing invoice data
2) stores the data in a collection of data classes
3) displays the data from the collection to the console

The program must do just these three things and then exit—that is all.

Obtain the filename via a command line argument. The program must not prompt for a filename or contain a hard coded filename.

The program must be able to handle amounts up to 999, 999.99.

The program must also be able to handle 2 taxes, GST and PST. Since we are a reseller, we have a PST license and are exempt from paying PST. (We will always pay GST) Occasionally, we will purchase items for internal use and pay the PST to the vendor. This will be indicated by a flag in each detail line.

The tax rates are GST 5% and PST 7%.

The format of the file is described below. To keep the assignment from getting too large, your program can assume that the file is always formatted correctly.

Data File Details

Name your data file: **invoicedata.txt**.

Although a data file is included as a sample, you will be creating your own data file. Use Notepad or another text editor to create the file. (create at least 6 invoices)

Each line in the file represents one invoice. There are two delimiter characters, the pipe (|) character delimits the header and line items. The colon (:) character delimits the elements in the header and each line item.

High level structure: (first element represents the header followed by one or more line items (Max 20)

```
Header|Line Item 1|Line Item 2|…
```

Header Elements:

```
InvoiceNumber:InvoiceDate:Terms
3221409:2016/01/07:215
```

InvoiceNumber: AlphaNumeric 8 character max

InvoiceDate: YYYY/MM/DD

Terms: three digits, first digit is discount percentage (maximum 9)
second and third digit is discount period (minimum 10 days)

110 means 1% discount 10 day period

Line Item Elements:

```
Quantity:Sku:Description:Price:Taxable
10:WD2002:2TB Hard Drive:121.66:N
```

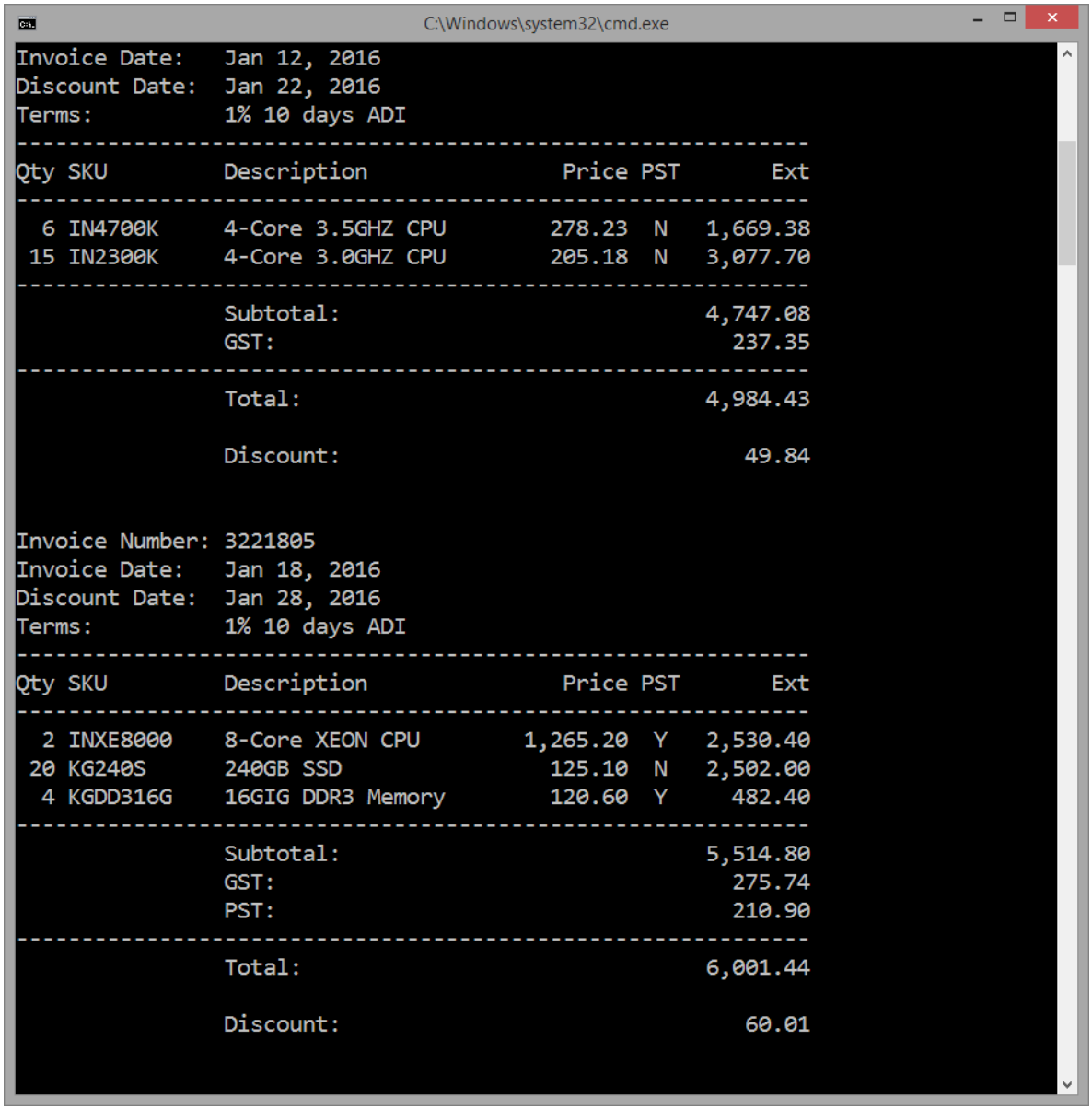
Quantity: 999 maximum value

Sku: 8 characters maximum

Description: 20 characters maximum

Price: 2decimal places

Taxable: Y or N to indicate that PST is payable on this line item



Sample Output

Data Structure Details

Design a set of classes to store the parsed data. Determine the classes required and give them appropriate names. Define the fields for each class and use the most appropriate data type for each field.

You should have classes for Invoice and Invoice Detail Line. The Invoice should hold the header data and a collection for the detail lines. Include any additional classes that you determine necessary to complete the project.

Console Output Details

Emulate the screenshot above. Note content, formatting and alignment.

The terms section is created from the terms data. (ADI means “After Date of Invoice”)

There should be no output until all data has been read, parsed and stored in the objects.

Read the data from the objects to generate the console output.

For all your code, be sure to:

- Comment your code intelligently (no need to comment the obvious)
- Use **XML comments** for each class summary and for each of its *public members*

Please Hand In:

Zip your entire project (**including your datafile**) and upload the zipped file to the Assign03 Dropbox on D2L