Billing1 – Design

You will build a tool that can read text files with information on a business' accounts receivable, support updates and save them back to disk, and produce a few reports in a plain-text format.

The files can be in a variety of formats, and your tool must support at least two:

- A comma-separated-value format: see files **customers.csv** and **invoices.csv**, found in the **data** folder of the starter project
- A flat format with fixed-width records: see **customers.flat** and **invoices.flat**

In both formats, each customer has first and last name, where the string "*first last*" will be unique, and payment terms, which can be cash up front, 30-day credit, 45-day credit, 60-day credit, or 90-day credit. Each invoice has a unique number, first and last name of the customer, amount, issue date, and payment date. If there is no payment date, it means that the invoice has not yet been paid – what we will call an "outstanding" invoice.

Your tool will be given a pair of filenames, and must read the customer and invoice data into memory from the given format. You can rely on the filename extension to indicate the file format: **.csv** for the CSV format, **.flat** for the flat format. (You will not be asked to process customers from one format and invoices from another.)

Your tool will support the following updates to the billing information:

- Create a new customer
- Create a new invoice
- Pay an existing invoice

The tool will also offer query methods that can provide:

- All invoices, ordered by number
- All invoices, ordered by issue date
- All invoices, grouped by customer and ordered by number
- All overdue invoices, ordered by issue date
- All customers and their total volume of business (the sum of all their invoices, whether paid or not, overdue or not)

We define an “overdue” invoice as one that was paid after it was due, where the due date for customers on credit terms is the issue date plus the agreed number of days, and the due date for cash customers is the issue date. For outstanding invoices, the caller will provide an “as of” date, and an outstanding invoice is overdue if that date is later than its due date. For example an invoice issued to a 30-day-credit customer on October 1 would not be overdue if it were outstanding “as of” October 20, but it would be overdue if it were outstanding “as of” November 9.

Your tool will also offer reporting methods that produce the same information as the query methods, but that format the information to a plain-text file of a pre-defined name, such as **InvoicesByNumber.txt**. The tool should be able to produce reports on demand, and also update reports to disk whenever the underlying data changes (new customers, new invoices, and paid invoices, as described above).

Start out by developing a class design for your tool. As you analyze the problem, try to identify relevant design patterns and to incorporate them into the emerging design.