CS 3450 – Design Patterns

Program 2

Date Due: See Syllabus

The Chicago Sun-Times reports stocks of local interest to Chicagoans. Write a **LocalStocks** object that "monitors" local stock activity as it is reported. You will simulate a live feed by reading snapshots from the file **ticker.dat** (details below).

Several periodic custom reports of the stock data provided by your **LocalStocks** object are required, as follows:

1. A report that displays the average of all local stock prices of each snapshot, along with the time the snapshot was taken.
2. A report that displays all companies that have had a price change of 10% or more. List the ticker symbol, the price and the percentage change.
3. A report that displays all fields for the following companies (listed here by ticker symbol): ALL, BA, BC, GBEL, KFT, MCD, TR, WAG

As each snapshot is "received" by the **LocalStocks** object, each report is appended to a unique file associated with the report. Write your code so that a new custom report or a new type of stock data object (like **LocalStocks**) could be added with minimal code impact. Write each report to its own file.

Note that some languages support observers directly (e.g. Java's Observable/Observer classes.) Since you should be learning how observers work, program your own. It's not much code.

Each line of the file **ticker.dat** contains the following 9 fields:

Company
Ticker Symbol
Current Price
$ Change
% Change
YTD % Chg
52-Week High
52-Week Low
P/E Ratio

Each snapshot is a group of lines preceded by a line of the form:

Last updated Sep 1, 2005 4:58:41 PM ET

The snapshot is ended by a blank line. The number of lines per snapshot (equivalent to the number of companies observed) can change at any time. Note that the company name can contain an arbitrary number of spaces.

Don’t worry about minute details of formatting. My output files look like the following:

Average.dat:

```
Sep 1, 2005 4:58:41 PM , Average price: 31.847586206896555
Sep 2, 2005 12:50:26 PM, Average price: 31.926293103448273
Sep 2, 2005 5:30:42 PM , Average price: 30.794545454545446
Sep 6, 2005 3:09:40 PM , Average price: 31.08958677685951
Sep 6, 2005 4:52:15 PM , Average price: 31.157685950413235
```

Change10.dat:

```
Sep 2, 2005 12:50:26 PM:

Sep 2, 2005 5:30:42 PM :
NLEQ 8.75 -12.5

Sep 6, 2005 3:09:40 PM :
HOST 0.2 -31.03
NLEQ 10.0 14.29
UFMC 5.4 29.19
```

Selections.dat:

```
Last updated Sep 2, 2005 12:50:26 PM ET:
ALLSTATE CORPORATION  ALL 55.33 -0.46 -0.82 6.98 63.22 45.50 11.06
BOEING CO  BA 64.52 -1.47 -2.23 24.63 68.38 48.10 31.76
BRUNSWICK CORP  BC 42.91 -0.09 -0.21 -13.31 49.85 38.96 12.75
KRAFT FOODS INC CL A  KFT 30.98 0.24 0.78 -13.00 36.06 30.11 20.39
MCDONALD'S CORPORATION  MCD 31.89 0.34 1.08 -0.53 34.70 26.89 17.08
TOOTSIE ROLL INDUSTRIES  TR 32.35 0.38 1.19 -6.58 33.86 28.24 25.49
WALGREEN CO  WAG 45.32 -0.43 -0.94 18.11 49.01 35.05 30.28

Last updated Sep 2, 2005 5:30:42 PM ET:
ALLSTATE CORPORATION  ALL 55.53 -0.26 -0.47 7.37 63.22 45.50 11.06
BOEING CO  BA 64.34 -1.65 -2.50 24.28 68.38 48.10 31.76
BRUNSWICK CORP  BC 42.99 -0.01 -0.02 -13.15 49.85 38.96 12.75
GRUBB & ELLIS CO  GBEL 6.80 -0.05 -0.73 46.24 7.30 3.55 8.77
KRAFT FOODS INC CL A  KFT 30.79 0.05 0.16 -13.54 36.06 30.11 20.39
MCDONALD'S CORPORATION  MCD 31.90 0.35 1.11 -0.50 34.70 26.89 17.08
TOOTSIE ROLL INDUSTRIES  TR 32.44 0.47 1.47 -6.32 33.86 28.24 25.49
WALGREEN CO  WAG 45.13 -0.62 -1.36 17.62 49.01 35.05 30.28
```

(Note: Selections.dat shows the observer beginning observation after the first snapshot, observing for two snapshots, and then going away.)

My test program may be different from yours, so your files can vary. In my case, I created an Average observer and then ran a snapshot. Then I added a Change10 observer and ran another snapshot. Then finally I added a Selection snapshot, etc. Then I removed them in LIFO order, running snapshots in between (there are 5 snapshots in Ticker.dat). The above is from a Java version.

Note: People seem to have the most trouble with parsing the records. I have found it to be pretty easy in C++ (just use the >> operator). People using C# tend to find themselves writing a lot of code just to parse the data.