Asgn 1 Demo Specs (& Related Notes)

Pseudocode for what Main (the "driver/tester/controller") does

- Declare objects for TheLog & CountryDataTable
- Call Setup – sending in "Sample" for the fileNameSuffix
  (and whatever else might need passing in)
  (so Setup can pass fileNameSuffix along to RawData's constructor)
- For loop with i going from 1 to 3
  Call UserApp sending in i (as a string? or an int? or. . .)
  (and whatever else might need passing in)
  (so UserApp can pass i to TransData's constructor so it knows the file to open)
- Call FinishUp method for CountryDataTable
  send in a TRUE for the Boolean parameter, printTable
  (so that snapshot will be called)
- Call Setup – sending in "All" for the fileNameSuffix . . .
- For loop with i going from 4 to 4
  Call UserApp sending in i . . .
- Call FinishUp method for CountryDataTable
  send in a FALSE for the Boolean parameter, printTable
  (so that snapshot will NOT be called)
- Call FinishUp in TheLog

NOTE ON WHEN TO CALL CountryDataTable.snapshot

- FInishUp in CountryDataTable calls it
  IF Main sends in a TRUE for the Boolean parameter, printTable

WHAT TO DO FOR THE DEMO

1. RawDataSample.csv, RawDataAll.csv and the 4 TransData?.txt files must be in the correct folder in your project
2. Run the program
3. Print TheLog.txt file in NOTEPAD (or WordPad or…).
   - *Use a FIXED-WIDTH FONT (like Courier New) so record fields line up nicely for Snapshot's & SA transaction's output.*
   - *Use a smaller font, if needed, to avoid wrap-around in TheLog file printout*
   - *NOTE: This in one long file which includes Main running Setup and UserApp multiple times – all captured in a SINGLE TheLog.txt file*
4. Print all of your program code files.

WHAT TO HAND IN (in the order specified below)

1. Cover sheet (fill in the top & sign it)
2. Printout of TheLog.txt file
3. The 2 BST worksheets (done by hand)
4. YOUR program code: *(IN THIS ORDER) (There are at least 6 actual separate files]*
   The main program
   Setup class
   UserApp class
   RawData class
   TransData class
   TheLog class
   CountryDataTable class
   any other code files/classes you used in your program
5. CIRCLE THE FOLLOWING in your program code:
   - The increment in BST SEARCH (so I know you're not doing linear search)
   - Any mention of "BST" in SetupProgram or UserApp (so I can take points off)

##

HOW MUCH COMMENTING IS NEEDED?

- **Self-documenting** code including:
  - descriptive **NAMING** of programs, methods, classes, objects, records, fields, namespaces/packages, variables, constants, etc. *[according to traditional C#/Java/C++ naming conventions]*
  - using the same naming as used in the **SPECS** (so "everyone's on the same page")
  - good **MODULARIZATION** using OOP, short modules (no method > 1 page/screen-ish), sharing of TheLog and CountryDataTable classes and using the modularization described in the specs and in class (so "everyone's on the same page")
  - following the **REQUIREMENT SPECS** closely, so that your "boss's" specs act as a form of external documentation (which does NOT need repeating within your program).
- A **top-comment** on each physical file with: the module name & the code author's name & the overall app name
- A **comment-line-of-*'s** between chunks of code (e.g., methods, constructor, …)
- Comments on **tricky code** or unusual ways of doing things or things which don't follow the specs (since a maintenance programmer would read the specs and ASSUME that the program would OF COURSE follow them)
- You do **NOT need line-by-line** commenting

NOTES:

- Re-read specs for A1 to make sure you're doing everything right (to maximize points)
- Both Setup and UserApp use the sequential stream processing algorithm (on RawData and on TransData, respectively). So, looping through the 2 data files is done by Setup and UserApp and NOT inside RawData class and TransData class.