CS3310 – Kaminski - Spring 2013
Asgn 4 Project Specs

World Data App 1.2 – partial
CodeIndex implemented as an external B+ Tree providing QC & LC functionality

############################## OVERVIEW ######################################

This asgn is NOT a modification of an earlier project, although conceptually it's related. The goal is to develop the CodeIndex, implemented as an external B+ tree, for countryCode access of the MainData file.

However, to facilitate quicker development, only PARTS OF CodeIndex class are implemented for A4. Some simplifications:

- Only QC (Query by Code) and LC (List all by Code) functionality are provided.
- IN (INsert) and DC (Delete by Code) are NOT implemented.
- Since Insert is not implemented, you can't take advantage of using that to CREATE the index. But you NEED a B+ tree to be able to test your QC and LC code. So. . . temporarily (for A4), the B+ tree will be created by hand – which I DID ALREADY.

Other simplifications

- NO MainData file is used – as with A1, the results for QC and LC display only the DRP rather than using the DRP to do random access the actual data record (as in A2).
- There is NO NameIndex, so QN (Query by Name), LN (List all by Name) and DN (Delete by Name) are NOT implemented.
- There is NO UserInterface class – instead TempUserApp handles processing of the TransData.txt file and writing to the Log.txt file.

############################## YOUR JOB FOR A4 #################################

1. <u>TempSetup program</u>
   - This converts AsciiCodeIndex.txt file into BinaryCodeIndex.bin file
   - Uses the Sequential Stream Processing Algorithm – i.e., loop til EOF doing
     {ReadOneNode, WriteOneNode}.
     Do NOT read & store the entire ASCII file into memory. Do NOT create the entire BINARY file in memory before dumping it to the file.
   - This is just a utility program, so you need not use OOP. This program does not involve CodeIndex class.
2. <u>TempUserApp program</u>
   - This is just a utility program, so it handles TransData.txt file and Log.txt file directly rather than using a separate UserInterface class.
   - Uses the sequential stream processing algorithm for TransData file
   - Calls CodeIndex's QueryByCode or ListAllByCode methods, as appropriate (for QC or LC tranCodes)
   - This program has NO IDEA how the codeIndex object is implemented.
3. <u>CodeIndex (OOP) class in a physically separate code file</u>
   - The index is implemented as a B+ tree (an EXTERNAL index)
   - The file used is BinaryCodeIndex.bin – and NOT AsciiCodeIndex.txt.
   - The file is opened ONLY ONCE (in the constructor), and HeaderRec data is immediately read into memory at that point. (Don't forget to close the file in FinishUp class – called by TempUserApp AT THE END).
   - The index (file) is NOT LOADED INTO MEMORY – this is an EXTERNAL index – all processing is done using the data on the FILE
   - QueryByCode and ListAllByCode can NOT access the file directly. They control when ReadOneNode needs to be called, specifying the appropriate RRN.
   - sizeOfHeaderRec and sizeOfNode (needed for byteOffset calculation for the seek) are calculated ONLY ONCE (e.g., in the constructor after reading the HeaderRec) using M – no hard-coding of 7 allowed
   - Only ReadOneNode method can access the CodeIndex file. The method body MUST read in an ENTIRE NODE (field-by-field?) into memory, and NOT just a single countryCode (i.e., keyValue) or its drp or. . .
   - There must NEVER be more than a single node in memory at once – so the class only needs storage for a SINGLE NODE
   - There must be NO HARD-CODING of M's actual value (7) – any program code uses the variable name M (read in from the HeaderRec).
   - CAUTION: If you do a linear search of the file, you get NO CREDIT FOR THIS ASGN!!!! Linear searching a NODE in MEMORY is acceptable.

############################## AsciiCodeIndex.txt FILE ############################

It's a BPlus tree (not a plain B Tree).

As an ASCII text file (created manually with NotePad), it contains:

- <CR><LF>'s after each record
- a space between fields within a record (see file itself).

1st record: HeaderRec contains these fields in this order:
m, rootPtr, nextEmptyRRN, firstLeafPtr, nKV
All subsequent records: BPlus tree Nodes contain:
1st) leafOrNonLeaf (L or N)
2nd) m pairs, where each pair contains: countryCode and drpOrTp
(a leaf node contains a DRP, a nonLeaf node contains a TP)
3rd) nextLeafPtr

Non-full nodes have their good-data pairs left-justified within the node, with the "empty" pairs on the right end. Empty pairs contain ^^^ for countryCode (since "^^^" > "ZZZ" in terms of ASCII code order) and 000 for drpOrTp

############################## BinaryCodeIndex.txt FILE ##########################

It's a BPlus tree (not a plain B Tree).

As a binary file, there are NO <CR><LF>'s after records and NO spaces between fields

1st record: HeaderRec contains these fields as SHORT integers, in this order:
m, rootPtr, nextEmptyRRN, firstLeafPtr, nKV
All subsequent records: BPlus tree Nodes contain:

1st) leafOrNonLeaf (L or N) – a single char
2nd) nextLeafPtr – a SHORT integer
3rd) array of m countryCode's – all strings or char arrays
- but use built-in String comparison methods rather than manual char-by-char comparisons
4th) array of m drpOrTp's- all SHORT integers
(NOTE: a leaf node contains a DRP, a nonLeaf node contains a TP)

Non-full nodes have their good-data pairs left-justified within the node, with the "empty" pairs on the right end. Empty pairs contain ^^^ for countryCode (since "^^^" > "ZZZ" in terms of ASCII code order) and 0 for drpOrTp

[NOTE: C#'s String.Compare doesn't use the strict ASCII code order, but String.CompareOrdinal does].

############################ Log file ##

The . . . response to LC is OF COURSE filled in with actual data!!!

```
*** TempSetup program started
*** TempSetup program completed
*** TempUserApp program started

QC FRA
>> DRP: 003 - 5 nodes read in - 13 key-comparisons done
QC WMU
>> NO MATCH - 5 nodes read in - 6 key-comparisons done
LC
ABW 211
AFG  60
AGO 146
. . .
ZWE  24
+++++ END OF DATA +++++ (239 countries)

*** TempUserApp program completed (3 transactions)
```