CS3310 – Kaminski - Spring 2014
Asgn 4 Project Specs

World Data App 2.5
external CodeIndex (as B Tree)
CountryData (as Direct Address)

########################   OVERVIEW OF ASGN 4 ##########################

This asgn is part of a larger project (like A3). However, **YOUR TASK** for A4 is to focus on:

- the UserApp
- and in particular, the SelectByCode functionality, which uses code index, implemented as an external BTree in order to access the main country data file.
- And since the focus is on the index mainly, your program doesn't actually care what random access structure or primary key is used for the main country data file. It only cares that the index's DRPs point to the correct data record. [For simplicity for A4, it's a direct address file on id, it uses only a few "fake" (easy to create) fields, and it's a text file rather than a binary file].

However, you still need (each in its own physical file):

1. The main program as the overall controller/tester to deal with different test data sets, specifying different fileNameSuffixes
2. The UserApp (procedural) class where userAppMain is the controller
3. TheLog class – handles EVERYTHING to do with `TheLog.txt`
4. TransData class – handles EVERYTHING to do with `A4TransData?.txt`
5. CountryIndex class – handles EVERYTHING to do with `CodeIndex?.bin`
6. CountryData class – handles EVERYTHING to do with `CountryData?.txt`

##################   REDUCING THE SCOPE OF ASGN 4 ####################

Since you need to show your progress to your development group in a week and a half (ish), you'll reduce the scope of the project for now:

- Setup's not done yet, so we need someone to create the necessary files.
  [ALREADY DONE!]
- No other transaction functionality besides SelectByCode works yet (you won't even do dummy stubs – there just won't be any other transactions types in the TransData files, including:
  - Insert
  - Delete by code/id/name/…
  - Select by id/name/…
  - SelectAllByCode even though we have a CodeIndex in the form of a BTree which DOES support key-sequential access (i.e., the main data file is "logically sorted" on code, though not physically)
  - SelectAllBy??? (id/name/…)

###########################   THE DATA FILES ###########################

To allow you to proceed with testing YOUR portion of the project (i.e., A4) before the real Setup module is finished, temporary test files needed to be created. Notepad and a conversion utility were used to create the CodeIndex or CountryData files.

CodeIndex?.bin

- CodeIndex?.**txt** was created with NotePad [*thanks DrK]* – so it's an ASCII text file (with spaces as field-separators and <CR><LF>'s), making it easier to create, read (as a human) and check for errors. But the file is NOT in the proper format as to what the Setup module will actually (someday) create. So. . .
- Rather than using real country codes, simple 3-letter words were used since it was easier to make up test data that way. It doesn't affect your program!
- A conversion program *[thanks Austin B]* then created the equivalent file using a format which Setup will (someday) create:
  1. CodeIndex?.bin, a BINARY file (without field/record separators)
  2. the array of triples (TP/KV/DRP) in each record, which you see in the original **text** file, are rearranged into **3 parallel arrays instead, i.e.,**
     all M TPs, then all M-1 KVs, then all M-1 DRPs
  3. numeric fields (M, RootPtr, N, TPs, DRPs) are short's using littleEndian
  4. character fields (KVs, i.e., CountryCode) are fixed-length character arrays (3 bytes) rather than strings. Two versions are provided (choose one):
     - 8-bit ASCII character codes
     - 16-bit Unicode character codes

CountryData?.txt

- File was created with NotePad [*thanks DrK]* - so it's an ASCII text file (with spaces as field-separators and <CR><LF>'s) making it easier to . . . However, Setup will (someday) create a binary file like A2/A3 did, so someday the program would need changing.
- Setup will (someday) make this a hash file on name. But since the focus here is on the index, the file is actually a direct address file on id – since it doesn't matter to A4 as long as the index's DRPs point to the correct records.
- Rather than using real country data, it was easier to make-up data while typing it (which doesn't affect your program).
- Rather than using lots of fields, only 3 fields were used (so someday the program would need changing). Here's the fields used
  id – 2 char's followed by a space
  code – 3 char's followed by a space
  restOfData – 16 char' s (which may include spaces) followed by a <CR><LF>
- Records are fixed-length (as they need to be).

NOTES:

- RRN's start with 1 not 0
- HeaderRec on CodeIndex file (which doesn't count as being at RRN 1) contains: M RootPtr N (all short's with no field-separators nor <CR><LF>)
- NO HeaderRec on CountryData file

A4TransData?.txt

- All records in the form: `SC USA` (followed by a <CR><LF>)
- NOTE: it's now A4TransData?.txt rather than just TransData?.txt

########################## 3 TEST DATA SETS ###########################

There are 3 data sets with the following fileNameSuffixes: 1, 2, 3

Main contains a FOR loop from 1 to 3,
calling userAppMain, sending in i for the fileNameSuffix.
UserAppMain will provide this fileNameSuffix to the 3 constructors which need it in:
CodeIndex class, CountryData class, TransData class

############################## USING M ##############################

The 3 different CodeIndex files facilitate testing the program with various M values for the different M-way Btree indexes.

The 3 CodeIndex files contain M values of 5, 8 and 9 respectively,
but MUST BE NO USE OF THESE ACTUAL VALUES HARDCODED IN YOUR PROGRAM.
Things must be based on some FUNCTION OF M, e.g., for loop from 0 to M-1 or 0 to M-2.

This forces the program to be robust enough to handle various "any" M values
(up to MAX_M = 50),

QUESTIONS: What should M be using a 512-byte block for CodeIndex:

1. **With the implementation specified in this asgn for KV, TP, DRP?**
2. **If countryCode was stored as a string instead of a charArray (using ASCII codes)?**
3. **If countryCode used 16-bit Unicode rather than ASCII codes?**
4. **If long's were used for TP & DRP, and countryCode was a charArray using ASCII codes)?**

############################### TheLog ###############################

```
======================================
PROCESSING A4TransData1

SC IMP
>>> 10 IMP ish        49132
   [# nodes read:  1]
SC CMU
>>> ERROR - code not in index
   [# nodes read:  3]
.  .  .
```
Similarly for A4TransData2 and A4TransData3.

NOTES:

- No status message needed.
- userAppMain declares theLog object,
  which calls the constructor, which opens the file in append mode
- so the Main program should delete the file before starting the for loop which calls userAppMain

########################### CodeIndex CLASS ############################

Contains ALL handling of the external code index BINARY file.

Program MUST use the BINARY versions of the files, NOT the TEXT versions.

Header record data is read into memory ONCE (for a particular CodeIndex file)
in the constructor just after opening file

Do NOT read in the entire FILE into MEMORY. This is an EXTERNAL storage structure,
not an internal data structure.

This class contains storage for a SINGLE node (or perhaps for a "big node" – see below). There is NEVER more than ONE node in memory at once – so re-use the same storage space each time you read in a node into memory. (If you're using a separate node class, don't keep declaring new objects for every node you read – just keep re-using the same storage space).

To simplify searching by reducing the number of loop-stopping conditions to 2 (rather than 3), define the KV array to be of size M rather than M-1 (even though that's NOT what's in the FILE node), and initialize that extra KV as]]] (once at the start, before reading in any nodes).

selectByCode is a public method which calls readANode and searchANode as needed.

The actual B Tree handling methods are private, including (with these names):
[NOTE: TRUTH IN ADVERTISING !!!]
readANode
searchANode
readANode's body could contain a single physical read of the whole block
OR it might contain 3 for loops, each containing a physical read of a field
OR. . .
BUT THE METHOD MUST READ IN AN <u>ENTIRE NODE</u> INTO MEMORY
searchANode must contain a loop rather than if/else's
since it has to be able handle any value for M

NOTES:

- searching must allow for a successful search or an unsuccessful search
- only search as far into the node as needed
- don't search a node for ==, then if no match, go back and search for <

WARNING: If you search the entire file rather than a root-to-hit-target path or root-to-leaf path, you will get 0 points for the asgn ! ! !