CS 3114 Spring 2021 GIS

Student:

GTA:

Submission date: Sub #: (from the submit log, not a file timestamp)	Deduction for: fundamentals	0 /250
	design/engr	0 /80
	documentation	0 /20
	correctness	0 /200
	Total deductions	0 /300
	Total score out of 300	0 /300

Fundamental Requirements Have the student show you the relevant areas in his/her implementation	**Item Deduction**
Required data structures elements[0]:	
PR quadtree – does not use a PR quadtree -100	
Hash table – does not use a hash table -100	
Buffer pool – does not use a buffer pool -50	

Design and Engineering (-80 points maximum) Look at the relevant areas in the submitted implementation[1]	**Item Deduction**
PR Quadtree implementation[2]: -40 maximum	
Hash table implementation[3]: -40 maximum	
Buffer pool implementation[4]: -30 maximum	
Feature name/state and location indices[5]: -40 maximum	
General infrastructure[6]: -30 maximum	
The details of the design evaluation are NOT shown here.	
Showing them would amount to imposing design decisions that you should be making yourself.	
Do not make the mistake of thinking that means that all design decisions are equally good, or that some design decisions will not receive deductions.	
If you've paid attention (design lectures, specifications, Forum discussions), you should not be surprised by much of what I'm looking for here.	
If you haven't, you may be disappointed.	
Comments:	**Total deduction for this section:**

Internal Documentation (-20 points maximum)	
Spot-check comments in a randomly selected source file[7]: Inadequate function/method headers: -10 (statement of purpose, parameter comments, pre/post conditions) Inadequate internal comments in function bodies: -10	
Comments:	**Total deduction for this section:**
Correctness of Program Operation (-200 points maximum)	
Basic test of index-building (-40 points maximum) Command line: `db01.txt DemoScript01.txt Log01.txt` General log issues: Commands not echoed and numbered in log file -10 Log format is badly organized, hard to read -10 PR quadtree integrity shown in display of the location index[8]: Nodes are not organized as a PR quadtree, or not clear from display -30 Doesn't have 13 leaf nodes -10 Doesn't show a record holding two offsets (like 1366 and 3008 in mine at line 49) -10 Hash table integrity shown in display of the feature ID index[9]: Index entries are not organized as a hash table, or not clear from display -30 Doesn't have 25 full slots -10 Did not create a db file containing the imported records[10] -30	
Comments:	**Total deduction for this section:**

Test of simple searching with a small dB file (-20 points maximum)
Command line: `db02.txt DemoScript02.txt Log02.txt`
 	
Basic searches[11]: -3 per search if anything is wrong
 	
`what_is United States Mountain CO`
`what_is Cottonwood Creek CO`
 	
`what_is_at 380145N 1074019W`
`what_is_at 375437N 1074146W`
 	
`what_is_in 380122N 1074015W 15 15`
`what_is_in 380122N 1074015W 30 45`
`what_is_in 380122N 1074015W 15 15`	

Comments:	**Total deduction for this section:**

Test of multiple imports[12] (-20 points maximum) Command line: `db03.txt DemoScript03.txt Log03.txt` Runtime crash during the test -20 Searches: `what_is_in  381257N  0794039W  120  60` -10 `what_is_in  381621N  0794457W  1200  30` -10	
	Total deduction for this section:
Test of the buffer pool with a small dB file[13] (-30 points maximum) Command line: `db04.txt DemoScript04.txt Log04.txt` Runtime crash during the test -10 Buffer pool contents and ordering: deduct as listed if anything is wrong `Command:` `3` -2 `8` -4 `19` -10 `21` -4 `23` -4 `25` -4 `27` -6 `30` -6 `35` -6 `40` -6	
Comments:	**Total deduction for this section:**
Test for search failures[14] (-20 points maximum) Command line: `db05.txt DemoScript05.txt Log05.txt` Runtime crash during the test -20 Report incorrect records (none should match) -5 per search Don't log an informative message but don't log incorrect records either -5 global	
Comments:	**Total deduction for this section:**

More varied test of region search[15] (-30 points maximum)

Command line: `db06.txt DemoScript06.txt Log06.txt`

Runtime crash during test -10

```
Command                                   -5 each if anything is wrong
   2
   3
   4
   5
   6
   7
   8
```

Comments:

Total deduction for this section:

Test with large database file[15] (-50 points maximum)

Command line: `db07.txt DemoScript07.txt Log07.txt`

Runtime crash during the test -10

Any sort of failure in any of the searches

```
what_is  Nester Draw               NM           -5 each
what_is  Screaming Left Hand Turn  NM
what_is  Window Rock               NM
what_is  Buena Vista               NM[16]
;
; Now do some location searches:
what_is_at  363957N  1054049W                   -5 each
what_is_at  351018N  1034328W
what_is_at  362846N  1085222W
what_is_at  334236N  1055604W
;
; And some region searches:                     -6 each
what_is_in  362846N  1085220W  120 120
what_is_in  333859N  1062731W  120 120
what_is_in  345326N  1073457W   60  60
```

Comments:

Total deduction for this section:

Other Adjustments

Other Issues[17]

Note any problems you noticed that weren't covered above and suggest a penalty for them:

Notes:

0 This is simply checking whether the solution actually implements the three mandatory data structures. You are concerned yet with whether they are implemented correctly. Be careful of situations where a very incomplete implementation is supplied, but not actually used. There should be enough of an implementation to convince you that the student has actually made a serious attempt to complete the requirements.

If the deductions for the quadtree or hash table apply, either the student will have substituted some other structure, probably something much simpler, or else the student will not have a working solution.

If the student is penalized here, try to avoid double-jeopardy in the later sections. For example, if the student did not implement a buffer pool, skip the test of the buffer pool when you test the functionality (and do not penalize the student for that test, but do enter a comment for that test indicating it was skipped, and why).

1 The students were told to include a README file that points you to most of the elements listed below. If the student did not do that, email the student and demand they email you such a file within 24 hours. If they don't do that, or you simply cannot find one of the elements, then apply the full deduction for that element.

The students were also given the .class files for my PR quadtree and hash table implementations. If they used those, then their submission must contain those .class files instead of .java files. If so, make a note that the student used the supplied code and do not penalize here for either the PR quadtree or the hash table requirements. However, you can still penalize if things don't look right on the displays in the first test case.

2-6 See my comments on the first page… I'm not showing details here.

7 For method header comments, I would expect to see a brief description of what the function does. For accessors and mutators that would be sufficient. For more complex methods, I would expect to see pre-conditions, and possibly post-conditions.

For internal comments in a method, I would expect to see comments on the more complex code blocks, explaining the purpose of the block, or explaining how it does what it does.

Feel free to give partial credit here.

8 They must display the PR quadtree in such a way that you can verify it looks like a proper PR quadtree. It must be possible to tell which nodes are in each level, and how what the parent/child relationships are. It's OK if they don't use the same format I used in my logs, but it MUST be clear to you that the structure is really a PR quadtree. I would expect their PR quadtree to have the same number of nodes and the same number of levels as mine, but they may display the quadrants in a different order. Record offsets might be slightly different than mine; if so that's OK.

9 The hash table display should at minimum show the filled slots, including both the file offset for the record and some indication of what the record is. The simplest approach is to just show the entire line from the file, but it's acceptable to show less than that as long as it's clear what the record is.

It is entirely possible that they will show a many records in different slots than I do; that's OK since they may have formed the key differently than I did.

10 It's possible that the student's db file will be organized differently than mine. That's OK, as long as it contains exactly 25 records. Note that it may contain a header line as well, in which case there would be 26 lines.

11 For each search, they should report exactly the same record(s) that I do. For the searches, the specification says exactly what record fields must be displayed.

I don't think there is much reason for partial credit on an individual search unless you want to deduct points for not showing all of the specified record fields.

12 Each of the region searches is designed so that the results should include records from both of the files that were imported, so if any records are missing, that probably indicates the student did not handle the second import correctly. Give no partial credit for either search.

13 The student must display the records in MRU to LRU order. The searches were chosen so that each should yield a single match. So, the contents of the student's pool should be the same as mine, and they should be in the same order. I would not give any partial credit for any buffer pool display.

If the search results are incorrect, it's probably not going to be possible to verify that the student implemented the buffer pool correctly; do not give credit for anything here unless you are convinced by the output that the implementation of the buffer pool is correct.

14 This is all about how, or if, they handle searches that do not find any matching records. There is a high probability that they will have runtime errors on this test, if they have them anywhere. For each search, there are three possible outcomes:

- They log a message indicating nothing was found, and do not have a runtime error.
- They log incorrect records that do not match the search criteria, and do not have a runtime error.
- They have a runtime error.

If they have any runtime errors, deduct points as specified and then evaluate their output, if there is any.

If they report records that really do not match the search criteria, they should lose 5 points for each such search. Finally, if they don't report any incorrect records, and don't have a runtime error, but don't log an informative message, they should lose 5 points (not per test).

15 With region searches, the order in which they report the matching records does not matter. If a search results in more than 5 matches, just verify that they report the correct number of records, unless they're getting wrong results for the searches with a small number of matches.

There are 8 separate searches here, and only 50 points to go around. Assign 5 points each to the `what_is` and `what_is_at` searches, and don't give any partial credit for a search (if you see a case where you think that's too harsh, ask me about it). Assign 6 points to each of the `what_is_in` searches, and evaluate them as in the previous test.

16 There is a record for Buena Vista in the imported file that does not list latitude/longitude. I show that record in my output for this search because I do import such records and index them by name. It's OK if the student chose to not import such records; so don't penalize them for omitting that record from their output.

17 This is the place to make note of anything you think I should look at. Basically, if you see something you don't think fit the earlier instructions, a kind of error I didn't consider, then describe it briefly and suggest a penalty.

One example would be a solution that doesn't fit the specification for the command-line interface. For instance, a student may take the parameters in a different order than specified; I'd probably penalize that 15% for your trouble in figuring it out. Or, a student may hardcode names for the input/output files; in that case, my inclination is to email the student and require they resubmit a fixed version of the assignment; I'd probably penalize that by applying the late penalty according to the date of the resubmission.