COMP5511 Data Structures & Algorithms:

Bipin C. Desai Asg. 3 Due date: See CrsMgr

Question 1 (10 points)

As described in the notes, insertion sort goes sequentially through the array when making comparisons to find the proper place for the element that is currently being processed. Suppose that the sequential search is replaced by a binary search.

a) Will the change decrease the overall cost of insertion sort? Explain.

b) With respect to overall cost, will the modified version of insertion source be in a faster θ-category than the version in the notes? Explain.

Question 2 (10 points)

Consider applying radix sort to a list of 10,000 integers in the range from 1,000,000 to 9,999,999.

a) Not counting the space needed to store the integers themselves, how many bytes of memory would be required? Assume pointers occupy 4 bytes.

b) How many times would radix sort examine each value in the list?

Question 3 (80 points)

Canadian Geographic Name database contains names of significant geographical places and information pertaining to geographic features and provide various kinds of access to the information. A geographic feature may possess many attributes (see below). In particular, a geographic feature has a specific location. There are a number of ways to specify location. For this project, we will use latitude and longitude, which will allow us to deal with geographic features at any location on earth. A reasonably detailed tutorial on latitude and longitude can be found in the Wikipedia at https://en.wikipedia.org/wiki/Latitude and https://en.wikipedia.org/wiki/Longitude.

We will employ public data obtained from the Geographic Names Information System:

https://open.canada.ca/data/en/dataset/e27c6eba-3c5d-4051-9db2-082dc6411c2c.

There is a guide which describes the data: the data is in various format but for this project we are limiting it to a subset in text format(csv): each field in a csv file is separated by comma(,). You can open it in a spreadsheet and delete columns for the details not required and create a record file with the required fields. You are to decide on the format of the record file - CSV may be an option.[1]

You will implement a system that various indexes and provides search features for this file: the program you write would be modified in the next assignment to add other indexing structures. The system should build and maintain several in-memory index data structures to support these operations:

[1]See https://users.encs.concordia.ca/$\tilde{\text{b}}$cdesai/COMP5511/cgn_qc_csv_eng.csv.zip for data on Quebec

Retrieving records matching given geographic coordinates
Retrieving records matching a given unique record ID
Retrieving records that fall within a given geographic region
Displaying the in-memory indices in a human-readable manner

You will implement a single program to perform all system functions. The distribution of the work among team members should be done in an equitable manner.

Program Invocation:

The program will take the names of three files from the command line, as follows:

```
Yoursystem <record file> <command script file name> <log file name>
```

If the record file or the script file are not found the program should log an error message and exit. If a log file name is not specified, the program should echo an error message and exit.

Data and File Structures:

There is no guarantee that the record file will not contain two or more distinct records that have the same geographic coordinates. So, we will not treat geographic coordinates as a primary key.

The records will be indexed using the CGNDB ID, using a perfectly height-balanced BST. The index entries will store and a pointer to an array containing the matching record. Since all of the possible records that must be indexed are known when the index is built (i.e., you will never add or remove records in this assignment), it is possible to build a BST that has the theoretically minimal height, and that is what you will do.

In addition your program will create an in-memory inverted index on the Geographical Name; The index would contain only non-common names or part thereof. Words such as: á aux, des, du, of, la, sur etc should be not in the index.

When building the index structures, your program will make one complete pass through the record file. Aside from where specific data structures are required(BST, inverted index), you are free to create your own structures and ADTs.

Each index object should have the ability to write a nicely-formatted display of itself to an output stream.

Query to the system would be of the type:

What is at latitude, longitude given as:
What-is-at: 453431N 733256W (45° 34' 31" North::73° 32' 56" West)
Describe: Notre-Dame-du-Bel-Amour

Find(all): Paroisse
Find the details about: EMHBD
Find (all): Lav Vert

The query file would be prepared by the Lab. instructor and your program should produce an output in the log file. The instructor will also specify the exact format of the log file.

Documentation:

Make sure to submit the code with internal as well as external documentations as well as the log file of the output of your program.