Comp 5511/2-FF

Bipin C. Desai

Assignment 4 **Due: See CrsMgr** **Max. Marks: 100 points**

Part 1: 20 points

Onlytwo of the following question would be graded: you are required to submit the answer to all to get any mark for this part. .

Question 1: 10 points

(a) What is the maximum depth of a *2,3-Tree*that has 15 values?

(b) What is the minimum depth of a *2,3-Tree*that has 15 values?

(c) What is the maximum depth of a BST that has 15 values?

(d) What is the minimum depth of a BST that has 15 values?

Question 2: 10 points

Draw the *2,3-Tree*that you would get by starting with an empty tree and inserting the following values, in order:

```
1, 150, 35, 145, 19, 24*
10, 17, 20, 30, 201 *
140, 207, 120, 5*
115, 40, 7*
```

Give the figure for the tree at points indicated by *.

Question 3: 10 points

Draw the B_+-tree of order 4 that you would get by starting with an empty tree and inserting the following values, in order:

```
1, 78, 37, 150, 35, 145, 19, 24*
10, 210, 17, 20, 30, 201 *
140, 207, 120, 5*
115, 51, 40, 7*
```

Assume that the maximum number of keys in the B_+-tree is 4 and the minimum is 2: the leaf nodes have same size as the internal nodes (so they also store a maximum of 4 keys). Give the figure for the tree at points indicated by *.

Question 4: 10 points

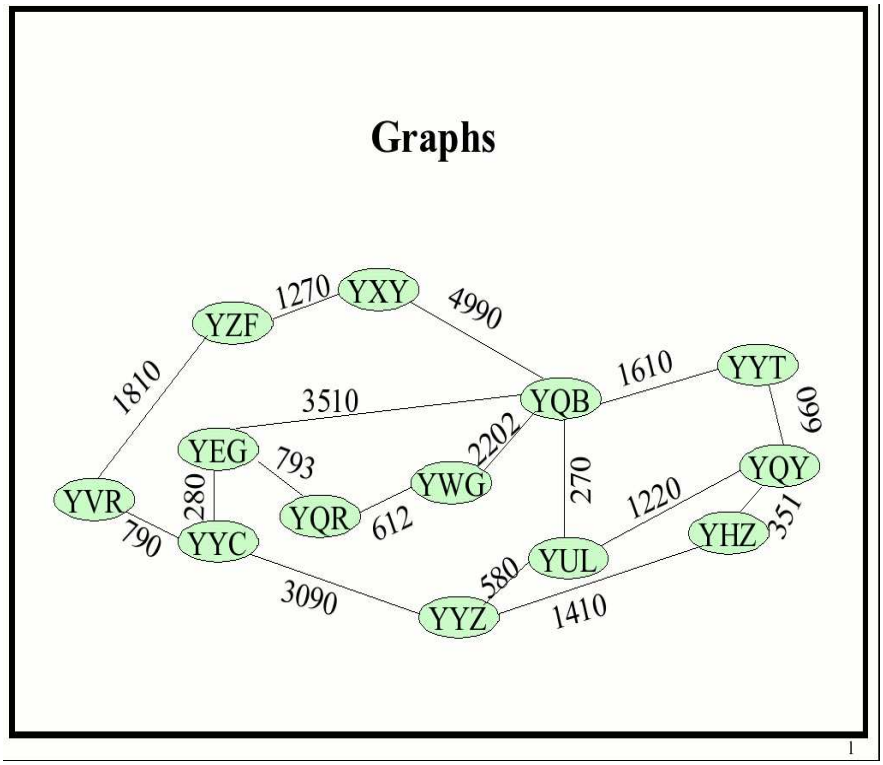
Compute the last-occurrence function and $f(j)$ for the following patterns

(a) supercalifragilisticexpialidocious

(b) abracadabra

Question 5: 10 points

For the following graph, give the BFS and the DFS traversal starting from YUL assuming that the order of the nearest neighbour is clockwise starting from the noon position. Your answer should give the order of airports visited, the "explored" flight paths and the "cross" flight paths.



Part 2: 80 points

Question 6

Complete the programming part of A3. 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:
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.

[1]See https://users.encs.concordia.ca/~bcdesai/COMP5511/cgn_qc_csv_eng.csv.zip for data on Quebec

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 will store CGNDB ID 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 it would have the types of queries given below. You may need too include the exact nature of the query file required and should include in your submission a sample query file and one that could be edited and changed. Your program should produce an output in the log file..

Programming, Documentation:

The code used must be original and must not used any-canned programs, libraries and short cuts. Any IDE used must be FLOSS/FOSS.
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.
The marker should be able to change the input csv file for another; for example use the one for ON instead the one for QC).

The task involves, scanning a document in text format and eliminate all the

Suggestions, Queries, log, etc.

```
The problem needs more than one index of types: BST, B+-tree and
 inverted index There is no need to create links between indices. The
 result of each index could be used to do logical operations to get
 the required results.

The BST/B+-tree could be on the ID


One cannot get away from duplication of pointers to records: however
there should be only a single copy of the complete records.
```

For inverted indices, some of the term may not be eliminated so one needs to decide what is a noise word and what is not.

Some of the following have to be resolved

Anse au Sable has more than one ID so all of them must be in the "document" part of the inverted index,

Anse au Sapin : this may be entered as Anse Sapin or keep the whole term

Latitude and longitude may need another index a B+-tree may be considered for it.

So the problem is an 'open' one and should be resolved in the lab and group discusiions.

Pointers could be the unique identifiers which in turn could be in a B+-tree to point to the full records.

Also, if all the indices are built in memory, there may not be sufficient space to deal with the entire file which has over 125K records! In this case, one may work with a subset. However the program should be able to work any subset of any province from the data set.

==========

Here are some typical queries:

- Find the details of : Lac Verte , Athabasca, Amherst

- Find all details for lakes with sub-string(ignore case): 'Verte', 'Leamy'

- What is at latitude and longitude: x, y

- What is at between latitudes and longitudes: x, y, x1, y1

- Find the details of : Parc nationals/regional

- Find all : places with the substring: 'Abrupt' 'Anse', 'Bouleaux'

- What is at latitude and longitude: Anse aux Indiens, Parc national de la Pointe-Taillon

Find the details of : <GeoID>
Output:
Goegraphic Name:
Latitude:
Longitude:

```
Find all : <GeoName>
Output
Geographic ID:
Geographic Name:
Latitude:
Longitude:

<If multiple values>
Geographic ID:
Geographic Name:
Latitude:
Longitude:

What is at latitude and longitude: <lat>,<long>
Output:
Geographic ID:
Geographic Name:


For all the queries, if data does not exist then print a message such as:
Data does not exist for <query term>


========================================
```