ECE250: Lab Project 3

Due Date: March 13th, 2020, 11:00 pm

1. Project Description

The goal of this project is to write a C++ implementation for a **Quadtree** that stores informations related to cities based on their latitudes and longitudes. Each node of the tree holds information such as population, cost of living, and average net salary.

A quadtree is a tree data structure in which each internal node has exactly four children. A node of a **point quadtree** [1] is similar to a node of a binary tree, with the major difference being that it has four pointers (one for each quadrant) instead of two ("left" and "right") in the binary tree. Also, a key is usually decomposed into two parts, referring to x and y coordinates. Therefore, a node contains the following information:

- four pointers: quad['NW'], quad['NE'], quad['SW'], and quad['SE']
- point which in turn contains:
    - key: usually expressed as x, y coordinates
    - value: such as the name of a city.

When inserting a node (x1,y1) under node (x,y) the new node is placed in one of the four directions following these rules:

- If x1 >= x and y1 > y, (x1,y1) is placed in the NE child node.
- If x1 < x and y1 >= y, (x1,y1) is placed in the NW child node.
- If x1 <= x and y1 < y, (x1,y1) is placed in the SW child node.
- If x1 > x and y1 <= y, (x1,y1) is placed in the SE child node.

We ask that you write a **C++ class Quadtree using a recursive approach**. This Quadtree will store cities and data about the city. When inserting each city into the tree, you will use the longitude and latitude of the city position as the coordinates x and y, respectively.

Your C++ implementation allows a user to issue different queries on the cities based on their geographical locations. For examples: What is the total population for cities located in North East of Toronto? What is the city with the lowest cost of living in South East of Ottawa?

2. Program Design

Write a short description of your design. You will submit this document along with your C++ solution files for marking. This document must include your design decisions. Please refer to the course website for "Programming Guidelines" and the expected content for your design document.

3. Project Requirements

Write a test program (named **qttest.cpp**) which will read commands from standard input and write the output to standard output. The program will respond to the commands described in this section.

In the table below "*city_info*" represents a set of attributes separated by a semicolon:

- Name
- Longitude (*x*)
- Latitude (*y*)
- Population (*p*)
- Cost of Living (*r*)
- Average Net Salary (*s*)

For example, "Toronto;43.66;-79.42;5213000;2157;3396".

In addition, "*d*" represents a direction: NW, NE, SW or SE. "*attr*" can be "*p*", "*r*", or "*s*" which represents corresponding attribute described above.

Command	**Parameters**	**Description**	**Output**
i	*city_info*	Inserts a node for a city with a set of attributes	**success:** if the insertion command was successful **failure:** if the insertion command was unable to complete since the city was already there
s	*x;y*	Searches for node (x,y)	**found** city_name **not found**
q_max	*x;y;d;attr*	Finds the maximum value for *attr* under node (*x,y)* in direction *d*	**max** max_value **failure:** if the city is not found or the direction is empty
q_min	*x;y;d;attr*	Finds the minimum value for *attr* under node (*x,y)* in direction *d*	**min** min_value **failure:** if the city is not found or the direction is empty
q_total	*x;y;d;attr*	Finds the total value for *attr* under node *x,y* in direction *d*	**total** total_value **failure:** if the city is not found or the direction is empty
print		Prints the city names in the tree using an inorder traversal	city1_name city2_name .. **Note:** This is just for checking whether insertions are correct, the order should be NE, NW, ROOT, SW, SE
clear		Deletes all the nodes from the tree	**success**
size		Returns the number of cities in the tree	**tree size** count

The time complexity for insert should be O(lg(n)) if balanced, and O(n) for clear.

- **Test Files**

The course website contains example input files for the corresponding output files. The files are named *test01.in*, *test02.in* and so on with the output files named *test01.out, test02.out* and so on. The data in the test cases are borrowed from these online sources [2][3].

4. How to Submit Your Program

Once you have completed your solution and tested it comprehensively in your computer or on the lab computers, you have to transfer your files to the *eceUbuntu server* and test there since we perform the automated testing using this environment. Once you finish testing in the *eceUbuntu server*, you will create a compressed file (tar.gz) that should contain:

- A typed document (maximum three pages) describing your design. A document beyond 3 pages will not be marked. Submit this document in PDF format. The name of this file should be:

  xxxxxxxx _design_ pn.pdf in which xxxxxxxx is your UW user id (e.g., jsmith) and n is the project number that is 3 (three) for this submission

- A test program (**qttest.cpp)** that reads the commands and writes the output
- Required header files and classes (ending in *.h .hpp .cpp*)
- A make file (named Makefile), with instructions on how to compile your solution and create an executable file named **qtdriver**

The name of your compressed file should be ***xxxxxxxx***_p***n***.tar.gz, where ***xxxxxxxx*** is your UW user id (e.g., jsmith) and ***n*** is the project number that is 3 (three) for this submission.

References

[1] https://en.wikipedia.org/wiki/Quadtree#Node_structure_for_a_point_quadtree

[2] https://simplemaps.com/data/ca-cities

[3] https://www.numbeo.com/cost-of-living/