# ECE 250 - Project 3
# Cities Information as a Quadtree
# Design Document

# Ayushi Tiwari, UW UserID: attiwari

# Mar 13th, 2020

## 1. Overview of Classes

**Class 1:** City

**Description:**
Stores all the city information and node pointers of city types.

**Member Variables:**
1. String Name (city Name)
2. x stored as a double (Longitude)
3. y stored as a double (Latitude)
4. Population
5. Cost of living
6. Average Net Salary
7. City type pointer northWest
8. City type pointer northEast
9. City type pointer southWest
10. City type pointer southEast

**Member Functions (operations):**
1. insert: inserts the node using the method of recursion
2. search: similar to insert, uses recursion to search the node in the tree if it exists
3. q_max: using inorder traversal finds the max value of attributes in a particular direction
4. q_min: using inorder traversal finds the min value of attributes in a particular direction
5. q_total: using inorder traversal finds the total value of attributes in a particular direction

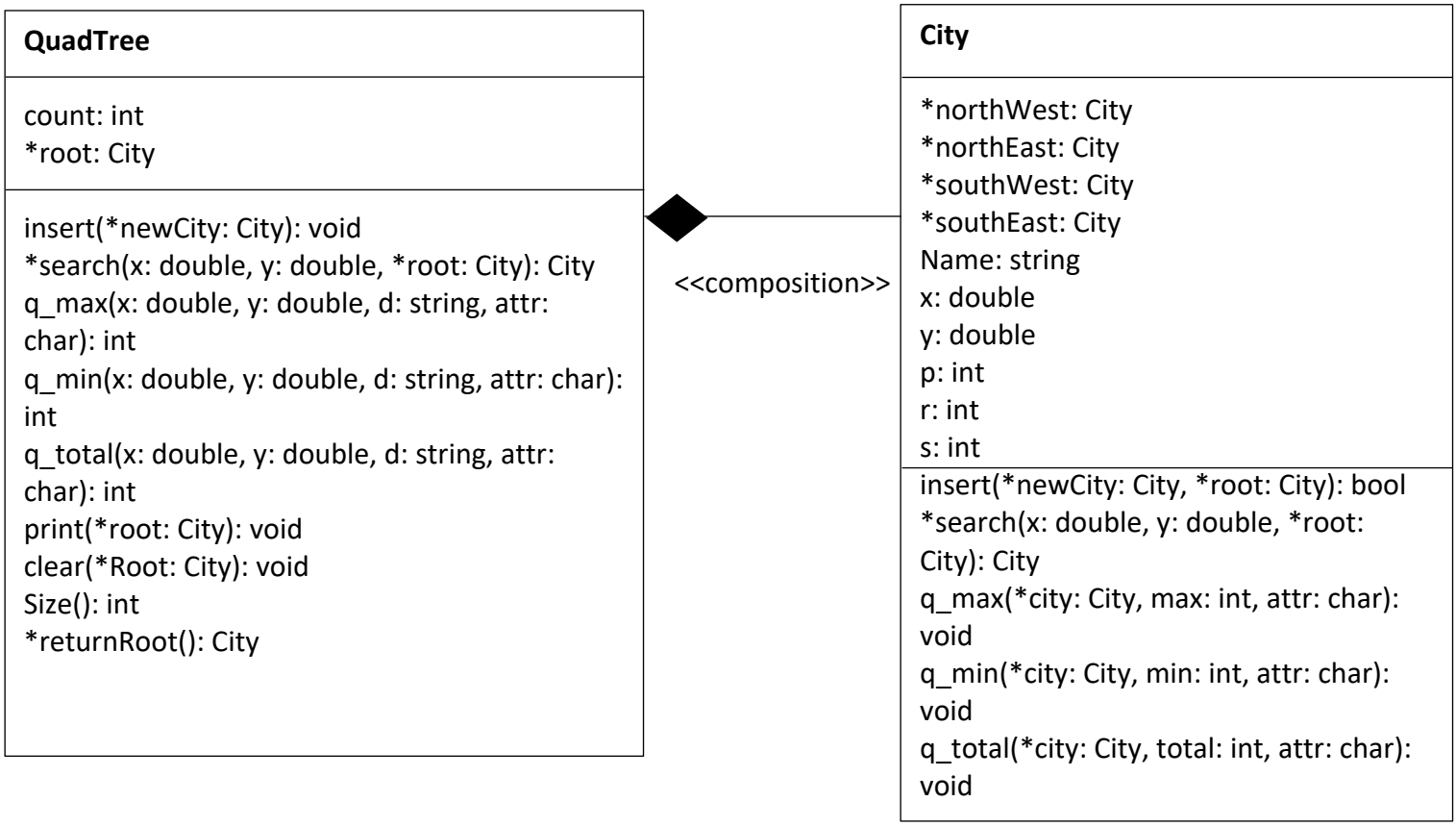
**Class 2:** QuadTree

**Description:**
It is the main class implementing tree data structure that has all the functions and

**Member Variables:**
1. counter
2. City type pointer root

**Member Functions (operations):**
1. insert: adds a new City node if the tree is empty or inserts the node by calling the insert function of the City using the method of recursion
2. search: similar to insert, uses recursion to search the node in the tree if it exists
3. q_max: finds the max value of the attribute in the specified direction
4. q_min: finds the min value of the attribute in the specified direction
5. q_total: finds the total value of the attribute in the specified direction
6. print: prints the entire tree using Inorder traversal
7. clear: clears the entire tree
8. size: returns the size of the tree
9. returnRoot: returns the address stored in the root of the tree

**UML Class Diagram -**




**2. Constructors/Destructors**

**Class City (Constructor):**
The constructor assigns all the values of member values to be 0 in the beginning.

**Class City (Destructor):**
Not required.

**Class QuadTree (Destructor):**
Not required

**Class QuadTree (Destructor):**
Since there is a need to release all dynamically allocated memory, for this we call the City's clear function inside the Destructor to free memory assigned by us and make all pointers null.

## 1. Test Cases

Test 1: Check if a tree was created properly by inserting a few cities and print the tree.

Test 2: Now check if the tree's search function works properly. If it is able to find cities that exist as well as those that don't exist. Check for the size of the tree as well if it is returned properly.

Test 3: Lastly, checking for the maximum, minimum and total values if they are returned correctly. Now deleting the whole tree and printing it to check if it was deleted properly.

Test File Example 1:

```
i Toronto;-
79.42;43.66;5213000;2157;3396
i Montreal;-
73.58;45.5;3678000;1219;3215
i Vancouver;-
123.13;49.25;2313328;2028;3734
i London;-
81.25;42.98;346765;1060;2972
i Hamilton;-
79.86;43.26;721053;1077;3287
i Oshawa;-
78.87;43.9;450963;1190;3112
print
```

Test File Example 2:

```
i Toronto;-
79.42;43.66;5213000;2157;3396
i Montreal;-
73.58;45.5;3678000;1219;3215
i Vancouver;-
123.13;49.25;2313328;2028;3734
i London;-
81.25;42.98;346765;1060;2972
i Hamilton;-
79.86;43.26;721053;1077;3287
i Oshawa;-
78.87;43.9;450963;1190;3112
s 73.58;45.5
s 74.89;89.42
size
```

Test File Example 3:

```
i Toronto;-
79.42;43.66;5213000;2157;3396
i Montreal;-
73.58;45.5;3678000;1219;3215
i Vancouver;-
123.13;49.25;2313328;2028;3734
i London;-
81.25;42.98;346765;1060;2972
i Hamilton;-
79.86;43.26;721053;1077;3287
i Oshawa;-
78.87;43.9;450963;1190;3112
q_max 123.13;49.25;NE;p
q_total 81.25;42.98;SE;p
q_min 123.13;49.25;SW;p
```

## 2. Performance

Insert function and Search FUnction has a time complexity of O(logN) for average case.
Q_max, q_min and q_total functions uses Inorder Traversal that follows a time complexity of O(N) for average case.
Clear will have a time complexity of O(N) as it's in a loop.
Size returns the size of the function so its time complexity is O(1).
Best time complexity of all functions would be O(1) when the tree is empty and there is no node in it so it doesn't have to call the functions.
Worst time complexity of all functions is O(N) when it's the end node and the height of the tree is N.

## 3. Sources


https://www.geeksforgeeks.org/quad-tree/
http://www.stargroup.uwaterloo.ca/~ece250/materials/notes/Lecture10-Trees.pdf
https://stackoverflow.com/questions/37642688/how-can-worst-case-complexity-of-quad-tree-on