```cpp
#include "Header.h"
#include "cities.h"

void Generate(City*& root, int numOfCities) {
    string name;
    int x;
    int y;
    int distance;


    srand(time(0));

    for (int i = 1; i <= numOfCities + 1; i++) {
        name = "City #";
        name += to_string(i);
        x = rand() & 99;
        y = rand() & 99;
        distance = Distance(x, y);
        distance = int(distance);
        Insert(root, distance, name, x, y);
    }
}

void Insert(City * &root, int distance, string name, int x, int y) {
    City* a;
    City* b;
    City* c;

    a = NULL;
    b = NULL;
    c = NULL;

    if (root == NULL) {
        root = new City();
        root->name = "My City";
        root->x = 0;
        root->y = 0;
        root->distance = 0;
    }
    else {
        a = new City;
        a->left = NULL;
        a->right = NULL;
        a->x = x;
        a->y = y;
        a->name = name;
        a->distance = Distance(a->x, a->y);

        c = root;
```

```
50          while (c != NULL) {
51              b = c;
52              if (c->distance == distance) {
53                  delete(a);
54                  return;
55              }
56              else if (c->distance > distance) {
57                  c = c->left;
58              }
59              else {
60                  c = c->right;
61              }
62          }
63
64          if (b->distance > distance) {
65              b->left = a;
66          }
67          else {
68              b->right = a;
69          }
70      }
71  }
72
73  double Distance(int x, int y) {
74      double distance;
75
76      x = double(x);
77      y = double(y);
78      distance = pow(x, 2) + pow(y, 2);
79      distance = sqrt(distance);
80
81      return distance;
82  }
83
84  void inorder_tree_walk(City* root, int &i, int k) {
85      City* current;
86
87      current = root;
88
89      if (current != NULL) {
90          inorder_tree_walk(current->left, i, k);
91          Visit(current, i, k);
92          inorder_tree_walk(current->right, i, k);
93      }
94      return;
95  }
96
97  void Visit(City* root, int &k1, int k2) {
98
```

```
 99      if (root == NULL) {
100          return;
101      }
102      else {
103          if (k1 > 0 && k1 <= k2) {
104              cout << "City: " << root->name;
105              cout << "\nDistance: " << root->distance;
106              cout << endl << endl;;
107          }
108          k1++;
109      }
110  }
111
112  void Free(City*& root) {
113      if (root == NULL) {
114          return;
115      }
116
117      Free(root->left);
118      Free(root->right);
119  }
120
```