PROJECT 4

AVL & BST

CSCI 220
DATA STRUCTURE 1

MAI PHAM

DEVELOPMENT ENVIRONMENT
MacOS – Xcode
Window 10 – MSVS 2017

TABLE OF CONTENTS

Mai Pham
CSCI 220 - Project 3
Prof. T. Vo

PROJECT NOTE

This project is to create an AVL tree to implement a population database storing county/state code, population, and county/state name. In general, I feel like the codes from the book are somewhat really confusing, mostly with all these typedef and template and inheritance. Therefore, I did struggle with that part of the project. I also have some problems with Xcode. I just found out that, for some reason, when I work on a new project, the input location is from an old project folder instead of the project folder I'm currently work on. Also, if I setup/change certain setting, the setting would reset to default when I work on new project. As for the extra credit, I'm doing the BST implementation extra credit instead because I think it is a little bit easier compare to drawing a tree. Finally, my project is completed and successfully run the main project and the extra credit.

BST vs AVL

An AVL tree is a binary search tree that satisfies the height-balance property. This mean the height on each side of the tree is no differ by at most 1. So, in an AVL tree, when we do insert or erase, the tree would check to make sure it is still balance. If not, it would restructure so each side of the tree the height is no more than 1. With the restructure, the number of nodes the tree need to goes through is much less compare to a BST.

Recording to the following output, it takes the BST 9 milliseconds to search for the node while the AVL only take 6 milliseconds. As for inserting, the BST takes 8 and 10 milliseconds to insert a node while the AVL only take 6 milliseconds for each insertion. For the deletion, the BST takes 9 and 11 milliseconds to search and delete a node while the AVL only take 6 and 7 milliseconds. Therefore, based on these output, we can tell that an AVL tree take shorter amount of time to search, insert, or delete a node as it takes less number of node to goes through compare to a BST tree.

INPUT FILE

```
06003,0,"Alpine, CA"
06049,0,"Modoc, CA"
06063,0,"Plumas, CA"
06091,0,"Sierra, CA"
06105,0,"Trinity, CA"
06005,1,"Amador, CA"
06009,1,"Calaveras, CA"
06035,1,"Lassen, CA"
06043,1,"Mariposa, CA"
06109,3,"Tuolumne, CA"
06093,4,"Siskiyou, CA"
06027,8,"Inyo, CA"
06015,19,"Del Norte, CA"
06051,19,"Mono, CA"
06103,25,"Tehama, CA"
06057,26,"Nevada, CA"
06089,29,"Shasta, CA"
06033,32,"Lake, CA"
```

```
06021,36,"Glenn, CA"
06023,42,"Humboldt, CA"
06011,60,"Colusa, CA"
06017,90,"El Dorado, CA"
06069,94,"San Benito, CA"
06045,102,"Mendocino, CA"
06115,105,"Yuba, CA"
06007,150,"Butte, CA"
06061,162,"Placer, CA"
06079,171,"San Luis Obispo, CA"
06101,172,"Sutter, CA"
06031,205,"Kings, CA"
06039,221,"Madera, CA"
06055,225,"Napa, CA"
06025,295,"Imperial, CA"
06047,341,"Merced, CA"
06087,373,"Santa Cruz, CA"
06041,399,"Marin, CA"
06113,438,"Yolo, CA"
06095,570,"Solano, CA"
06099,576,"Stanislaus, CA"
06107,577,"Tulare, CA"
06097,655,"Sonoma, CA"
06083,721,"Santa Barbara, CA"
06077,795,"San Joaquin, CA"
06029,875,"Kern, CA"
06053,1122,"Monterey, CA"
06111,1130,"Ventura, CA"
06019,1242,"Fresno, CA"
06013,1372,"Contra Costa, CA"
06081,1743,"San Mateo, CA"
06065,1784,"Riverside, CA"
06067,1809,"Sacramento, CA"
06071,1920,"San Bernardino, CA"
06075,2039,"San Francisco, CA"
06001,3648,"Alameda, CA"
06073,5351,"San Diego, CA"
06085,5889,"Santa Clara, CA"
06059,6214,"Orange, CA"
06037,22851,"Los Angeles, CA"
```

OUTPUT

```
Project 4 for DATA STRUCTURE 1 (PROF. T.VO)
Author: Mai Pham

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 4

Displaying the record.
ID: 06001     Population: 3648     County/State: "Alameda, CA"
ID: 06003     Population: 0        County/State: "Alpine, CA"
ID: 06005     Population: 1        County/State: "Amador, CA"
ID: 06007     Population: 150      County/State: "Butte, CA"
ID: 06009     Population: 1        County/State: "Calaveras, CA"
ID: 06011     Population: 60       County/State: "Colusa, CA"
```

```
ID: 06013     Population: 1372     County/State: "Contra Costa, CA"
ID: 06015     Population: 19       County/State: "Del Norte, CA"
ID: 06017     Population: 90       County/State: "El Dorado, CA"
ID: 06019     Population: 1242     County/State: "Fresno, CA"
ID: 06021     Population: 36       County/State: "Glenn, CA"
ID: 06023     Population: 42       County/State: "Humboldt, CA"
ID: 06025     Population: 295      County/State: "Imperial, CA"
ID: 06027     Population: 8        County/State: "Inyo, CA"
ID: 06029     Population: 875      County/State: "Kern, CA"
ID: 06031     Population: 205      County/State: "Kings, CA"
ID: 06033     Population: 32       County/State: "Lake, CA"
ID: 06035     Population: 1        County/State: "Lassen, CA"
ID: 06037     Population: 22851    County/State: "Los Angeles, CA"
ID: 06039     Population: 221      County/State: "Madera, CA"
ID: 06041     Population: 399      County/State: "Marin, CA"
ID: 06043     Population: 1        County/State: "Mariposa, CA"
ID: 06045     Population: 102      County/State: "Mendocino, CA"
ID: 06047     Population: 341      County/State: "Merced, CA"
ID: 06049     Population: 0        County/State: "Modoc, CA"
ID: 06051     Population: 19       County/State: "Mono, CA"
ID: 06053     Population: 1122     County/State: "Monterey, CA"
ID: 06055     Population: 225      County/State: "Napa, CA"
ID: 06057     Population: 26       County/State: "Nevada, CA"
ID: 06059     Population: 6214     County/State: "Orange, CA"
ID: 06061     Population: 162      County/State: "Placer, CA"
ID: 06063     Population: 0        County/State: "Plumas, CA"
ID: 06065     Population: 1784     County/State: "Riverside, CA"
ID: 06067     Population: 1809     County/State: "Sacramento, CA"
ID: 06069     Population: 94       County/State: "San Benito, CA"
ID: 06071     Population: 1920     County/State: "San Bernardino, CA"
ID: 06073     Population: 5351     County/State: "San Diego, CA"
ID: 06075     Population: 2039     County/State: "San Francisco, CA"
ID: 06077     Population: 795      County/State: "San Joaquin, CA"
ID: 06079     Population: 171      County/State: "San Luis Obispo, CA"
ID: 06081     Population: 1743     County/State: "San Mateo, CA"
ID: 06083     Population: 721      County/State: "Santa Barbara, CA"
ID: 06085     Population: 5889     County/State: "Santa Clara, CA"
ID: 06087     Population: 373      County/State: "Santa Cruz, CA"
ID: 06089     Population: 29       County/State: "Shasta, CA"
ID: 06091     Population: 0        County/State: "Sierra, CA"
ID: 06093     Population: 4        County/State: "Siskiyou, CA"
ID: 06095     Population: 570      County/State: "Solano, CA"
ID: 06097     Population: 655      County/State: "Sonoma, CA"
ID: 06099     Population: 576      County/State: "Stanislaus, CA"
ID: 06101     Population: 172      County/State: "Sutter, CA"
ID: 06103     Population: 25       County/State: "Tehama, CA"
ID: 06105     Population: 0        County/State: "Trinity, CA"
ID: 06107     Population: 577      County/State: "Tulare, CA"
ID: 06109     Population: 3        County/State: "Tuolumne, CA"
ID: 06111     Population: 1130     County/State: "Ventura, CA"
ID: 06113     Population: 438      County/State: "Yolo, CA"
ID: 06115     Population: 105      County/State: "Yuba, CA"

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 1
Searching for a record...
Please enter the county/state code: 06113
ID: 06113     Population: 438      County/State: "Yolo, CA"
AVL Time: 4 milliseconds.
```

```
BST Time: 8 milliseconds.

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 1
Searching for a record...
Please enter the county/state code: 06071
ID: 06071     Population: 1920     County/State: "San Bernardino, CA"
AVL Time: 6 milliseconds.
BST Time: 9 milliseconds.

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 2
Inserting a record...
Please enter the county/state code, population, & name: 06222 1234 "Pasadena, CA"
Inputting...
ID: 06222     Population: 1234     County/State: "Pasadena, CA"
AVL Time: 6 milliseconds.
BST Time: 8 milliseconds.
Completed!

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 2
Inserting a record...
Please enter the county/state code, population, & name: 06022 7654 "Rosemead, CA"
Inputting...
ID: 06022     Population: 7654     County/State: "Rosemead, CA"
AVL Time: 6 milliseconds.
BST Time: 10 milliseconds.
Completed!

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 3
Deleting a record...
Please enter the county/state code to delete: 06029
AVL Time: 6 milliseconds.
BST Time: 9 milliseconds.
Completed!

Menu:
1. Search for a record
2. Insert a record
```

```
3. Delete a record
4. List all record
5. Exit

Select an option: 3
Deleting a record...
Please enter the county/state code to delete: 06073
AVL Time: 7 milliseconds.
BST Time: 11 milliseconds.
Completed!

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 4
Displaying the record.
ID: 06001     Population: 3648     County/State: "Alameda, CA"
ID: 06003     Population: 0        County/State: "Alpine, CA"
ID: 06005     Population: 1        County/State: "Amador, CA"
ID: 06007     Population: 150      County/State: "Butte, CA"
ID: 06009     Population: 1        County/State: "Calaveras, CA"
ID: 06011     Population: 60       County/State: "Colusa, CA"
ID: 06013     Population: 1372     County/State: "Contra Costa, CA"
ID: 06015     Population: 19       County/State: "Del Norte, CA"
ID: 06017     Population: 90       County/State: "El Dorado, CA"
ID: 06019     Population: 1242     County/State: "Fresno, CA"
ID: 06021     Population: 36       County/State: "Glenn, CA"
ID: 06022     Population: 7654     County/State: "Rosemead, CA"
ID: 06023     Population: 42       County/State: "Humboldt, CA"
ID: 06025     Population: 295      County/State: "Imperial, CA"
ID: 06027     Population: 8        County/State: "Inyo, CA"
ID: 06031     Population: 205      County/State: "Kings, CA"
ID: 06033     Population: 32       County/State: "Lake, CA"
ID: 06035     Population: 1        County/State: "Lassen, CA"
ID: 06037     Population: 22851    County/State: "Los Angeles, CA"
ID: 06039     Population: 221      County/State: "Madera, CA"
ID: 06041     Population: 399      County/State: "Marin, CA"
ID: 06043     Population: 1        County/State: "Mariposa, CA"
ID: 06045     Population: 102      County/State: "Mendocino, CA"
ID: 06047     Population: 341      County/State: "Merced, CA"
ID: 06049     Population: 0        County/State: "Modoc, CA"
ID: 06051     Population: 19       County/State: "Mono, CA"
ID: 06053     Population: 1122     County/State: "Monterey, CA"
ID: 06055     Population: 225      County/State: "Napa, CA"
ID: 06057     Population: 26       County/State: "Nevada, CA"
ID: 06059     Population: 6214     County/State: "Orange, CA"
ID: 06061     Population: 162      County/State: "Placer, CA"
ID: 06063     Population: 0        County/State: "Plumas, CA"
ID: 06065     Population: 1784     County/State: "Riverside, CA"
ID: 06067     Population: 1809     County/State: "Sacramento, CA"
ID: 06069     Population: 94       County/State: "San Benito, CA"
ID: 06071     Population: 1920     County/State: "San Bernardino, CA"
ID: 06075     Population: 2039     County/State: "San Francisco, CA"
ID: 06077     Population: 795      County/State: "San Joaquin, CA"
ID: 06079     Population: 171      County/State: "San Luis Obispo, CA"
ID: 06081     Population: 1743     County/State: "San Mateo, CA"
ID: 06083     Population: 721      County/State: "Santa Barbara, CA"
ID: 06085     Population: 5889     County/State: "Santa Clara, CA"
ID: 06087     Population: 373      County/State: "Santa Cruz, CA"
ID: 06089     Population: 29       County/State: "Shasta, CA"
ID: 06091     Population: 0        County/State: "Sierra, CA"
```

```
ID: 06093     Population: 4        County/State: "Siskiyou, CA"
ID: 06095     Population: 570      County/State: "Solano, CA"
ID: 06097     Population: 655      County/State: "Sonoma, CA"
ID: 06099     Population: 576      County/State: "Stanislaus, CA"
ID: 06101     Population: 172      County/State: "Sutter, CA"
ID: 06103     Population: 25       County/State: "Tehama, CA"
ID: 06105     Population: 0        County/State: "Trinity, CA"
ID: 06107     Population: 577      County/State: "Tulare, CA"
ID: 06109     Population: 3        County/State: "Tuolumne, CA"
ID: 06111     Population: 1130     County/State: "Ventura, CA"
ID: 06113     Population: 438      County/State: "Yolo, CA"
ID: 06115     Population: 105      County/State: "Yuba, CA"
ID: 06222     Population: 1234     County/State: "Pasadena, CA"

Menu:
1. Search for a record
2. Insert a record
3. Delete a record
4. List all record
5. Exit

Select an option: 5
Program ended with exit code: 0
```

SOURCE CODE

***NOTE: source codes from the book that didn't change much are omitted.**

POPULATION RECORD

```
//
//  PopulationRecord.h
//  Project
//
//  Created by Mai Pham on 11/29/17.
//  Copyright © 2017 Mai Pham. All rights reserved.
//

#ifndef PopulationRecord_h
#define PopulationRecord_h

#include <string>
#include <iostream>
using namespace std;

class PopulationRecord
{
private:
    string code;                            // county/state code
    string population;                      // population
    string name;                            // county/state name
public:
    PopulationRecord()
    {
        code = "";
        population = "";
        name = "";
    }
    PopulationRecord(string c, string p, string n)
    {
        code = c;
        population = p;
        name = n;
```

```
    }
    string getCode()
        {   return code; }
    string getPopulation()
    {   return population; }
    string getName()
    {   return name; }
};

#endif /* PopulationRecord_h */
```

MAIN DRIVER (include extra credit)

```
//
//  main.cpp
//  Project 4
//
//  Created by Mai Pham on 11/22/17.
//  Copyright © 2017 Mai Pham. All rights reserved.
//

#include "AVLTree.h"
#include "bst.h"
#include "PopulationRecord.h"
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
using namespace std;

int main()
{
    string code;                            // county/state code
    string population;                      // population
    string name;                            // county/state name

    string input;                           // input file data
    string separated;                       // separated file data
    int menu;                               // menu selection
    AVLTree avlt;                           // AVL search tree
    SearchTree bst;                         // BST tree

    cout << "Project 4 for DATA STRUCTURE 1 (PROF. T.VO)\n";
    cout << "Author: Mai Pham\n\n";

    ifstream inputFile;
    inputFile.open("p4large.txt");
    if (!inputFile)
    {
        cout << "Error opening file. \n";
        cout << "The file was not found\n\n";
        return 1;
    }

    while (!inputFile.eof())
    {
        getline(inputFile,input);
        stringstream ss(input);

        getline(ss, separated, ',');
        code = separated;
        getline(ss, separated, ',');
        population = separated;
        getline(ss, separated, '\n');
        name = separated;
```

```
        PopulationRecord rec (code, population, name);
        avlt.insert(code, rec);
        bst.insert(code, rec);
    }

    cout << "Menu: \n";
    cout << "1. Search for a record\n";
    cout << "2. Insert a record\n";
    cout << "3. Delete a record\n";
    cout << "4. List all record\n";
    cout << "5. Exit\n\n";
    cout << "Select an option: ";
    cin >> menu;
    cout << endl;
    while (menu != 5)
    {
        switch (menu)
        {
            case 1:
            {
                cout << "Searching for a record...\n";
                cout << "Please enter the county/state code: ";
                cin >> code;
                bst.find(code);                                  // for bst time purpose
                AVLTree::Iterator iterator = avlt.find(code);
                PopulationRecord temp = (*iterator).value();
                cout << "ID: " << temp.getCode()<< "\tPopulation: " << temp.getPopulation()
<< "  \tCounty/State: " << temp.getName()<< endl;
                cout << "AVL Time: " << avlt.getCount() << " milliseconds." << endl;
                cout << "BST Time: " << bst.getCount() << " milliseconds." << endl;
                break;
            }
            case 2:
            {
                cout << "Inserting a record...\n";
                cout << "Please enter the county/state code, population, & name: ";
                cin >> code >> population;
                cin.ignore();
                getline(cin, name);
                cout << "Inputting...\n";
                cout << "ID: " << code<< "\tPopulation: " << population <<
"   \tCounty/State: " << name << endl;
                PopulationRecord rec (code, population, name);
                avlt.insert(code, rec);
                bst.insert(code, rec);
                cout << "AVL Time: " << avlt.getCount() << " milliseconds." << endl;
                cout << "BST Time: " << bst.getCount() << " milliseconds." << endl;
                cout << "Completed!" << endl;
                break;
            }
            case 3:
            {
                cout << "Deleting a record...\n";
                cout << "Please enter the county/state code to delete: ";
                cin >> code;
                avlt.erase(code);
                bst.erase(code);
                cout << "AVL Time: " << avlt.getCount() << " milliseconds." << endl;
                cout << "BST Time: " << bst.getCount() << " milliseconds." << endl;
                cout << "Completed!" << endl;
                break;
            }
            case 4:
            {
```

```
                cout << "Displaying the record.\n";
                AVLTree::Iterator iterator = avlt.begin();
                ++iterator;
                while(!(iterator == avlt.end()))
                {
                    PopulationRecord temp = (*iterator).value();
                    cout << "ID: " << temp.getCode()<< "\tPopulation: " <<
temp.getPopulation() << "   \tCounty/State: " << temp.getName()<< endl;
                    ++iterator;
                }
                break;
            }
        }
        cout << "\nMenu: \n";
        cout << "1. Search for a record\n";
        cout << "2. Insert a record\n";
        cout << "3. Delete a record\n";
        cout << "4. List all record\n";
        cout << "5. Exit\n\n";
        cout << "Select an option: ";
        cin >> menu;
    }
    return 0;
}
```

BST (SEARCH TREE)

```
#ifndef BST_H
#define BST_H
// Modified for CSCI 220 Fall 15

#include <string>
#include "PopulationRecord.h"
#include "BinaryTree.h"
#include "RuntimeException.h"

class SearchTree {                                          // a binary search tree
public:
    class Iterator;                                         // an iterator/position
private:                                                    // member data
    BinaryTree T;                                           // the binary tree
    int n;                                                  // number of entries
protected:
    int count;
public:
    SearchTree(): T(), n(0)
    { count = 0;T.addRoot(); T.expandExternal(T.root()); }      // create the super root
    int size() const;                                           // number of entries
    bool empty() const;                                         // is the tree empty?
    int getCount()
        { return count;}
    Iterator find(const string& k)
    {
        count = 0;
        TPos v = finder(k, root());                             // search from virtual root
        if (!v.isExternal()) return Iterator(v);                // found it
        else return end();                                      // didn't find it
    }
    Iterator insert(const string& k, const PopulationRecord& x)       // insert (k,x)
        { count = 0;TPos v = inserter(k, x); return Iterator(v); }
    void erase(const string& k)
    {
        count = 0;
        TPos v = finder(k, root());                 // search from virtual root
        if (v.isExternal())                         // not found?
```

```
                throw NonexistentElement("Erase of nonexistent");
        eraser(v);                                          // remove it
    }
    void erase( Iterator& p)                                // remove entry at p
        { eraser(p.v); }
    Iterator begin()
    {
        TPos v = root();                                    // start at virtual root
        while (!v.isExternal()) v = v.left();               // find leftmost node
        return Iterator(v.parent());
    }
    Iterator end()                                          // iterator to end entry
        { return Iterator(T.root()); }                      // return the super root

protected:                                                  // local utilities
    typedef BinaryTree::Position TPos;                      // position in the tree
    TPos root() const    { return T.root().left(); }       // left child of super root
    TPos finder(const string& k, const TPos& v)
    {
        count++;
        if (v.isExternal()) return v;                                // key not found
        if (k < v.v->elt.key()) return finder(k, v.left());          // search left subtree
        else if (v.v->elt.key() < k) return finder(k, v.right());// search right subtree
        else return v;                                                   // found it here
    }
    TPos inserter(const string& k, const PopulationRecord& x)
    {
        TPos v = finder(k, root());                             // search from virtual root
        while (!v.isExternal())                                 // key already exists?
                v = finder(k, v.right());                       // look further
        T.expandExternal(v);                                    // add new internal node
        (*v).setKey(k); (*v).setValue(x);                       // set entry
        n++;                                                    // one more entry
              return v;                                         // return insert position
    }
    TPos eraser(TPos& v)
    {
        TPos w;
        if (v.left().isExternal()) w = v.left();            // remove from left
        else if (v.right().isExternal()) w = v.right();     // remove from right
        else {                                              // both internal?
              w = v.right();                                // go to right subtree
              do { w = w.left(); } while (!w.isExternal());    // get leftmost node
              TPos u = w.parent();
              (*v).setKey((*u).key()); (*v).setValue((*u).value());// copy w's parent to v
        }
        n--;                                                // one less entry
        return T.removeAboveExternal(w);                    // remove w and parent
    }

    TPos restructure(const TPos& v)                         // restructure
    {
        TPos x, y, z, a, b, c, t0, t1, t2, t3;

        // node x with parent y and grandparent z
        x = v;
        y = x.v->par;
        z = y.v->par;

        if (z.left().v == y.v)
        {
            if (y.left().v == x.v)
            {
                a = x;
                b = y;
```

```
                c = z;
                t0 = x.left();
                t1 = x.right();
                t2 = y.right();
                t3 = z.right();
            }
            else
            {
                a = y;
                b = x;
                c = z;
                t0 = y.left();
                t1 = x.left();
                t2 = x.right();
                t3 = z.right();
            }
        }
        else
        {
            if (y.left().v == x.v)
            {
                a = z;
                b = x;
                c = y;
                t0 = z.left();
                t1 = x.left();
                t2 = x.right();
                t3 = y.right();
            }
            else
            {
                a = z;
                b = y;
                c = x;
                t0 = z.left();
                t1 = y.left();
                t2 = x.left();
                t3 = x.right();
            }
        }

        TPos w = z.v->par;
        if (z.v == w.v->right)
            w.v->right = b.v;
        else
            w.v->left = b.v;

        b.v->par = w.v;

        // Let a be the left child of b and let T0 and T1 be the left and right subtrees of
a, respectively.
        b.v->left = a.v;
        a.v->par = b.v;
        t0.v->par = a.v;
        a.v->left = t0.v;
        t1.v->par = a.v;
        a.v->right = t1.v;

        // Let c be the right child of b and let T2 and T3 be the left and right subtrees
of a, respectively.
        b.v->right = c.v;
        c.v->par = b.v;
        c.v->left = t2.v;
        t2.v->par = c.v;
        c.v->right = t3.v;
```

```
        t3.v->par = c.v;

        return b;
    }  // throw(BoundaryViolation);

public:
    // ...insert Iterator class declaration here
    class Iterator {                                    // an iterator/position
    private:
      TPos v;                                           // which entry
    public:
      Iterator(const TPos& vv) : v(vv) { }            // constructor
      //const AVLEntry& operator*() const { return *v.v; }     // get entry (read only)
      AVLEntry& operator*() { return *v; }                    // get entry (read/write)
      bool operator==(const Iterator& p) const                // are iterators equal?
        { return v.v == p.v.v; }
      Iterator& operator++(){
              TPos w = v.right();
              if (!w.isExternal()) {                          // have right subtree?
                do { v = w; w = w.left(); }                   // move down left chain
                while (!w.isExternal());
              }
              else {
                w = v.parent();                               // get parent
                while (v.v == w.right().v)                    // move up right chain
                     { v = w; w = w.parent(); }
                v = w;                                        // and first link to left
              }
              return *this;
         }

      friend class SearchTree;                                 // give search tree access
    };
};

#endif
```