No banned STL containers used, no bugs to my knowledge

RadixTree testing

RadixTree<int> a;

a.search("gfhgjh");

a.insert("watering", 8);

a.insert("waterined", 9);

a.insert("water", 1);

a.insert("slow", 2);

a.insert("water", 7);

a.insert("slower", 3);

a.insert("waste", 4);

a.insert("watch", 5);

a.insert("wa", 6);

//

int\* v = nullptr;

v= a.search("water"); assert((\*v) == 7);

v = nullptr; v= a.search("slow"); assert((\*v) == 2);

v = nullptr; v= a.search("slower"); assert((\*v) == 3);

v = nullptr; v= a.search("waste"); assert((\*v) == 4);

v = nullptr; v= a.search("watch"); assert((\*v) == 5);

v = nullptr; v= a.search("wa"); assert((\*v) == 6);

v = nullptr; v= a.search("wat"); assert(v == nullptr);

## PersonProfile testing

PersonProfile p1("john","johnny@gmail.com");

attval.attribute = "hobby";

attval.value = "eating";

p1.AddAttValPair(attval);

attval.attribute = "dreams";

attval.value = "unicorn";

p1.AddAttValPair(attval);

attval.attribute = "dreams";

attval.value = "peach";

p1.AddAttValPair(attval);

cerr << p1.GetName() << " "<< p1.GetEmail() << " numattvalpairs " << p1.GetNumAttValPairs()<< endl;

AttValPair targetPair("eats","bread");

assert ( !p1.GetAttVal(3, targetPair)&& targetPair.attribute == "eats" && targetPair.value == "bread" );

assert(p1.GetAttVal(0, targetPair) && targetPair.attribute == "hobby" && targetPair.value == "eating");

assert(p1.GetAttVal(1, targetPair) && targetPair.attribute == "dreams" && targetPair.value == "unicorn");

assert(p1.GetAttVal(2, targetPair) && targetPair.attribute == "dreams" && targetPair.value == "peach");

## MemberDatabase testing

MemberDatabase dataBase;

assert(dataBase.LoadDatabase(MEMBERS\_FILE));

assert(dataBase.GetMemberByEmail("AbFow2483@charter.net") != nullptr);

assert(dataBase.GetMemberByEmail("ELand@live.com") != nullptr);

assert(dataBase.GetMemberByEmail("ab@bjnk.com") == nullptr);

AttValPair attval("hobby","photography");

assert(dataBase.FindMatchingMembers(attval).size() == 6178);

attval.attribute = "hobby";

attval.value = "painting";

// cerr << dataBase.FindMatchingMembers(attval).size() << endl;

assert(dataBase.FindMatchingMembers(attval).size() == 6280); //only repeated hobby in file

attval.attribute = "bhjhvjbhknjl";

assert(dataBase.FindMatchingMembers(attval).size() == 0);

attval.attribute = "job";

attval.value = "psychologist";

assert(dataBase.FindMatchingMembers(attval).size() == 4336);

## AttributeTranslator testing

AttributeTranslator translate;

assert(translate.Load(TRANSLATOR\_FILE));

attval.attribute = "hobby";

attval.value = "bowling";

vector<AttValPair> results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() == 8);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

attval.attribute = "trait";

attval.value = "wishabrasive";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==1);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

cerr <<endl << "matchmaker testing/////////////////" << endl << endl;

MatchMaker matching(dataBase, translate);

//check that the translations match

// get list of translations, put them in alphabetical order, take out source traits, and then textcompare

// job,accountant

attval.attribute = "job"; attval.value = "accountant";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==18);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

// hobby,origami

attval.attribute = "hobby"; attval.value = "origami";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==7);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

// trait,narrow

attval.attribute = "trait"; attval.value = "narrow";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==1);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

// trait,quirky

attval.attribute = "trait"; attval.value = "quirky";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==4);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

// trait,pedantic

attval.attribute = "trait"; attval.value = "pedantic";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==1);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

// trait,uncritical

attval.attribute = "trait"; attval.value = "uncritical";

results = translate.FindCompatibleAttValPairs(attval);

cerr << endl<<attval.attribute << " " <<attval.value << "~~~" << endl;

assert(results.size() ==1);

for (int i = 0; i<results.size();i++)

{

cerr << results[i].attribute << " " << results[i].value << endl;

}

cerr << endl;

matching.IdentifyRankedMatches("MHa7@sky.com", 3);

### Matchmaker Testing

I created the MatchMaker class last tested it by using the provided main function. I compared my output with that of the demo using text compare and also checked that everything was working properly using the debugger and with a smaller version of the member and translator files for which I could print out the corresponding attribute value pairs.