Software Test Plan for Employee Record Database

Programming Assignment 3
March 16, 2020

Prepared By
Paul Abers
pa0034@uah.edu

Prepared For
Mr. James Williamson
CS 221, Data Structures in C++
Computer Science Department
University of Alabama in Huntsville

Contents

1 System Overview

The purpose of this assignment is to provide a simple and easy way to access an employee record database. The database consists of three classes, an employee record class, store class and customer list class. The employee record class must store an employee ID, employee name, department, annual salary of each employee, a pointer to a left and right employee record and a customer list for supported stores by each employee. The store class stores a pointer to the next store object (for the customer list) as well as the store id, name, address, city, state and zip code. The customer list stores a list of stores ordered by store id. An employee database that stores an input data file and a pointer to the root of a tree for employee records.

2 Referenced Documents

Programming Assignment 1: Statement of Work. Programming Assignment 2: Statement of Work. Programming Assignment 3: Statement of Work. Software Design Document for the Emplyoee Record Database program.

3 Test Procedures

The following tests will be performed on the software before its release.

3.1 Source File: EmployeeRecord.h and EmployeeRecord.cpp

3.1.1 Function: EmployeeRecord()

3.1.1.1 Purpose and procedure

This test will determine if the EmployeeRecord constructor is working correctly.

3.1.1.2 Inputs

First, a call to the default constructor. Then a call to constructor that takes all inputs.

3.1.1.3 Expected Output

The employee record to be initialized to the default state or the input initialized state depending on the call made to the constructor.

3.1.1.4 Success Criteria

The employee record will be checked by calling the constructor and then comparing its state with calls to all the get methods as well as printRecord(). All should match the expected values that were set by either the default or input initialized constructor calls.

3.1.2 Function: EmployeeRecord()

3.1.2.1 Purpose and procedure

This test will determine if the destructor for the employee record class is working properly. A test employee record already set will be supplied.

3.1.2.2 Inputs

An employee record class with all values already set.

3.1.2.3 Expected Output

The employee record class to properly deallocate memory from all internal variables.

3.1.2.4 Success Criteria

Initialize multiple employee record classes and destruct them. Make sure that memory for the program does not keep increasing after employee records are destroyed.

3.1.3 Function: getID()

3.1.3.1 Purpose and procedure

This test will determine if the getId() function is working correctly. A test employee record already set will be supplied.

3.1.3.2 Inputs

An employee record class with employee id set to a known integer value.

3.1.3.3 Expected Output

The employee record's currently set value.

3.1.3.4 Success Criteria

The int returned by getID() matches the int value set for the employee id.

3.1.4 Function: setID()

3.1.4.1 Purpose and procedure

This test will determine if the setId() function is working correctly. A test employee record already set will be supplied.

3.1.4.2 Inputs

An employee record class with employee id set to a known integer value. Employee record's id will then be set to a new value.

3.1.4.3 Expected Output

The employee record's new int value to be returned by getID().

3.1.4.4 Success Criteria

The int returned by getID() matches the int value set for the employee id.

3.1.5 Function: getName()

3.1.5.1 Purpose and procedure

This test will determine if the getName() function is working correctly. A test employee record already set will be supplied.

3.1.5.2 Inputs

An employee record class with employee first name and last name set to known character array values.

3.1.5.3 Expected Output

The employee record's currently set values for the first and last name.

3.1.5.4 Success Criteria

The character arrays passed into getName to be changed to the current values for the employee record's first and last names.

3.1.6 Function: setName()

3.1.6.1 Purpose and procedure

This test will determine if the setName() function is working correctly. A test employee record already set will be supplied.

3.1.6.2 Inputs

An employee record class with employee first name and last name set to known character array values.

3.1.6.3 Expected Output

The employee record's first name and last name to be changed to the first and last name passed to the function.

3.1.6.4 Success Criteria

The employee record's first and last name to start at one known value, be set by the call to the function and the values copied by getName() to match those passed by setName().

3.1.7 Function: getDept()

3.1.7.1 Purpose and procedure

This test will determine if the getDept() function is working correctly. A test employee record already set will be supplied.

3.1.7.2 Inputs

An employee record class with employee department set to a known integer value.

3.1.7.3 Expected Output

The employee record's currently set department value to be copied into the reference variable passed into the function.

3.1.7.4 Success Criteria

The int value copied into the passed reference variable to match the int value of the employee record's department value.

3.1.8 Function: setDept()

3.1.8.1 Purpose and procedure

This test will determine if the setDept() function is working correctly. A test employee record already set will be supplied.

3.1.8.2 Inputs

An employee record class with employee department set to a known integer value.

3.1.8.3 Expected Output

The employee record's department value to be set to the new value passed by the call of the method setDept().

3.1.8.4 Success Criteria

The department to start as a known integer, be set to a new integer value, and the subsequent call to getDept() to match the value of its referenced argument with that of the value passed to setDept().

3.1.9 Function: getSalary()

3.1.9.1 Purpose and procedure

This test will determine if the getSalary() function is working correctly. A test employee record already set will be supplied.

3.1.9.2 Inputs

An employee record class with employee salary set to a known integer value.

3.1.9.3 Expected Output

The employee record's currently set salary value.

3.1.9.4 Success Criteria

The double copied into the pointer by getID() matches the double value for the current employee's sacurrent value for the employee's salary.

3.1.10 Function: setSalary()

3.1.10.1 Purpose and procedure

This test will determine if the setSalary() function is working correctly. A test employee record already set will be supplied.

3.1.10.2 Inputs

An employee record class with employee salary set to a known integer value.

3.1.10.3 Expected Output

The employee record's salary to be set to the new salary.

3.1.10.4 Success Criteria

The salary to start as a known double, be set to a new double value, and the subsequent call to getSalary() to match the value of its pointer argument with that of the value passed to setDept().

3.1.11 Function: printRecord()

3.1.11.1 Purpose and procedure

This test will determine if the printRecord() function is working correctly. A test employee record already set will be supplied.

3.1.11.2 Inputs

An employee record class with all of its member variables set to known values.

3.1.11.3 Expected Output

Proper display of employee record to screen.

3.1.11.4 Success Criteria

All member variables to be properly printed to screen with their known values.

3.1.12 Function: removeCustomerList()

3.1.12.1 Purpose and procedure

This test will determine if the removeCustomerList() function is working correctly. A test employee record already set will be supplied.

3.1.12.2 Inputs

An employee record class with all of its member variables set to known values.

3.1.12.3 Expected Output

Proper removal of customer list and pointer set to null

3.1.12.4 Success Criteria

Pointer to customer list object to be set to null.

3.1.13 Function: destroyCustomerList()

3.1.13.1 Purpose and procedure

This test will determine if the destroyCustomerList() function is working correctly. A test employee record already set will be supplied.

3.1.13.2 Inputs

An employee record class with all of its member variables set to known values.

3.1.13.3 Expected Output

Proper deletion of customer list and pointer set to null

3.1.13.4 Success Criteria

Pointer to customer list object to be set to null and customer list deleted.

3.2 Source File: CustomerList.h and CustomerList.cpp

3.2.1 Function: CustomerList()

3.2.1.1 Purpose and procedure

This test will determine if the CustomerList constructor is working correctly.

3.2.1.2 Inputs

Instantiate an empty CustomerList object

3.2.1.3 Expected Output

The CustomerList to be initialized to an empty list.

3.2.1.4 Success Criteria

The CustomerList will be checked by calling the constructor and then calling a print of CustomerList to check if it is empty and can be called upon.

3.2.2 Function: ˜CustomerList()

3.2.2.1 Purpose and procedure

This test will determine if the CustomerList destructor is working correctly.

3.2.2.2 Inputs

A CustomerList object

3.2.2.3 Expected Output

A destroyed customer list object

3.2.2.4 Success Criteria

The CustomerList will be checked by calling the constructor and then destroying it repeatedly and checking if there is a memory leak.

3.2.3 Function: addStore()

3.2.3.1 Purpose and procedure

This test will determine if the add store function is working properly.

3.2.3.2 Inputs

A known customer list object.

3.2.3.3 Expected Output

The store specified with add store added to the customer list object and a retrurn of true from call to addStore.

3.2.3.4 Success Criteria

The customer list printed, store added, then customer list printed again. The store added should be in the correct position within the list.

3.2.4 Function: removeStore()

3.2.4.1 Purpose and procedure

This test will determine if the CustomerList remove store function is working correctly.

3.2.4.2 Inputs

a known customer list object

3.2.4.3 Expected Output

a pointer to a store object with the id specified in call (or null if id not in list)

3.2.4.4 Success Criteria

The store returned from call to have the same id as that specified in the argument. The customer list no longer containing the store with specified id.

3.2.5 Function: getStore()

3.2.5.1 Purpose and procedure

This test will determine if the CustomerList get store function is working correctly.

3.2.5.2 Inputs

A known customer list object

3.2.5.3 Expected Output

The store specified by the id passed to the function, or null if id not in customer list.

3.2.5.4 Success Criteria

The store returned from call to have the same id as that specified in the argument (or null if the id was not in the customer list).

3.2.6 Function: updateStore()

3.2.6.1 Purpose and procedure

This test will determine if the CustomerList update store function is working correctly.

3.2.6.2 Inputs

a known customer list object

3.2.6.3 Expected Output

A boolean with true if store updated and false if store update failed.

3.2.6.4 Success Criteria

The store specified by the id argument to have its values updated to the values passed in the call. A return of true if id in the customer list and a return of false if id not in the customer list.

3.2.7 Function: printStoresInfo()

3.2.7.1 Purpose and procedure

This test will determine fi the CustomerList printStoresInfo function is working correctly.

3.2.7.2 Inputs

A known customer list object

3.2.7.3 Expected Output

A print to screen of all the store info for each store in customer list in the proper order.

3.2.7.4 Success Criteria

The print to screen to have the proper information in the proper order for the known CustomerList object.

3.3 Source File: EmployeeDatabase.h and EmployeeDatabase.cpp

3.3.1 Function: EmployeeDatabase()

3.3.1.1 Purpose and procedure

This test will determine if the EmployeeDatabase constructor is working correctly.

3.3.1.2 Inputs

Instantiate an empty EmployeeDatabase object

3.3.1.3 Expected Output

The EmployeeDatabase to be initialized to an empty tree.

3.3.1.4 Success Criteria

The EmployeeDatabase will be checked by calling the constructor and then calling a print of EmployeeDatabase to check if it is empty and can be called upon.

3.3.2 Function: ˜EmployeeDatabase()

3.3.2.1 Purpose and procedure

This test will determine if the EmployeeDatabase destructor is working correctly.

3.3.2.2 Inputs

An EmployeeDatabase object

3.3.2.3 Expected Output

The EmployeeDatabase to be destructed properly.

3.3.2.4 Success Criteria

The EmployeeDatabase will be checked by creating a large number and destroying them and checking for any memory leaks.

3.3.3 Function: addEmployee()

3.3.3.1 Purpose and procedure

This test will determine if the add employee function is working correctly.

3.3.3.2 Inputs

An already created employee database tree

3.3.3.3 Expected Output

The EmployeeDatabase to insert a new employee record

3.3.3.4 Success Criteria

The EmployeeDatabase will be checked by calling the add employee function and then calling a print of EmployeeDatabase to check if it was inserted correctly.

3.3.4 Function: getEmployee()

3.3.4.1 Purpose and procedure

This test will determine if the get employee function is working correctly.

3.3.4.2 Inputs

An already created employee database tree

3.3.4.3 Expected Output

The EmployeeDatabase to get an employee record by id

3.3.4.4 Success Criteria

The EmployeeDatabase will be checked by calling the get employee function and checking if the return employee record has the same id as the one supplied in the call.

3.3.5 Function: removeEmployee()

3.3.5.1 Purpose and procedure

This test will determine if the remove employee function is working correctly.

3.3.5.2 Inputs

An already created employee database tree

3.3.5.3 Expected Output

The EmployeeDatabase to remove an employee record based on the id supplied.

3.3.5.4 Success Criteria

The EmployeeDatabase will be checked by calling the remove employee function and then calling a print of EmployeeDatabase to check if it was removed correctly. The returned employee record's id will also be checked for a match with the supplied id.

3.3.6 Function: printEmployeeRecords()

3.3.6.1 Purpose and procedure

This test will determine if the print employee records function is working correctly.

3.3.6.2 Inputs

An already created employee database tree

3.3.6.3 Expected Output

The EmployeeDatabase printed to match the known database.

3.3.6.4 Success Criteria

The EmployeeDatabase will be checked by calling a print of EmployeeDatabase to check if it matches the known employee database.

3.3.7 Function: buildDatabase()

3.3.7.1 Purpose and procedure

This test will determine if the build database function is working correctly.

3.3.7.2 Inputs

A character array for the input data file.

3.3.7.3 Expected Output

The EmployeeDatabase to build a tree corresponding to the input data file.

3.3.7.4 Success Criteria

The EmployeeDatabase will be checked by calling the build database function and then calling a print of EmployeeDatabase to check if it was built correctly.