CIS*2750, Winter 2018
Required data structure APIs

List

Description

Our assignments will use a list API. You have been provided with a header file `LinkedListAPI.h`. You will need to update a file `LinkedListAPI.c`, which implements all of the functions specified in the header. Most of the list has already been provided for you (see SimpleList.c in Week 1). You just need to add a couple of functions and make a few changes.

The list API list will become part of Assignment 1 submission and you do not need to submit it separately.

This API is very similar to the list you've created in CIS*2520 last summer/fall, with a few notable changes:
- Some functions have been removed or renamed.
- Some of the functions pass the List argument by value, and return a List value, not a pointer to it. This is done to minimize manual memory management and to enforce immutability where necessary (the `const` keyword in C is very limited in what it can "protect"). If this looks off, it should - this is in line with the coding practices used by many of the modern languages in the C family (notably C# and Swift), but it is not standard C practice.
- The list stores its length and has a function for retrieving list length. The length must be automatically updated when you insert/delete elements, and when the list is cleared.
- The list API contains a function for searching the list for a specific entry using a comparator function.

As a result, please read the header file carefully.

Testing

Assignments 1 and 2 - and possibly some aspects of Assignments 3 and 4 - will bested using a test harness. To give you some practice and make sure that your list matches the specifications, you have been provided with an automated test harness (ListTest.zip).

To use the harness:
- Place your `LinkedListAPI.c` into the `studentCode` directory
- Compile the code by typing `make`
- Run the tests from the main directory of the harness (i.e. the one that contains the Makefile) by executing `bin/listTests`

Make sure you compile and run your code in the lab or on the **linux.socs.uoguelph.ca** server. You can do this in the labs, or remotely using SSH or NoMachine.

Tree

Description

The tree API is also very similar to the API used in the summer/fall CIS2520 classes. The tree will be particularly useful in later assignments, but even A1 will require you to write a function that would create and return a family tree for a specific individual from a GEDCOM record.

If you haven't implemented a tree API before, don't panic. A tree is just a just a bushy list. Review your 2520 notes, and use the List API as a reference.

The API has been updated to be consistent with the List API:
- Some of the functions pass the Tree argument by value, and return a Tree value, not a pointer to it.
- Some types have been renamed
- isXXX/hasXXX functions return `bool`, not `int`

Again, read the `BinarySearchTreeAPI.h` header and refactor (update) your API.

Testing

Testing the updated Tree API is your responsibility. However, I assume you have some old test cases from your past curses, which you can repurpose with minimal modifications. The Tree test suite will be included in the test harness used to grade Assignment 1.