This is an individual assignment. Seeking direct help from students, tutors, and websites such as chegg or stack overflow will be construed as a violation of the honor code.

Semester Project Part 3: Managing the Stuff in my Closets using STLs and BSTs.

Data Structures and Analysis of Algorithms, akk5

Objectives

- To strengthen student's knowledge of C++ programming
- To give student experience reading and parsing strings of commands
- To give student experience in writing a non-linear data structure
- To give the student experience implementing a BST

Instructions

For this assignment you must write a program that implements and manages several user defined BSTs. Each BST will represent a container/location and the inventory of items it stores. This means that your BSTs should store the name of the container/location and a collection of items; each item will have a name and how many of them are in the container. You should use the name of each item as your key for creating the BST; should a user attempt to add more of the same item, just update the item count through addition. You can use the list or vector STL to manage your collection of BSTs.

Your program should implement a command line (text-based interface) capable of handling the following commands:

exit – exits the program

load *<file>* - parses the contents of the file as if they were entered from the command line

display *<container>* in – displays the contents of the specified container in order

display *<container>* pre – displays the contents of the specified container pre order

display *<container>* post – displays the contents of the specified container post order

find *<item>* - searches each container for the specified item, displaying the container name, item name, and item count each time it is found. Should inform the user on a failure.

find *<item>* in *<container>* - searches the given container for the specified item, displaying the item name and item count if it is found. Should inform the user on a failure.

remove *<item>* - Removes the specified item from each container. Should inform the user on a failure.

remove *<item>* from *<container>* - Removes the specified item from the given container. Should inform the user on a failure.

This is an individual assignment. Seeking direct help from students, tutors, and websites
such as chegg or stack overflow will be construed as a violation of the honor code.

insert *<item> <count>* into *<container>* - Inserts count of the specified item into the given container.

create *<container>* - Creates the specified container and adds it to the list of containers. Should report failure if a container of the same name already exists.

destroy <container> - Destroys the specified container and its contents. Should report failure if a container of that name doesn't exist.

Guidance

Both your BST class and your Node class will have to store additional data for this project to work. Keep that in mind as you plan and implement your program.

Pay close attention to your Node class, you can benefit from overloading the <, >, ==, and << operators. These overloads aren't strictly necessary, but they do make the code you will write cleaner and easier to read.

This is an individual assignment. Seeking direct help from students, tutors, and websites such as chegg or stack overflow will be construed as a violation of the honor code.

Grading Breakdown

Point Breakdown	
Structure	12 pts
The program has a header comment with the required information.	3 pts
The overall readability of the program.	3 pts
Program uses separate files for main and class definitions	3 pts
Program includes meaningful comments	3 pts
Syntax	28 pts
Implements Class Node correctly	9 pts
Implements Class BST correctly	19 pts
Behavior	60 pts
Program handles all command inputs properly	
• Exit the program	5 pts
• Display the container in order	5 pts
• Display the container pre order	5 pts
• Display the container post order	5 pts
• Load a valid file	5 pts
• Find an item in all containers	5 pts
• Find an item in a container	5 pts
• Remove an item from all containers	5 pts
• Remove an item from a container	5 pts
• Insert an item into a container	5 pts
• Create a container	5 pts
• Destroy a container	5 pts
Total Possible Points	**100pts**
Penalties	
Program does NOT compile	-100
Late up to 24 hrs	-30
Late more than 24hrs	-100

This is an individual assignment. Seeking direct help from students, tutors, and websites such as chegg or stack overflow will be construed as a violation of the honor code.

Header Comment

At the top of each program, type in the following comment:

/*

Student Name: <student name>

Student NetID: <student NetID>

Compiler Used: <Visual Studio, GCC, etc.>

Program Description:

<Write a short description of the program.>

*/

Example:

/*

Student Name: John Smith

Student NetID: jjjs123

Compiler Used: Eclipse using MinGW

Program Description:

This program prints lots and lots of strings!!

*/

Assignment Information

Due Date: 3/1/2020

Files Expected:

1. Main.cpp – File containing function main
2. BST.h - File containing the BST and Node class definitions.
3. BST.cpp - File containing the BST for the List and Node methods.

This is an individual assignment. Seeking direct help from students, tutors, and websites such as chegg or stack overflow will be construed as a violation of the honor code.