Programming Assignment #5

General

This project will give you experience writing a data structure in C/C++. More importantly, you will get to put some serious effort into using good algorithms in your programs and analyzing their time complexity. Part of your grade for this project will be based on how efficient your programs are. We will attempt to ascertain if your algorithms require time proportional to N^2, $Nlog_2N$, N, or log_2N.

Your Mission

Edit the file " Project5.cpp". You should not need to change any other files. You must implement all of the functions defined for the Set ADT (see Set.h for a complete list). I've taken care of several of these functions to get you started. However, the work that remains is substantial. Get started early.

General Design

I insist that you keep the elements inside the Set sorted. For example, whatever is stored in elements[0] should be smaller than any other element in the set. Whatever is in elements[1] should be the second smallest, etc.

How Sets Work

Recall that a set is an unordered collection of things. This means that the order in which you put elements into the data structure will not necessarily be the order in which they are stored. Your Set class will hold a collection of type `int`. The operations you must implement include inserting and removing elements from the set. Please note that the same element **cannot** appear twice in the same set. If `insertSet` is called where the parameter is a number that's already in the set, your function should do nothing. Similarly, `removeSet` should remove an element from the set. If someone attempts to remove an element that's not in the set, then you should do nothing.

In addition to inserting and removing elements, you must provide a membership test. In other words, it must be possible to determine if some arbitrary number is a member in your Set. The `isMemberSet` function should provide this capability. Two sets can also be compared. If the sets have exactly the same elements then they are equal. Note that two sets that each contain zero elements are equal. A set, X is a subset of another set Y if all of the elements in X are also elements of Y. So, a set containing zero elements is a subset of every other set. Also, every set is a subset of itself.

The interesting functions for sets are intersection, union and subtraction. Recall that the intersection of two sets is the elements that are common to both sets. So, if X has the elements {1, 3, 5, 7} and Y has the elements {1, 2, 3, 4} then the intersection of X and Y is a set with elements {1, 3}. The union of two sets is the set of elements that are in either one of the two sets. So, for the same X and Y above, the union would be a set containing {1, 2, 3, 4, 5, 7}. Finally, set subtraction is defined as the set of all elements that are in the first set but are not in the second set. So, if we ask for X-Y we would compute the set {5, 7}. You must write all three of these functions.

Performance

For this project we are interested in how efficiently your programs will run. Specifically we're interested in determining for each function you write whether the function takes time proportional to the number of elements in the set, the logarithm of the number of elements in the set, or the square of the number of elements in the set. The best way to determine the efficiency of your algorithm is to analyze the code by hand and count how many operations must be performed. We'll be trying to write a program that will automatically guess the performance. Such programs are very difficult to write, so don't put too much faith in the estimates it gives (it almost always is wrong for isEqualTo). Do your own analysis (besides it will be a great way to study for the next test). For each of the following functions, the worst - case time complexity must be as described below:

Function	Time Complexity (worst case, N is the number of elements in all sets of the operation).
insert	O(N)
remove	O(N)
isMember	O(log(N))
isEqualTo	O(N) (not graded on time complexity)
isSubsetOf	O(N)
unionIn	O(N)
intersectFrom	O(N)
subtractFrom	O(N)

Memory Leaks

Use Valgrind to check for memory leaks. If your solution is implemented correctly, you should pass every test case, meet the above time complexity numbers, and have no memory leaks of any kind when run on the ECE Linux server.

FAQ

Q: Do I need to check for well-formed input?

A: No, we will only pass in inputs that are well-formed.

Q: Will I be asked to merge two sets where they both point to the same place?

A: No, see above.

Q: Why aren't the time complexities of `insertSet` and `removeSet` tested?

A: We just did not do those. You should be able to look at your code to determine the time complexity.

Q: Running Valgrind causes my code to go from linear to super linear. Is this ok?

A: Valgrind is probably slowing everything down. Your time complexities should be checked while not running Valgrind.

Q: Can we assume that all sets given to us will be ordered?

A: Yes. Our implementation of a set in C is an ordered list, and you will not have to check to make sure that this is the case. Note, however, that even though the underlying structure is ordered, the user does not need to know about that. The ordering is just to make it faster to work with. When coding, remember that you are creating the mathematical concept of a set that has no order. So functions like subset should not pay attention to the order of the elements.

Q: Can we use standard C libraries?

A: Yes, you can if they are helpful, but they are not required. Remember that C++ libraries are not allowed. C libraries have a .h at the end and are included with ¡ ¿.

Q: Can we create our own separate functions?

A: You can make helper functions but they are not necessary or required.

Q: What should strCompare output when one string is a prefix of another? For example, if the two strings were "apple" and "apple pen", which would be considered greater?

A: "apple pen" is greater- you can think of the "space" character (or really any character) as being greater than the null character. Thus, the prefix is always considered less.

Q: So throughout Project5.cpp there are many functions that return the bool type rather than an integer to represent true/false. Does this mean that for this project we can use booleans?

A: Yes, just use boolean type.

Notes: This lab is much more concerned with time complexity than space complexity, so wasting a little space is fine. It might be helpful to make `maximum_set_size` smaller in main for testing purposes. When you're done testing and fixing bugs, though, make sure to change it back to 100 and retest.

CHECKLIST – Did you remember to:

- ☐ Re-read the requirements after you finished your program to ensure that you meet all of them?
- ☐ Make sure that your program passes all our testcases?
- ☐ Check for memory leaks?
- ☐ Make up your own testcases?
- ☐ Run your solution on the Linux servers?
- ☐ Upload your solution to Canvas?

..