CSCI 230 -- PA 6

Sets and Union/Find Structures

Feel free to discuss and help each other out but does not imply that you can give away your code or your answers! Make sure to read all instructions before attempting this lab.

You can work with a lab partner and each one must submit the same PDF file (include both names in the submission file). Each person must include a brief statement about your contribution to this assignment.

You must use an appropriate provided template from Canvas and output "Author: Your Name(s)" for all your programs. If you are modifying an existing program, use "Modified by: Your Name(s)".

Implement two of the exercises below:

- Use classes **Merge**, **UnionMerge**, **IntersectMerge**, and **SubtractMerge** from C++ book to perform basic set operations. Set up a driver to test the three set operations: union, intersection, and difference. There is no equivalent code in Java, but you can set up your own code using the four classes above as a guide.

- Set up a class named **IntSet** that keeps track a set of int values from 0 to 999 and you can perform basic operations like creating a set (constructor that accepts an array of int values), insert(e), remove(e), find(e), and print(). Set up three methods/functions to perform union, intersection, and difference of two sets and return the new set as shown below. Try set1 = {1, 4, 6}, set2 = {2, 4, 8, 10}.
    - set3 = setUnion(set1, set2); // set3 = {1, 2, 4, 6, 8, 10}
    - set3 = setInter(set1, set2); // set3 = {4}
    - set3 = setDiff(set1, set2); // set3 = {1, 6}

- Implement the find/union partition structure as a list-based approach as discussed in class/book.

- **Must be done in C++ unless it is for extra credit.** Implement the find/union partition structure as a tree-based approach. In Java, use Partition.java and Position.java, add a driver to test it for n operations (make cluster/set, find, and union) and collect some data to confirm it would cost nlog*n for n operations. For C++, use Partition.java and Position.java from Java textbook as a guide to create your own code, add a driver to test it for n operations (make cluster/set, find, and union). You can ignore the validate operation if you refer to a valid position.

Question 1: Describe the ***template method pattern***.

Question 2: What are some applications for the find/union partition structures?

Extra Credit: Implement another exercise above (a total of 3 exercises).

Fill out and turn in the PA submission file for this assignment (save as PDF format)