PART 01: SortStrings

Write a program that reads strings from input and outputs them sorted, by length, shortest string first. If a subset of input strings has the same length, your program should output them in

alphabetical order.

PART 02: SwapMap

Write a method that takes a Map<String,String> as a parameter and returns a new Map<String,String> in which keys and values are swapped. Throw an exception if there are duplicate values in the map that is passed as a parameter.

PART 03: TreeMultiSet

A MultiSet is like a Set, but allows duplicates. Consider the following interface for a MultiSet:

```
public interface MultiSet<AnyType>
{
        void add( AnyType x ); boolean
        contains( AnyType x ); int count(
        AnyType x ); boolean removeOne(
        AnyType x ); boolean removeAll(
        AnyType x ); AnyType [] toArray(
        AnyType [] arr );
}
```

There are many ways to implement the MultiSet interface. A TreeMultiSet stores items in sorted order. The data representation can be a TreeMap, in which the key is an item that is in the multiset, and the value represents the number of times the item is stored. Implement the TreeMultiSet, and make sure toString is provided.