CSC 1120 – Class exercise on LLStringLog

Add the following methods to the LinkedStringLog class AND StringLogInterface. Use case-insensitive comparisons. And put in comments at the beginning of each method the Big Oh analysis of how long it will take the method to run. Then put tests into UseStringLog to test each method, not necessarily thoroughly but enough to know it works.

#48 Design and code a new method to be exported from LinkedStringLog called isEmpty with the following signature:

public boolean isEmpty()

The method returns true if the StringLog is empty and false otherwise.

#49 Design and code a new method to be exported from LinkedStringLog called howMany with the following signature:

public int howMany(String element)

The method returns a int value indicating how many times element occurs in the StringLog

#50 Design and code a new method to be exported from LinkedStringLog called uniqInsert with the following signature:

public boolean uniqInsert(String element)

The method inserts element into the StringLog unless an identical string already exists in the StringLog, in which case it has no effect on the StringLog. If it does insert the string, it returns true ; otherwise it returns false.

#51 Design and code a new method to be exported from LinkedStringLog called first with the following signature:

public String first()

The method returns the first String in the StringLog. By “first” we mean in terms of the lexicographic ordering supported by the String class’s compareToIgnoreCase method. As a precondition you should assume that StringLog is not empty.

#52 An alternative design for the LinkedStringLog class is to include an instance variable size that represents the number of Strings contained in the StringLog. What changes need to be made to the LinkedStringLog implementation?

Create a new class LinkedStringLog2 to use this approach by inheriting all that’s possible from LinkedStringLog and changing what needs to be changed. Test it.