Northern Michigan University (Marquette Co, MI)
CS222-61-21W Data Structures (Andrew A. Poe)
Practice Quiz 9
Thursday 8 April 2021 10:00 A.M. EDT
Time: 10 minutes

Given the following classes:

```
class LL {
 private:
  LLN *head;
};
```

```
class LLN {
 private:
  string data;
  LLN *next;
};
```

write the method

```
void LL::DeleteFirstAfter (string f);
```

This method removes the node containing the string coming first in alphabetical order AFTER the word *f*. If there is more than one such node, this method removes all of them. You may assume that the empty string is not in the list.

For example, if DeleteFirstAfter ("FIRE") were called on
A-->TISKET-->A-->TASKET-->A-->GREEN-->AND-->YELLOW-->BASKET
after running it would be
A-->TISKET-->A-->TASKET-->A-->AND-->YELLOW-->BASKET

since GREEN is the first word alphabetically after FIRE. There is only one copy of GREEN, but if there had been more than one, all would have been removed.

Do not use loops; use recursion only. You may write additional methods in LL and LLN if you wish. You may assume that standard constructors, destructors, accessors, and mutators have already been written. Make sure your code contains no memory leaks.

Northern Michigan University (Marquette Co, MI)
CS222-61-21W Data Structures (Andrew A. Poe)
Practice Quiz 9
Thursday 8 April 2021 10:00 A.M. EDT
Time: 10 minutes

```
void LL::DeleteFirstAfter (string f) {

 if (!head) return;
 string first = head->getfirstafter(f);
 if (first != "") head = head->DelAll (first);
}

string LLN::getfirstafter (string f) {

 string fsf = "";
 if (next) fsf = next->getfirstafter (f);
 if (data <= f) return fsf;
 if (fsf=="") return data;
 if (data < fsf) return data;
 return fsf;
}

LLN * LLN::DelAll (string s) {

 if (next) next = next->DelAll (s);
 if (data==s) {
  LLN *t = next;
  next = nullptr;
  delete this;
  return t;
 }
 return this;
}
```