CISC 3130
Test-1

Please complete the exam and submit it as a plain text email with the subject "CISC 3130 Test-1" to nzhou@brooklyn.cuny.edu by midnight on Monday, March 1.

Question 1

The following function is supposed to take a list and return a copy of the list with the order of the elements reversed. However, it doesn't work as intended. Point out the problems, and explain how to fix them.

```
public static <E> List<E> reverse(List<E> lst){
    List<E> rev;
    if (lst instanceof ArrayList){
        rev = new ArrayList<E>();
    } else {
        rev = new LinkedList<E>();
    }
    Iterator<E> it = lst.listIterator(lst.size()-1);
    while (it.hasPrevious()){
        rev.add(1, it.previous());
    }
    return rev;
}
```

Question 2

The function `assocList(lst1, lst2)` returns the association list of `lst1` and `lst2`. For example, for `lst1 = ['a, 'b', 'c']` and `lst2 = [1,2,3]`, the returned association list is `[('a', 1), ('b', 2), ('c', 3)]`. It is assumed that the two given lists have the same length. Implement the function of the specification:

```
static <U, V> List<Pair<U, V>> assocList(LinkedList<U> lst1, LinkedList<V> lst2);
```

where the class `Pair` is defined as follows:

```
class Pair<U,V>{
    public U first;
    public V second;
    public Pair(U first, V second){
        this.first = first;
        this.second = second;
    }
}
```

Question 3

The function `exclusiveOr(s1,s2)` returns the exclusive-or of set `s1` and set `s2`, which contains elements in `s1` or `s2`, but not in both. Implement the function. The actual return type `Set` should be the same as the actual type of the parameter `Set`.

```
static <E> Set<E> exclusiveOr(Set<E> s1, Set<E> s2)
```

Question 4

The function `countOccfRoots` takes a text represented as a string, and a dictionary that maps words to their roots, and returns a map that tells the number of times each word occurs in the text, treating each word the same as its root. Words are separated by spaces, and words that do not occur in the dictionary are ignored.

```
static Map<String, Integer>  countOccRoots(String text, Map<String, String> dic);
```

For example, if the dictionary maps verbs *see*, *sees*, *seeing*, *saw*, and *seen* to their root form *see*, then the the word `see` occurs 8 times in the following text:

```
The past tense of see is saw. The third-person singular simple present indicative form of se
The present participle of see is seeing. The past participle of see is seen.
```