Lab 9: Iterator

```
Location: CourseWeb -> Labs/Recitations -> Lab 8: Gnome Sort
Download the following files:
  1. SListIterator.java
  2. Iterator.java
  2. SListIteratorTester.java
```

Introduction

Suppose we have a list named `list`. This is an ADT List using **linked list** to store a collection of data. To make it simple for this lab, this ADT List consist of only two methods; (1) `addToFirst(T newEntry)` to add `newEntry` into the first position, and (2) `getEntry(int position)` to retrieve an entry at `position`. As explained in class, if we want to access every entry in the list sequentially from the first position (position 1) to the last position, we need to use some type of loop and use the method `getEntry()` as shown below:

```
for(int i = 1; i <= numberOfData; i++)
{
        getEntryArray[i - 1] = list.getEntry(i);
}
```

Recall that the method `getEntry()` needs to call the method `getNodeAt(int position)` to obtain the reference to the `Node` associated with the `position`. Since we have only the reference to the first node of the link chain, the method `getNodeAt()` needs to traverse the link change starting at the first node every time it is called.

As explained in class, we can improve the performance of sequentially accessing all data using iterator. An iterator allows us to access the next data without traversing the link chain from the first node. To access all data using iterator, a user can use a loop as shown below:

```
int index = 0;

while(iterator.hasNext())
{
        iteratorArray[index] = iterator.next();
        index++;
}
```

What to Do?

For this lab, you are going to implement an iterator for this simply ADT List implementation in `SListIterator.java`. What you need to do are as follows:

1. Implement the method `getIterator()` which allows a user to obtain the iterator of this list

2. Implement the class `IteratorForSList` which implements the interface `Iterator`. For this class, you have to implements methods `hasNext()` and `next()`. The method `remove()` is given.

Test Class

The test class `SListIteratorTester.java` is given. This test class will create a list named `list`. Add random numbers into this list. Then retrieve all data sequentially and store them in the integer array named `getEntryArray` according to its position using the method `getEntry()` (see the first code fragment in the introduction section). Then it will use iterator and access all data sequentially and store them in the integer array named `iteratorArray` according to its position using iterator (see the second code fragment in the introduction section). Then these two array are compared to ensure the correctness of your iterator implementation. This test class also time both `getEntry()` and iterator to show how much performance is improved using iterator.

Due Date and Submission

For the due date, please check the lab in the CourseWeb. Submit your `SListIterator.java` to the CourseWeb under this lab by the due date. **No late submission will be accepted**.