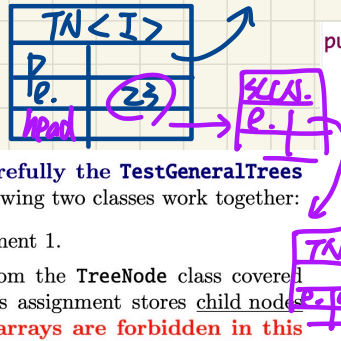
Review Q & A - Mar. 21

Programming Test 2

Assignment 3 Solution

Generic Classes



This assignment requires the more intermediate use of generics. **Study carefully the TestGeneralTrees class given to you (in the tests package)**, which contains how the following two classes work together:

- SLLNode<E>: this class is essentially the Node class you used in Assignment 1.
- TreeNode<E>: this class is similar to but should be distinguished from the TreeNode class covered in Lecture W8. The version of TreeNode class you are given for this assignment stores child nodes as **a chain of singly-linked nodes**. **Remember that primitive arrays are forbidden in this assignment.**
- In the TestGeneralTrees class, pay attention to the following type declarations:
  - `TreeNode<String> n;` declares a tree node storing some string value: we write `n.getElement()` to retrieve the string value. In this case, the generic parameter E declared in the TreeNode class (i.e., TreeNode<E>) is instantiated by String.
  - `TreeNode<Integer> n;` declares a tree node storing some integer value: we write `n.getElement()` to retrieve the integer value. In this case, the generic parameter E declared in the TreeNode class (i.e., TreeNode<E>) is instantiated by Integer.
  - `SLLNode<TreeNode<String>> tn;` declares a singly-linked node storing the reference of some tree node (which in turn stores some string value): we write `tn.getElement()` to retrieve the tree node (of type TreeNode<String>) and write `tn.getElement().getElement()` to retrieve the stored string value. In this case, the generic parameter E declared in the SLLNode class (i.e., SLLNode<E>) is instantiated by TreeNode<String> (which in tern instantiates the generic parameter E in the TreeNode class by String).
  - `SLLNode<TreeNode<Integer>> tn;` declares a singly-linked node storing the reference of some tree node (which in turn stores some integer value): we write `tn.getElement()` to retrieve the tree node (of type TreeNode<Integer>) and write `tn.getElement().getElement()` to retrieve the stored integer value. In this case, the generic parameter E declared in the SLLNode class (i.e., SLLNode<E>) is instantiated by TreeNode<Integer> (which in tern instantiates the generic parameter E in the TreeNode class by Integer).

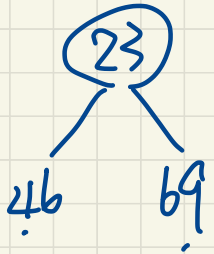
```
public class TreeNode<E> {
    private E element; /* data object */
    private TreeNode<E> parent; /* unique parent
    private SLLNode<TreeNode<E>> headOfChildList
    private SLLNode<TreeNode<E>> tailOfChildList

    public TreeNode(E element) {

    public E getElement() {

    public void setElement(E element) {

    public TreeNode<E> getParent() {

    public void setParent(TreeNode<E> parent) {

    public SLLNode<TreeNode<E>> getChildren() {

    public void addChild(TreeNode<E> child) {
}
```

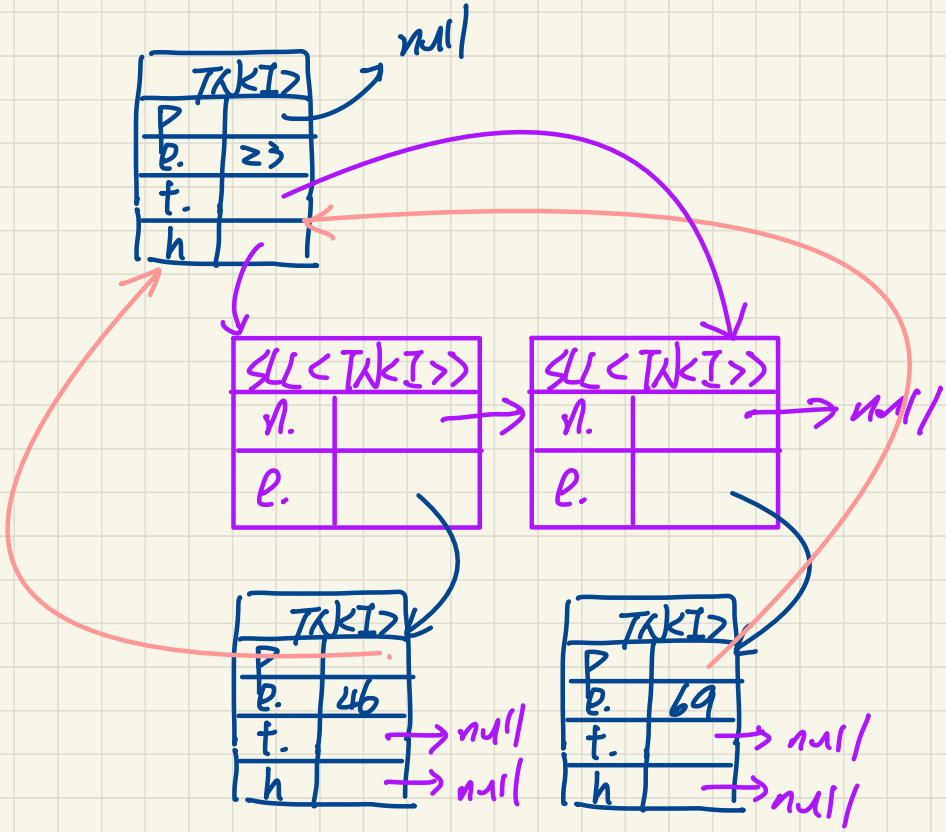
head.getE. → TN<E>

head.getE().getE()

TN<E>

```
public class SLLNode<E> {
    private E element;
    private SLLNode<E> next;

    public SLLNode(E e, SLLNode<E> n) {

    public E getElement() {

    public SLLNode<E> getNext() {

    public void setNext(SLLNode<E> n) {

    public void setElement(E e) {
}
```





A3 starter

↳ tests

↳ TestsTrees.java

Task 1 Rank

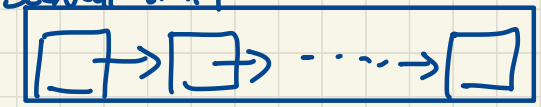
task1(TN n, int i, int j)

(1) n

(2) n

(a) pre-order or post-order traversal.

SLNode chain



(b) Sort the traversal result

↳ e.g. insertion sort or selection sort

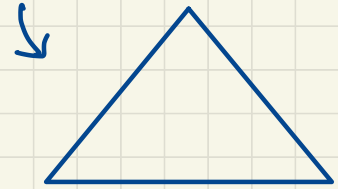
(c) extract nodes from indices i to j.

(a) traverse, and when each node is added, insert it s.t. the result chain of nodes remains sorted.

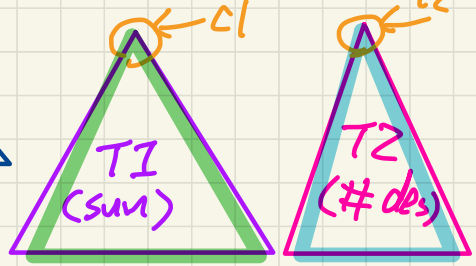
Task 2: Stats (sum of values, # des.)

(1) n

input →



(2) n



output