Φροντιστηριακή άσκηση #2-Β

Θέμα: Γενικές κλάσεις (generics), εσωτερικές κλάσεις και διαπροσπελαστές (iterators)

Η φροντιστηριακή αυτή άσκηση αποσκοπεί στην εξοικείωση με τις γενικές κλάσεις, τις εσωτερικές κλάσεις και τους διαπροσπελαστές. Η κλάση MyLLSet<E> υλοποιεί ένα σύνολο αντικειμένων τύπου E κάνοντας χρήση μιας απλά συνδεδεμένης λίστας. Στο σύνολο αυτό δεν θα επιτρέπονται περισσότερες από μια εμφανίσεις του ιδίου στοιχείου, συνεπώς η πράξη της προσθήκης αντικειμένων θα πρέπει να ελέγχει εάν το στοιχείο που προσθέτουμε ανήκει ήδη στο σύνολο. Η ισότητα δυο στοιχείων της λίστας καθορίζεται μέσω της μεθόδου equals() της κλάσης Object (η οποία μπορεί να έχει επανορισθεί). Περαιτέρω, το σύνολο θα εφοδιαστεί και με άλλες λειτουργικές πράξεις, όπως αυτές της διαγραφής, του ελέγχου εάν είναι κενό κ.ο.κ. Επίσης, θα υλοποιήσουμε και έναν Iterator, ο οποίος θα εξασφαλίζει γρήγορη και εύκολη διαπέραση όλων των στοιχείων του συνόλου.

1. Χρησιμοποιώντας το BlueJ, δημιουργήστε ένα έργο με την ονομασία “inner_generic_class_iterator_demo”.

2. Δημιουργήστε την κλάση MyLLSet<E>, η οποία θα υλοποιεί το σύνολο μέσω μιας απλά συνδεδεμένης λίστας.

3. Δημιουργήστε σε αυτήν την εσωτερική και **private** κλάση LLNode<E>. Η δομή της κλάσης MyLLSet<E> ύστερα από την προσθήκη της εσωτερικής κλάσης LLNode<E> δίνεται παρακάτω.

```
public class MyLLSet<E>
{
    . . .


    /** Εσωτερική κλάση LLNode<E> */
    private class LLNode<E>
    {
        . . .
    }

    . . .
}
```

4. Ενα αντικείμενο της κλάσης LLNode<E> αντιστοιχεί σε έναν κόμβο της λίστας. Τα πεδία που περιέχει είναι τα next και element. Ο τύπος των πεδίων αυτών είναι LLNode<E> και E, αντίστοιχα. Όλα τα πεδία είναι private.

`private E element`	Τα δεδομένα του κόμβου.
`private LLNode<E> next`	Ο επόμενος κόμβος

5. Ο κατασκευαστής (Constructor) αυτής της κλάσης θα δημιουργεί νέους κόμβου χωρίς δεδομένα οι οποίοι δεν θα έχουν επόμενο κελί.

`public LLNode()`	Κατασκευαστής. Δημιουργεί ένα νέο κόμβο χωρίς δεδομένα και επόμενο κόμβο.

6. Να υλοποιηθούν οι παρακάτω μέθοδοι:

`void setElement(E element)`	Θέτει τα δεδομένα του κόμβου.
`void setNext(LLNode<E> next)`	Θέτει τον επόμενο κόμβο
`E getElement()`	Επιστρέφει τα δεδομένα ενός κόμβου
`LLNode<E> getNext()`	Επιστρέφει το επόμενο κόμβο

7. Εν συνεχεία μπορούμε να συνεχίσουμε με την υλοποίηση της κλάσης MyLLSet<E>, η οποία περιέχει τα πεδία head και size. Ο τύπος των πεδίων αυτών είναι LLNode<E> και int, αντίστοιχα. Όλα τα πεδία είναι private.

`private LLNode<E> head`	Αναφορά στο πρώτο αντικείμενο του συνόλου (κεφαλή της λίστας).

private int size	Το μέγεθος του συνόλου.

8. Ο κατασκευαστής (Constructor) αυτής της κλάσης θα δημιουργεί ένα κενό σύνολο (ουσιαστικά μια νέα λίστα χωρίς κανένα κελί).

public MyLLSet()	Κατασκευάζει ένα κενό σύνολο.

9. Να υλοποιηθούν οι παρακάτω μέθοδοι:

boolean isEmpty()	Ελέγχει αν το σύνολο είναι κενό.
int size()	Επιστρέφει το πλήθος των στοιχείων του συνόλου.
boolean isIn(E e)	Ελέγχει αν το στοιχείο e ανήκει στο σύνολο
void add(E e)	Προσθέτει ένα στοιχείο στο σύνολο .
void remove(E e)	Διαγράφει ένα στοιχείο του συνόλου. Υποθέτει ότι το στοιχείο ανήκει στο σύνολο.

10. Στην κλάση MyLLSet<E> δημιουργείστε την **εσωτερική** και **private** κλάση MyLLIterator, η οποία θα υλοποιεί την διαπροσωπεία (interface) java.util.Iterator<E>. Η κλάση αυτή περιέχει μονάχα ένα private πεδίο nextNode τύπου LLNode<E>.

private LLNode<E> nextNode	Αναφορά σε έναν κόμβο του συνόλου.

11. Ο κατασκευαστής (Constructor) αυτής της κλάσης θα δημιουργεί έναν νέο Iterator για το σύνολο με αναφορά στον πρώτο κόμβο της λίστας, δηλ. τον κόμβο head.

public MyLLIterator()	Κατασκευάζει έναν νέο Iterator.

12. Να υλοποιηθούν οι παρακάτω μέθοδοι:

boolean hasNext()	Ελέγχει εάν υπάρχει επόμενο στοιχείο.
E getNext()	Επιστρέφει το επόμενο στοιχείο.
void remove()	Δεν εκτελεί καμία πράξη.

Μια υλοποίηση αυτής της κλάσης δίνεται παρακάτω:

```
private class MyLLIterator implements Iterator<E>
{
    /** Αναφορά σε κάποιο κόμβο */
    private LLNode<E> nextNode;

    /*
     * Δημιουργεί ένα νέο αντικείμενο τύπου MyLLIterator.
     */
    public MyLLIterator()
    {
        nextNode = head;
    }

    /**
      * Ελέγχει εάν υπάρχει επόμενο στοιχείο.
      */
    public boolean hasNext()
    {
        return (nextNode!=null);
    }

    /**
      * Επιστρέφει το επόμενο στοιχείο
      */
    public E next()
    {
        E element = nextNode.getElement();
        nextNode = nextNode.getNext();
        return element;
```

```
    }

    /**
      * Δεν υλοποιείται.
      */
    public void remove(){}
}//MyLLIterator
```

13. Στην κλάση MyLLSet να προσθέσετε την παρακάτω μέθοδο:

`Iterator<E> iterator()`	Επιστρέφει έναν Iterator για το σύνολο με αναφορά στο πρώτο του στοιχείο.

14. Να υλοποιηθεί η κλάση Test_String η οποία (μέσω της μεθόδου της main()) δημιουργεί ένα σύνολο, προσθέτει 10 αντικείμενα τύπου String με τιμές String-0, String-1,...,String-9 και κατόπιν (με χρήση ενός iterator) τα τυπώνει. Στην συνεχεία διαγράφει τα «άρτια» αντικείμενα (String-0, String-2,...), και εκτυπώνει (πάλι με την χρήση ενός iterator) τα εναπομείναντα στοιχεία.
15. Να δημιουργηθεί η κλάση Person η οποία εχει δύο πεδία. Το πρώτο αναφέρεται στο πλήρες όνομα του ατόμου (τύπου String) και το δεύτερο στο φύλο του (True εάν άνδρας, False εάν γυναίκα). Τα αντικείμενα τύπου Person έχουν έναν κατασκευαστή που λαμβάνει το πλήρες όνομα και το φύλο ως παραμέτρους, και υποστηρίζουν τις μεθόδους isMale() και toString().
16. Να υλοποιηθεί η κλάση Test_Person η οποία (μέσω της μεθόδου της main()) δημιουργεί ένα σύνολο, προσθέτει 10 αντικείμενα τύπου Person με τιμές {Person-0, Άνδρας}, {Person-1, Γυναίκα}, ..., {Person-8, Άνδρας}, (Person-9, Γυναίκα) και κατόπιν (με χρήση ενός iterator) τα τυπώνει. Στην συνεχεία διαγράφει τους άνδρες και εκτυπώνει (πάλι με την χρήση ενός iterator) τα εναπομείναντα στοιχεία.