CSCI 2120:
Software Design & Development II

UNIT 2: Collections Framework & Generics

LinkedList Examples

Overview

Introduction

In this lecture, we will see different kinds of linked list example programs in Java.

If you have any doubt related to the basics of linked list, I recommend you check the LinkedList lecture.

How to add Element to LinkedList?

Let’s take an example program where we will add elements in different ways using add(), add(int index, Object o), addFirst(Object o), addLast(Object o), addAll() methods of list interface and deque interface. Follow all steps in the coding.

Example 1: Add Elements to LinkedList

```java
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Vector;
public class LinkedListTester1 {
   public static void main(String[] args) {
       // Create a LinkedList object.
       LinkedList list = new LinkedList(); // empty linked list.

       // Call add() method for adding Heterogeneous elements using reference variable list.
       list.add("One");
       list.add(2);
       list.add(null); // null elements are allowed in the linked list.
       list.add("Four");
       System.out.println("Initial LinkedList order: " +list);

       // Adding an element specified in the linked list.
       list.add(3, "Three");

       // Adding an element at the first position of list using addFirst() method of Deque interface.
       System.out.println("LinkedList Elements after adding the first element");
       list.addFirst("Zero");
       System.out.println(list);

       // Adding an element at the end of the list using addLast() method of Deque interface.
       System.out.println("LinkedList Elements after adding the last element");
       list.addLast("Five");
       System.out.println(list);

       // Adding all elements from existing ArrayList collection to the end of the LinkedList. Create an ArrayList object.
       ArrayList al = new ArrayList();
       al.add("Six");
       al.add(7);
       al.add("Eight");

       // Call addAll() method to add all elements to the end of the linked list.
       list.addAll(al);
       System.out.println("LinkedList Elements after adding all elements from ArrayList");
       System.out.println(list);

       // Adding all elements from an existing Vector collection at the specified position of LinkedList.
       Vector v = new Vector();
       v.add(7.5);
       v.add(7.8);
       list.addAll(9, v);
       System.out.println("Linkedlist elements after adding all elements from vector");
       System.out.println(list);
   }
}
```

Example 1: Add Elements to LinkedList

Output:

```
Initial LinkedList order: [One, 2, null, Four]
LinkedList Elements after adding the first element
[Zero, One, 2, null, Three, Four]
LinkedList Elements after adding the last element
[Zero, One, 2, null, Three, Four, Five]
LinkedList Elements after adding all elements from ArrayList
[Zero, One, 2, null, Three, Four, Five, Six, 7, Eight]
Linkedlist elements after adding all elements from vector
[Zero, One, 2, null, Three, Four, Five, Six, 7, 7.5, 7.8, Eight]
```

How to remove Element from LinkedList?

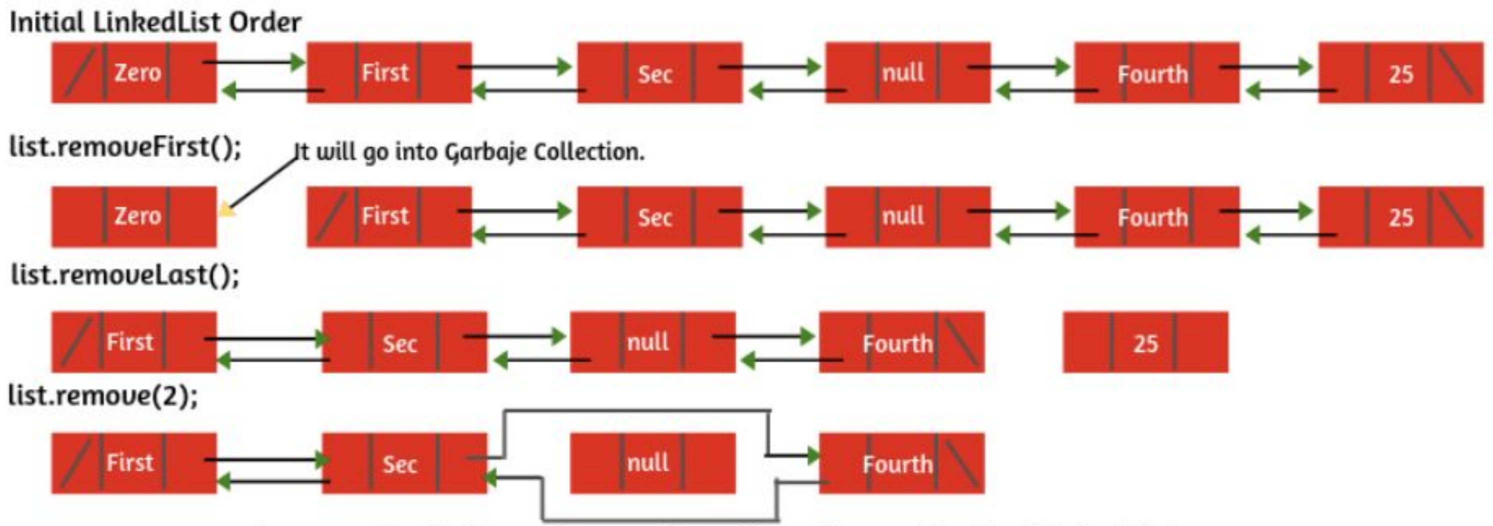
Let’s take an example program and see how to remove elements from the list in different ways using methods such as remove(), removeFirst(), removeLast().

Example 2: Remove Elements from LinkedList

```java
import java.util.ArrayList;
import java.util.LinkedList;
public class LinkedListTester2 {
   public static void main(String[] args) {
       // Create a generic LinkedList object of type String.
       LinkedList<String> list = new LinkedList<String>();
       int size = list.size();
       System.out.println("Size of Linkedlist: " +size);

       // Adding elements of String type.
       list.add("Zero");
       list.add("First");
       list.add("Second");
       list.add(null); // null elements are allowed in the linked list.
       list.add("Fourth");
       list.add("25");
       System.out.println("Initial LinkedList order: " +list);

       // Removing the first element from list using removeFirst() method.
       list.removeFirst();
       System.out.println("LinkedList Elements after removing the first element");
       System.out.println(list);

       // Removing the last element from the list using removeLast() method.
       System.out.println("LinkedList Elements after removing the last element");
       list.removeLast();
       System.out.println(list);

       // Removing element at the specified position from the list.
       list.remove(2);
       System.out.println("LinkedList Elements after removing the element at index position 2 ");
       System.out.println(list);

       // Adding all elements from existing an ArrayList collection to the end of LinkedList. Creating a generic ArrayList object of String type.
       ArrayList<String> al = new ArrayList<String>();
       al.add("Third");
       al.add("Fourth");

       list.addAll(2, al);
       list.removeLastOccurrence("Fourth");

       System.out.println("LinkedList Elements after removing last occurrence");
       System.out.println(list);
   }
}
```

Example 2: Remove Elements from LinkedList

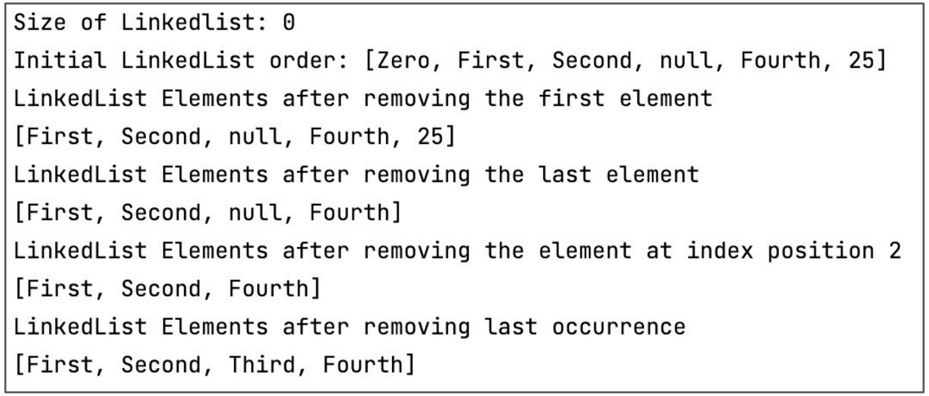
A conceptual view of removing an element in LinkedList is shown in the below figure.



A conceptual view of removing of an element in the LinkedList

Example 2: Remove Elements from LinkedList

Output:



```
Size of Linkedlist: 0
Initial LinkedList order: [Zero, First, Second, null, Fourth, 25]
LinkedList Elements after removing the first element
[First, Second, null, Fourth, 25]
LinkedList Elements after removing the last element
[First, Second, null, Fourth]
LinkedList Elements after removing the element at index position 2
[First, Second, Fourth]
LinkedList Elements after removing last occurrence
[First, Second, Third, Fourth]
```

How to get Element from LinkedList in Java?

Let’s create a program where we will get an element from the list in different ways using get(), getFirst(), and getLast() method.

Example 3: Get Elements from LinkedList

```java
import java.util.LinkedList;
public class LinkedListTester3 {
   public static void main(String[] args) {
       // Create a LinkedList object.
       LinkedList<Integer> list = new LinkedList<Integer>();
       // Adding even numbers from 0 to 20 as elements in the list.
       for(int i = 0 ; i <= 20; i++) {
           if(i % 2 == 0) // It will check even number.
               list.add(i);
       }
       // Call getFirst() method to get first even number.
       Object firstEvenNumber = list.getFirst(); // Return type of getFirst() methods is an Object.
       System.out.println("First even number: " +firstEvenNumber);
       Object lastEvenNumber = list.getLast();
       System.out.println("Last even number: " +lastEvenNumber);
       Object getElement = list.get(5);
       System.out.println("Even number at index 5: " +getElement);
   }
}
```

Example 3: Get Elements from LinkedList

Output:

```
First even number: 0
Last even number: 20
Even number at index 5: 10
```

How to retrieve and remove Element in LinkedList?

We can retrieve and remove an element from LinkedList using peekFirst(), peekLast(), pollFirst(), and pollLast() methods.

1. peekFirst(): This method retrieves the first element from the list but does not remove the element from the list.

2. peekLast(): This method retrieves the last element from the list but does not remove it.

3. pollFirst(): This method retrieves and removes the first element from the list.

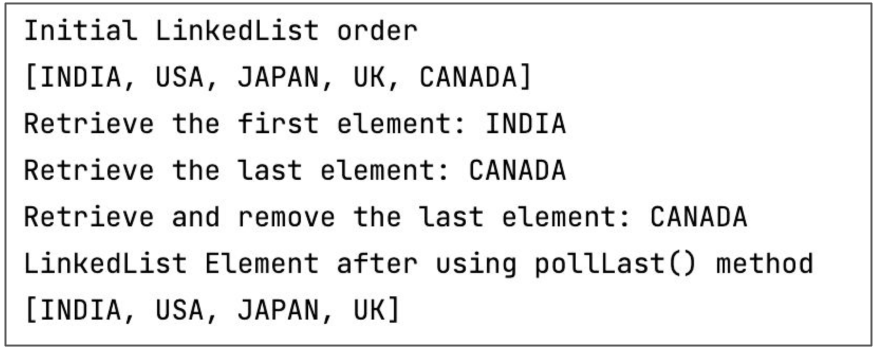
4. pollLast(): This method retrieves and removes the last element from the list.

Example 4: Retrieve & Remove Elements in LinkedList

```java
import java.util.LinkedList;
public class LinkedListTester4 {
   public static void main(String[] args) {
       // Create a LinkedList object.
       LinkedList<String> list = new LinkedList<String>();

       // Adding elements to the list.
       list.add("INDIA");
       list.add("USA");
       list.add("JAPAN");
       list.add("UK");
       list.add("CANADA");
       System.out.println("Initial LinkedList order");
       System.out.println(list);

       // Call peek() method to retrieve the first element from list.
       Object firstElement = list.peekFirst(); // Return type of this method is an Object.
       System.out.println("Retrieve the first element: " +firstElement);
       Object lastElement = list.peekLast();
       System.out.println("Retrieve the last element: " +lastElement);

       // Call pollLast() to retrieve and remove the last element from the list.
       Object element1 = list.pollLast();
       System.out.println("Retrieve and remove the last element: " +element1);
       System.out.println("LinkedList Element after using pollLast() method");
       System.out.println(list);
   }
}
```

Example 4: Retrieve & Remove Elements in LinkedList

Output:



```
Initial LinkedList order
[INDIA, USA, JAPAN, UK, CANADA]
Retrieve the first element: INDIA
Retrieve the last element: CANADA
Retrieve and remove the last element: CANADA
LinkedList Element after using pollLast() method
[INDIA, USA, JAPAN, UK]
```

Pop and Push Element from and onto Stack in LinkedList

Pop() method: This method is equivalent to removeFirst() method that removes and returns the first element from the list. In other words, It pops an element from the stack represented by the list.

Push() method: This method is equivalent to addFirst() method that adds an element at the first position of the list. In other words, It pushes an element onto the stack represented by the list.

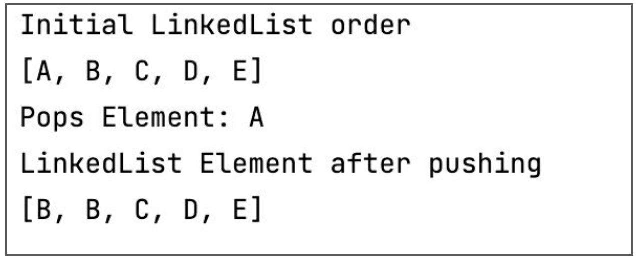
Let’s make a example program where we will pop and push an element from and onto stack in LinkedList.

Example 5: Pop & Push Elements on Stack in LinkedList

```java
import java.util.LinkedList;
public class LinkedListTester5 {
   public static void main(String[] args) {
       LinkedList<Character> list = new LinkedList<Character>();

       // Adding elements in the list.
       list.add('A');
       list.add('B');
       list.add('C');
       list.add('D');
       list.add('E');

       System.out.println("Initial LinkedList order");
       System.out.println(list);
       Object element = list.pop(); // Return type of this method is an Object.

       System.out.println("Pops Element: " +element);
       list.push('B');
       System.out.println("LinkedList Element after pushing");
       System.out.println(list);
   }
}
```

Example 5: Pop & Push Elements on Stack in LinkedList

Output:



```
Initial LinkedList order
[A, B, C, D, E]
Pops Element: A
LinkedList Element after pushing
[B, B, C, D, E]
```

END