**King Fahd University of Petroleum & Minerals**

**College of Computer Science and Engineering**

**Information and Computer Science Department**

**ICS 202 – Data Structures**

# Lab02 SinglyLinked lists

**Objectives**

The objective of this lab is to design, implement and use Linked Lists

**Outcomes**

After completing this Lab, students are expected to:

* Understand the structure of a singly linked list class
* Add methods and enhance the functionality of a linked list class
* Use linked lists class in an application

**Note:**

* **For Task02 and Task03 you must only use the SLL class provided in this Lab session. Using a different class will result in a grade of zero for these two tasks.**
* **Each student is required to do the lab tasks alone. Copying from another student or from the internet will result is a grade of zero.**

**Lab Tasks**

**Task 01**

Study the **JavaSinglyLinkedList.docx** document and then modify the given program **JavaLinkedListDriver** by adding the required Java code:

**// Add code that will insert "Tabouk" before "Dammam", display error message if  
// "Dammam" is not in list or if the list is empty. Your code must work for any   
// two strings str1 and str2.  
   
   
 // Add code that will insert "AlKhafj" before the last node, your code must work  
 // for any non-empty list. Display an error message if the list is empty.**

**Sample output:**

List: [Madinah, Dammam, Riyadh, Dhahraan]  
List: [Najraan, Madinah, Dammam, Riyadh, Jubail, Dhahraan]  
Removed Element: Najraan  
Updated list: [Taif, Madinah, Dammam, Riyadh, Jubail, Abha]  
List after adding Tabouk: [Taif, Madinah, Tabouk, Dammam, Riyadh, Jubail, Abha]  
List after adding AlKhafj: [Taif, Madinah, Tabouk, Dammam, Riyadh, Jubail, AlKhafj, Abha]

**Task02**

Assume that the nodes of a singly linked list of size **n** nodes are indexed starting at **0** to **n – 1**, where **0** is the index of the first node. Add the following methods to the class **SLL<T>.java**:

(a) **public void insertBefore(int index, T newElem)** that inserts an element **newElem** before the node with the given index. Your method must throw an appropriate exception if the list is empty or if index is not valid.

For example, suppose your linked list is: [ 7 5 4 50 9 ],

Then a call to **insertBefore(3, 20)** will result in the following linked list: [ 7 5 4 **20** 50 9 ]

A call to **insertBefore(0, 30)** will result in the following linked list: [**30** 7 5 4 50 9 ]

A call to **insertBefore(5, 30)** will generate an exception because 5 is an invalid index.

Note: Your method must be general and it must not use any existing SLL methods except size() and

isEmpty().

(b) **public T delete(int index)**  thatdeletes the node with the given index. Your method must throw

an appropriate exception if the list is empty or if index is not valid.

Note: Your method must be general and it must not use any existing SLL methods except size and

is Empty().

(c) **public void insertAfterSecondOccurrence(T e1, T e2)** that will insert a new node with data

**e1** after the second occurrence of a node with data **e2**. Your method must throw appropriate

exceptions if the list is empty or if the list has no second occurrence of **e2**.

For example, suppose your linked list is: [ 9 12 5 4 12 9 ],

Then a call to **insertAfterSecondOccurrence (3, 12)** will result in the following linked list:

[ 9 12 5 4 12 **3** 9 ]

a call to **insertAfterSecondOccurrence (7, 9)** will result in the following linked list:

[ 9 12 5 4 12 9 **7**]

a call to **insertAfterSecondOccurrence (7, 5)** will generate an exception because there is no second

occurrence of 5.

Note: Your method must be general and it must not use any existing SLL methods.

Task03

Write a **test class** to test each one of the additional methods that were required to be implemented in task 02.

**Sample program output:**

Original Integer array: [ 7 5 3 50 7 9 ]

After inserting 20 before index 4: [ 7 5 3 50 20 7 9 ]

Element deleted from index 4: 20

After deleting element from index 4: [ 7 5 3 50 7 9 ]

After inserting 30 after the second occurence of 7: [ 7 5 3 50 7 30 9 ]