Tutorial 5

Introduction

In this tutorial, students are asked to implement two forms of list data structure: ArrayList and Singly Linked List. The ADT of ArrayList and Singly Linked List are illustrated in the figures below:

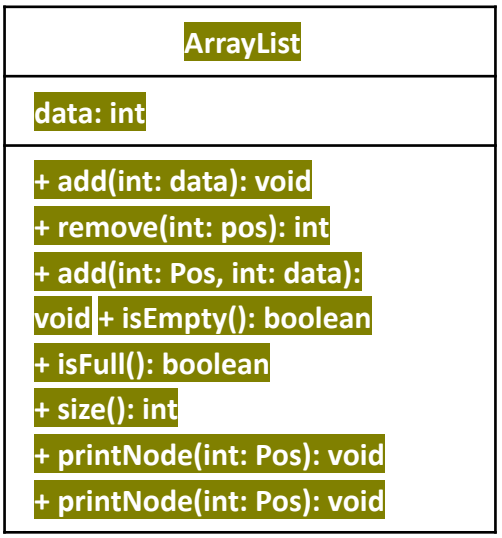

Figure 1 ArrayList ADT

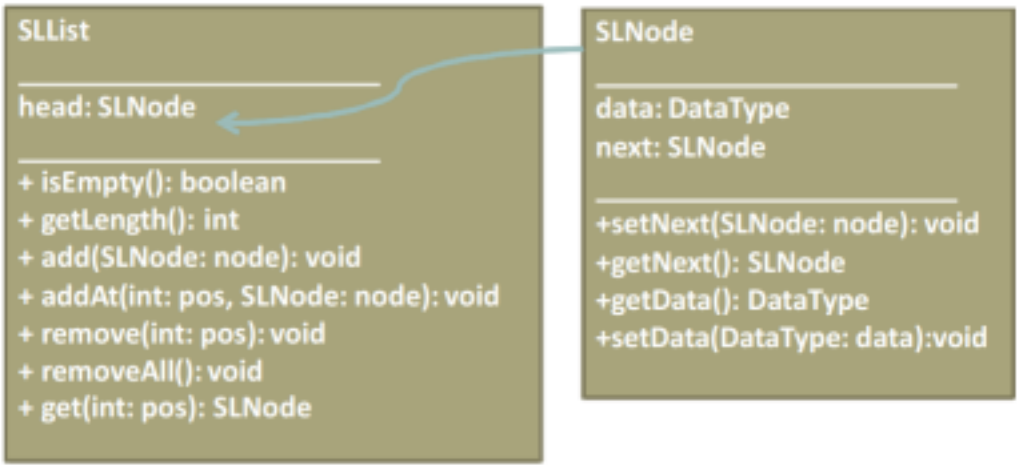

Figure 2 Singly Linked List ADT

The ArrayList ADT represents a list of operations that is implemented using an array. The list of following operators are supported:

· add(int data): Add a new element to the end of the list

· remove(int Pos): Remove an element at the position Pos from the list ·
add(int Pos, int data): Add a new element into the list at the position Pos ·
isEmpty(): Check if the list is empty
· isFull(): Check if the list is full (suppose that the maximum length of the list is 100) ·
size(): Return the current length of the list
· printNode(int Pos): Show data of the node at the position Pos
· printList(): Show data of all nodes in the list

Example 01 demonstrates the implementation of the ArrayList ADT above (without the remove operator).

The Singly Linked List ADT represents a list of strings that is implemented as a linked list. Every node (SLNode) of this list contains two parts: data (a string) and next (reference variable points to the next node in the list). The following operators are supported:

Node operations:

· setNext(SLNode n): set the next point to node n
· getNext(): return the next node after the current node
· getData(): return the node's data (a string)
· print(): show data of the node

List operations:

· add(SLNode n): Add a new node n into the beginning of the list
· remove(int Pos): Remove node at the position Pos from the list
· addAt(int Pos, SLNode n): Add a new node n into the list at the position Pos ·
isEmpty(): Check if the list is empty
· size(): Return the current length of the list
· print (): Show data of all nodes in the list
· search(String data): search for a string data in the list. If found, the operator will return the position of the node that containt the string, otherwise the operators will return -1.

Example 02 demonstrates the implementation of the Singly Linked List above (without the search operator).
In the exercise section, students are asked to implement more operators for the list data structure and use list to slove a related problem.

Examples

1. Example 01 – ArrayList implementation

Student should create classes that implements the ArrayList mentioned above. In this

example two classes should be created:

- Class ArrayList implements the ArrayList ADT
- Class ArrayListApp which containts the main function, demonstrates how to use the implemented ArrayList

Please refer to class ***ArrayList*** and ***ArrayListApp*** in the Tutorial example code.

2. Example 02 – Singly Linked List implementation

Student should create classes that implements the Singly Linked List mentioned above. In this example three classes should be created:

- Class SLNode implements the Node of the list in the Singly Linked List ADT
- Class SLList implements the list it self
- Class SLListApp demonstrates how to use the implemented Linked List. This calss containts the main function.

Please refer to class ***SLNode***, ***SLList*** and ***SLListApp*** in the Tutorial example code.

Exercises

1. Exercise 1

The operator remove(int Pos) is missing in the implementation of the ArrayList ADT. Please add the Java code for the remove() operator.

2. Exercise 2

The operator search(String data) is missing in the implementation of the Singly Linked List ADT. Please add the Java code for the search() operator.

3. Exercise 3

Please create a Singly Linked List data structure and add necessary operators to solve the Polynomial problem described in the lecture section.