Data Structure and Algorithm

Training Assignments

Document Code	**25e-BM/HR/HDCV/FSOFT**
Version	**1.1**
Effective Date	**20/02/2021**

Hanoi, 02/2021

RECORD OF CHANGES

No	Effective Date	Change Description	Reason	Reviewer	Approver
1.	20/02/2021	Create a new Assignment 1	Create new	DieuNT1	VinhNV

Contents

CODE:	DSA_Assignment2_Opt01
TYPE:	N/A
LOC:	N/A
DURATION:	180 MINUTES

Assignment 2: Linked List

Objectives:

- Understand the basics of LinkedList?
- Distinguish characteristics of LinkedList types (Singly LinkedList, Doubly Linked List, Circular Linked List)
- Able to use Singly LinkedList to solve basic algorithms.

Assignment Descriptions:

Problem 1: What is wrong with the following declaration?

```
class Element { double value; Element link; } ;
```

Problem 2: Suppose that a linked list is made up of nodes of type

```
class List{
  static Node head;
static class Node {
   int key;
   Node next;
};
```

that you are given a pointer "list" of type link, which points to the first node of the list; and that the last node has NULL as its link.

a) Write a code fragment to delete the second node of the list.
Assume that there are at least two nodes on the list.

b) Write a code fragment to add a new node with key 17 just after the second node of the list. Assume that there are at least two nodes on the list.

c) Write an iterative function count() that takes a link as input, and prints out the number of elements in the linked list.

d) Write an iterative function max() that takes a link as input, and returns the value of the maximum key in the linked list.

Assume all keys are positive, and return -1 if the list is empty.

Problem 3: What does the following program print out?

```
class LinkedList {
    static Node head;
    static class Node {
        int data;
        Node next;
        Node(int d)
        {
            data = d;
            next = null;
        }
    }
    Node function(Node node)
    {
        Node prev = null;
        Node current = node;
        Node next = null;
        while (current != null) {
            next = current.next;
            current.next = prev;
            prev = current;
            current = next;
            System.out.println(current + ” ”);
        }
        node = prev;
        return node;
    }
```

-- THE END --