Assignment2

● Graded

Student

IBRAHIM ALSHAYEA

Total Points

94 / 100 pts

Question 1

(no title) **44** / 50 pts

- **– 0 pts** Excellent Work
- **– 50 pts** Missing Code
- **– 30 pts** Doubly Linked Lists are not used
- **– 20 pts** Add function is missing
- **– 20 pts** Delete function is missing
- **– 45 pts** Code does not compile
- **– 40 pts** Code does not run (Error)
- **– 15 pts** Extending existing classes
- **– 6 pts** Nodes are not ordered
- ✔ **– 6 pts** Repeated values are allowed
- **– 10 pts** Delete does not work properly
- **– 10 pts** Add does not work properly
- **– 5 pts** Code is not general for all classes
- **– 5 pts** Error when deleting from empty list
- **– 5 pts** Throws an error instead of returning false
- **– 2 pts** Minor error in the code
- **– 50 pts** Cheating case
- **– 5 pts** Mistake in variables

Question 2

(no title) **50** / 50 pts

+ 0 pts Wrong

The behavior of the program

- ✔ **+ 3 pts** The code compiles but does not run
- ✔ **+ 2 pts** The code compiles and runs
- ✔ **+ 3 pts** Convert to binary
- ✔ **+ 3 pts** convert to radix 3
- ✔ **+ 3 pts** convert to radix 4
- ✔ **+ 3 pts** convert to radix 5
- ✔ **+ 3 pts** convert to radix 6
- ✔ **+ 3 pts** convert to radix 7
- ✔ **+ 3 pts** convert to radix octal
- ✔ **+ 3 pts** convert to radix 9
- ✔ **+ 3 pts** convert to decimal
- ✔ **+ 3 pts** convert to radix 11
- ✔ **+ 3 pts** convert to radix 12
- ✔ **+ 3 pts** convert to radix 13
- ✔ **+ 3 pts** convert to radix 14
- ✔ **+ 3 pts** convert to radix 15
- ✔ **+ 3 pts** convert to hexadecimal

– 5 pts No exception handling

– 30 pts Not using a stack

– 5 pts Non-generic code (If statement for each radix)

– 2 pts Minor error in the code (Hardcoding, bad practices)

– 10 pts Error in the submission

Q1
50 Points

Develop and implement a new class ODLList for an Ordered Doubly Linked List (without extending any existing classes). The nodes in this list are generic, comparable, **distinct**, and **ordered**.
Your class should have the following:
1. $boolean\ add(T\ element)$--adds $element$ in its proper position if it does not exist.
2. $boolean\ delete(T\ element)$--deletes $element$ from the list if it exists.
Both methods should return true if the element is added/deleted, otherwise return false.

Submit only the file **ODLList.java**

ODLList.java Download

```java
package Hw;

import LabPrograms.DLLNode;

public class ODLList<T extends Comparable<T>> {
    private Node<T> head;
    private Node<T> tail;
    static class Node<T>{
        private T info;
        private Node<T> next, prev;
        Node(T info){
            this.info = info;
        }
    }

    public boolean add(T element){
        Node<T> newNode = new Node<>(element);
        if(head == null){
            head = newNode;
            tail = newNode;
            return true;
        }
        Node<T> current = head;
        while (current != null && current.info.compareTo(element) < 0) {
            current = current.next;
        }
        if(current == null){
            tail.next = newNode;
            tail.next.prev = tail;
            tail =newNode;
        }
        else if (current.prev == null) {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;

        }
        else {
            newNode.prev = current.prev;
            newNode.next = current;
            current.prev.next = newNode;
            current.prev = newNode;
        }
        return true;

    }
    public boolean delete(T element){
        Node<T> current = head;
        while (current != null && current.info.compareTo(element) != 0) {
            current = current.next;
        }
        if(current == null){
            return false;
```

```
54        }
55        if(current.prev == null){
56            head = head.next;
57            if (head != null) {
58                head.prev = null;
59            } else {
60                tail = null;
61            }
62        }
63        else if (current.next == null) {
64            tail = tail.prev;
65            tail.next = null;
66        }
67        else {
68            current.prev.next = current.next;
69            current.next.prev = current.prev;
70        }
71        return true;
72    }
73
74    public void printAll() { // I've implemented this method just to test my program.
75        for (Node<T> tmp = head; tmp != null; tmp = tmp.next)
76            System.out.print(tmp.info + " ");
77        System.out.println();
78    }
79
80 }
81
```

Q2

50 Points

Write a program to convert a number from decimal notation to a number expressed in a number system whose base (or radix) is a number between 2 and 16. The conversion is performed by repetitious division by the base to which a number is being converted and then taking the remainders of division in the reverse order **using a stack**. For example, in converting to binary, number 6 requires three such divisions: 6/2 = 3 remainder 0, 3/2 = 1 remainder 1, and finally, 1/2 = 0 remainder 1. The remainders 0, 1, and 1 are put in the reverse order so that the binary equivalent of 6 is equal to 110.

Number systems with bases greater than 10 require more symbols. Therefore, use capital letters. For example, a hexadecimal system requires 16 digits: 0, 1, . . ., 9, A, B, C,D, E, F. In this system, decimal number 26 is equal to 1A in hexadecimal notation, because 26/16 = 1 remainder 10 (that is, A), and 1/16 = 0 remainder 1.

Submit only the file **Assignment2Q2.java**

Assignment2Q2.java Download

```java
package Hw;
import java.util.Scanner;
import java.util.Stack;

public class Assignment2Q2 {

    public static String decimalToBase(int n, int base) {
        if (base < 2 || base > 16) {
            throw new IllegalArgumentException("Base must be between 2 and 16");
        }

        Stack<Character> stack = new Stack<>();
        while (n > 0) {
            int remainder = n % base;
            char digit;
            if (remainder < 10) {
                digit = (char) ('0' + remainder);
            } else {
                digit = (char) ('A' + remainder - 10);
            }
            stack.push(digit);
            n /= base;
        }

        String result = "";
        while (!stack.isEmpty()) {
            result += stack.pop();
        }
        return result;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a decimal number: ");
        int decimalNumber = scanner.nextInt();
        System.out.print("Enter the base to convert to (between 2 and 16): ");
        int base = scanner.nextInt();

        String convertedNumber = decimalToBase(decimalNumber, base);
        System.out.println("The decimal number " + decimalNumber + " is equal to " +
convertedNumber +
                " in base " + base + " notation.");
    }
}

```