CSC248 – Fundamentals of Data Structure
Academic Session October 2023 – February 2024
Lab Assignment 4 – Linked List (UDT)

Course Outcomes (CO)	LO1	LO2	LO3
CO1			
CO2	√	√	√
CO3			

The list using dynamic storage to store computer's information for a computer laboratory. Given the respective classes as follows:

```
public class Computer
{
 private int serialNo; //computer identification
private String brand; //brand name  private int
year        //year of buying   private double price
      //buying price

 //Normal constructor
   //Getter

}  public class
ListNode{  private
Object obj;
 private ListNode next;
         :
         :
}

public class List
{
 private ListNode firstNode;  //reference to the first node in the list
private ListNode lastNode;    //reference to the last node in the list
private ListNode currNode;    //to traversal purpose
         :
         :

   publicList();
   public void insertAtFront(Object);
   public void insertAtBack(Object); public
   void insertAtAtMiddle(Object); public
   Object remove(int);
 public void searchComputer(int);    public
   int countComputer(double);
}
```

a) Write all definition functions for the above operation to do the following tasks:

   i. To insert a new node (computer's information) at the front/back/middle of list. The information is given by a parameter. If the information existed in the list, you don't have to insert the node. (NOTE** Every computer has a unique serial number identification)

   ii. To remove a node from the list based on the serial number of the computer. Computer serial number is given by a parameter.

   iii. To print the output of computer's information based on the searching index (the serial number). Computer serial number is given by a parameter.

   iv. To count and return the number of computers which exceed a certain amount price. The amount is given by a parameter. This method also will print the output of brand code and year of buying which computers fulfill the above criteria.

b) Write an application program by implementing **a menu selection** to do the following tasks.

   i. Insert a new node into list. The computer to be inserted can be at the front, at the back and at the middle of the list based on the user selection.

   ii. To delete any node from a list based on serial number of the computer

   iii. To print the output of computer's information based on the searching index

   iv. To count and return the number of computers which exceed a certain amount price

Computer.java

```java
public class Computer {
    private int serialNo;
    private String brand;
    private int year;
    private double price;

    public Computer(int serialNo, String brand, int year, double price) {
        this.serialNo = serialNo;
        this.brand = brand;
        this.year = year;
        this.price = price;
    }

    public int getSerialNo() {
        return this.serialNo;
    }

    public void setSerialNo(int serialNo) {
        this.serialNo = serialNo;
    }

    public String getBrand() {
        return this.brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public int getYear() {
        return this.year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public double getPrice() {
        return this.price;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public String toString() {
        return "Serial number: " + this.serialNo + "\nBrand: " + this.brand + 
"\nYear: " + this.year + "\nPrice: "
                + this.price;
    }

}
```

List.java

```java
public class List {
    private ListNode firstNode;
    private ListNode lastNode;
    private ListNode currNode;

    public List() {
        this.firstNode = null;
        this.lastNode = null;
        this.currNode = null;
    }

    public void insertAtFront(Object obj) {
        ListNode newNode = new ListNode(obj, this.firstNode);
        this.firstNode = newNode;
        if (this.lastNode == null) {
            this.lastNode = newNode;
        }
    }

    public void insertAtBack(Object obj) {
        ListNode newNode = new ListNode(obj, null);
        if (this.lastNode == null) {
            this.firstNode = newNode;
            this.lastNode = newNode;
        } else {
            this.lastNode.setNext(newNode);
            this.lastNode = newNode;
        }
    }

    public void insertAtMiddle(Object obj) {
        ListNode newNode = new ListNode(obj, null);
        if (this.firstNode == null) {
            this.firstNode = newNode;
            this.lastNode = newNode;
        } else {
            int count = 0;
            ListNode curr = this.firstNode;
            while (curr != null) {
                count++;
                curr = curr.getNext();
            }
            int middle = count / 2;
            curr = this.firstNode;
            for (int i = 0; i < middle; i++) {
                curr = curr.getNext();
            }
            newNode.setNext(curr.getNext());
            curr.setNext(newNode);
        }
        // if (this.firstNode == null) {
        // this.insertAtFront(obj);
        // } else if (this.firstNode.getNext() == null) {
        // this.insertAtBack(obj);
        // } else {
        // ListNode newNode = new ListNode(obj, null);
        // ListNode curr = this.firstNode;
        // ListNode prev = null;
        // while (curr != null) {
        // if (((Computer) curr.getObj()).getYear() > ((Computer)
        // newNode.getObj()).getYear()) {
        // break;
        // }
        // prev = curr;
        // curr = curr.getNext();
        // }
        // if (prev == null) {
        // newNode.setNext(this.firstNode);
        // this.firstNode = newNode;
        // } else {
        // newNode.setNext(curr);
        // prev.setNext(newNode);
        // }
        // }
    }

    public Object remove(int serialNo) {
        ListNode curr = this.firstNode;
        ListNode prev = null;
        while (curr != null) {
            if (((Computer) curr.getObj()).getSerialNo() == serialNo) {
                break;
            }
            prev = curr;
            curr = curr.getNext();
        }
        if (curr == null) {
            return null;
        }
        if (prev == null) {
            this.firstNode = curr.getNext();
        } else {
            prev.setNext(curr.getNext());
        }
        if (curr.getNext() == null) {
            this.lastNode = prev;
        }
        return curr.getObj();
    }

    public void searchComputer(int serialNo) {
        ListNode curr = this.firstNode;
        while (curr != null) {
            if (((Computer) curr.getObj()).getSerialNo() == serialNo) {
                break;
            }
            curr = curr.getNext();
        }
        if (curr == null) {
            System.out.println("Computer not found");
        } else {
            System.out.println(curr.getObj());
        }
    }

    public int countComputer(double price) {
        int count = 0;
        ListNode curr = this.firstNode;
        while (curr != null) {
            if (((Computer) curr.getObj()).getPrice() > price) {
                count++;
                // print out the computer
                System.out.println(curr.getObj() + "\n");
            }
            curr = curr.getNext();
        }
        return count;
    }

    public void print() {
        ListNode curr = this.firstNode;
        while (curr != null) {
            System.out.println(curr.getObj() + "\n");
            curr = curr.getNext();
        }
    }
}
```

ListNode.java

```java
public class ListNode {
    private Object obj;
    private ListNode next;

    public ListNode(Object obj, ListNode next) {
        this.obj = obj;
        this.next = next;
    }

    public Object getObj() {
        return this.obj;
    }

    public void setObj(Object obj) {
        this.obj = obj;
    }

    public ListNode getNext() {
        return this.next;
    }

    public void setNext(ListNode next) {
        this.next = next;
    }

    public String toString() {
        return this.obj.toString();
    }
}
```

Main.java

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner strInput = new Scanner(System.in);
        Scanner intInput = new Scanner(System.in);

        System.out.print(
                "1. Insert a new node into list\n2. Delete note from list based
on serial number\n3. Print output of computer's information\n4. Count and return
the number of computers which exceed a certain amount price\n5. Display all
lists\n6. Exit\n\nEnter your choice: ");
        int choice = intInput.nextInt();

        List list = new List();

        System.out.println();
        while (choice != 6) {
            if (choice == 1) {
                System.out.print("Enter serial number: ");
                int serialNo = intInput.nextInt();
                System.out.print("Enter brand: ");
                String brand = strInput.nextLine();
                System.out.print("Enter year: ");
                int year = intInput.nextInt();
                System.out.print("Enter price (RM): ");
                double price = intInput.nextDouble();

                System.out.print(
                        "\n1. Insert at the beginning of the list\n2. Insert at
the end of the list\n3. Insert at middle of the list\n\nEnter your choice: ");
                int choice2 = intInput.nextInt();

                if (choice2 == 1) {
                    list.insertAtFront(new Computer(serialNo, brand, year,
price));
                } else if (choice2 == 2) {
                    list.insertAtBack(new Computer(serialNo, brand, year,
price));
                } else if (choice2 == 3) {
                    list.insertAtMiddle(new Computer(serialNo, brand, year,
price));
                } else {
                    System.out.println("Invalid choice");
                }
            } else if (choice == 2) {
                System.out.print("Enter serial number: ");
                int serialNo = intInput.nextInt();
                list.remove(serialNo);
            } else if (choice == 3) {
                System.out.print("Enter serial number: ");
                int serialNo = intInput.nextInt();
                System.out.println();
                list.searchComputer(serialNo);
            } else if (choice == 4) {
                System.out.print("Enter price to print out which computers price
exceed it (RM): ");
                double price = intInput.nextDouble();

                System.out.println("\nThere's " + list.countComputer(price) + "
computers which exceed RM" + price);
            } else if (choice == 5) {
                list.print();
            } else {
                System.out.println("Invalid choice");
            }
            System.out.print(
                    "\n1. Insert a new node into list\n2. Delete note from list
based on serial number\n3. Print output of computer's information\n4. Count and
return the number of computers which exceed a certain amount price\n5. Display
all lists\n6. Exit\n\nEnter your choice: ");
            choice = intInput.nextInt();
            System.out.println();
        }

        strInput.close();
        intInput.close();

        System.out.println("Program terminating...");
    }
}
```