UNIVERSITI TEKNOLOGI MARA

CSC248
FUNDAMENTALS OF DATA STRUCTURE

LAB ASSIGNMENT 6

NAME : MUHAMMAD REDZA BIN MAHAYADIN

STUDENT ID : 2022676696

GROUP : RCDCS1103B

LECTURER : SIR MOHD NIZAM BIN OSMAN

QUESTION 1

```java
import java.util.*;

public class Q1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        Scanner in1 = new Scanner(System.in);

        Queue qHouse = new Queue();
        Queue qSemi_D = new Queue();
        Queue qTerrace = new Queue();
        Queue temporary = new Queue();

        for (int i = 0; i < 2; i++) {
            System.out.print("1. Semi-D¥n2. Terrace¥nEnter house type: ");
            int typeInt = in.nextInt();

            String type = "";
            if (typeInt == 1)
                type = "Semi-D";
            else if (typeInt == 2)
                type = "Terrace";
            else
                System.out.println("Invalid input.");

            System.out.print("Enter location: ");
            String location = in1.nextLine();
            System.out.print("Enter size (Metre): ");
            double size = in.nextDouble();
            System.out.print("Enter price per unit (RM): ");
            double price = in.nextDouble();

            qHouse.enqueue(new House(type, location, size, price));
            System.out.println();
        }

        while (!qHouse.isEmpty()) {
            House house = (House) qHouse.dequeue();

            if (house.getType().equals("Semi-D")) {
                qSemi_D.enqueue(house);
            } else if (house.getType().equals("Terrace")) {
                qTerrace.enqueue(house);
            }
```

```
            temporary.enqueue(house);
        }

        while (!temporary.isEmpty()) {
            qHouse.enqueue(temporary.dequeue());
        }

        int countTerrace = 0;
        while (!qTerrace.isEmpty()) {
            House house = (House) qTerrace.dequeue();

            if (house.getPrice() < 150000) {
                countTerrace++;
                if (countTerrace == 1)
                    System.out.println("Houses with price less than RM
150,000.00: ");
                System.out.println(house);
            }
        }

        if (countTerrace == 0)
            System.out.println("No houses with price less than RM
150,000.00.");

        while (!temporary.isEmpty()) {
            qTerrace.enqueue(temporary.dequeue());
        }

        int count = 0;
        while (!qHouse.isEmpty()) {
            House house = (House) qHouse.dequeue();

            if (house.getPrice() > 300000) {
                count++;
                if (count == 1)
                    System.out.println("Houses with price more than RM
300,000.00: ");
                System.out.println(house);
            }
        }

        if (count == 0)
            System.out.println("No houses with price more than RM
300,000.00.");

        while (!temporary.isEmpty()) {
```

```
            qHouse.enqueue(temporary.dequeue());
        }
        System.out.println("Number of houses with price more than RM
300,000.00: " + count);

        in1.close();
        in.close();
    }
}

class Queue extends LinkedList<Object> {
    protected LinkedList<Object> list;

    public Queue() {
        list = new LinkedList<Object>();
    }

    public void enqueue(Object element) {
        list.addFirst(element);
    }

    public Object dequeue() {
        return list.removeLast();
    }

    public boolean isEmpty() {
        return list.isEmpty();
    }
}

class House {
    private String type;
    private String location;
    private double size;
    private double price;

    public House(String type, String location, double size, double price)
{
        this.type = type;
        this.location = location;
        this.size = size;
        this.price = price;
    }

    public String getType() {
        return type;
```

```
    }

    public String getLocation() {
        return location;
    }

    public double getSize() {
        return size;
    }

    public double getPrice() {
        return price;
    }

    @Override
    public String toString() {
        return "House type: " + type + "¥nLocation: " + location + "¥nSize
(Metre): " + String.format("%,.2f", size)
                + "¥nPrice: RM " + String.format("%,.2f", price) + "¥n";
    }
}
```

QUESTION 2

```
import java.util.*;

public class Q2 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        Scanner in1 = new Scanner(System.in);

        Queue qCustomer = new Queue();
        Queue qQualify = new Queue();

        for (int i = 0; i < 10; i++) {
            System.out.print("Enter customer name: ");
            String name = in.nextLine();
            System.out.print("Enter account number: ");
            int accountNo = in1.nextInt();
            System.out.print("Enter saving (RM): ");
            double saving = in1.nextDouble();
            System.out.print("Enter total transaction (RM): ");
            double totalTransaction = in1.nextDouble();

            Customer customer = new Customer(name, accountNo, saving,
totalTransaction);

            qCustomer.enqueue(customer);
            if (customer.process()) {
                qQualify.enqueue(customer);
            }
        }

        while (!qQualify.isEmpty()) {
            System.out.println(qQualify.dequeue() + "¥n");
        }

        in1.close();
        in.close();
    }
}

class Customer {
    private String name;
    private int accountNo;
    private double saving;
    private double totalTransaction;
```

```
    public Customer(String name, int accountNo, double saving, double
totalTransaction) {
        this.name = name;
        this.accountNo = accountNo;
        this.saving = saving;
        this.totalTransaction = totalTransaction;
    }

    public String getName() {
        return name;
    }

    public int getAccountNo() {
        return accountNo;
    }

    public double getSaving() {
        return saving;
    }

    public double getTotalTransaction() {
        return totalTransaction;
    }

    @Override
    public String toString() {
        return "Customer name: " + name + "¥nAccount No: " + accountNo +
"¥nSaving: RM "
                + String.format("%,.2f", saving)
                + "¥nTotal Transaction: RM " + String.format("%,.2f",
totalTransaction);
    }

    public boolean process() {
        return saving > 1000;
    }
}

class Node {
    Object data;
    Node link;

    public Node(Object elem) {
        this.data = elem;
        this.link = null;
    }
```

```
    public Node(Object elem, Node nextElem) {
        this.data = elem;
        this.link = nextElem;
    }

    public Object getData() {
        return data;
    }

    public Node getLink() {
        return link;
    }
}

class ListNode {
    Node first;
    Node last;

    public ListNode() {
        this.first = null;
        this.last = null;
    }
}

class Queue extends ListNode {
    public Queue() {
        super();
    }

    public void enqueue(Object elem) {
        Node newNode = new Node(elem);
        if (this.first == null) {
            this.first = newNode;
            this.last = newNode;
        } else {
            this.last.link = newNode;
            this.last = newNode;
        }
    }

    public Object dequeue() {
        if (this.first != null) {
            Object data = this.first.data;
            this.first = this.first.link;
            return data;
```

```
        }
        return null;
    }

    public boolean isEmpty() {
        return this.first == null;
    }

    public Object getFirst() {
        if (this.first != null) {
            return this.first.data;
        } else {
            return null;
        }
    }

    public Object getNext() {
        if (this.first != null && this.first.link != null) {
            return this.first.link.data;
        } else {
            return null;
        }
    }

    public Object getLast() {
        if (this.last != null) {
            return this.last.data;
        } else {
            return null;
        }
    }
}
```