LinkedList Demo

Node class

```
package com.citi.linklist;

public class Node {

    int data;
    Node next;
}
```

LinkedList class→

```
package com.citi.linklist;

public class LinkedList {
    Node head;
    //insert
    public void insert(int data) {
        Node node=new Node();
        node.data=data;
        node.next=null;

        if(head==null) {
            head=node;
        }else {
            Node n=head;
            while(n.next!=null) {
                n=n.next;
            }
```

```
            n.next=node;
        }
    }

    public void insertAtStart(int data)
    {
        Node node = new Node();
        node.data = data;
        node.next = null;
        node.next = head;
        head = node;
    }

    public void insertAt(int index,int data)
    {
        Node node = new Node();
        node.data = data;
        node.next = null;

        if(index==0)
        {
            insertAtStart(data);
        }
        else{
        Node n = head;
        for(int i=0;i<index-1;i++)
        {
            n = n.next;
        }
        node.next = n.next;
        n.next = node;
        }
    }
    public void deleteAt(int index)
```

```
    {
        if(index==0)
        {
            head = head.next;
        }
        else
        {
            Node n = head;
            Node n1 = null;
            for(int i=0;i<index-1;i++)
            {
                n = n.next;
            }
            n1 = n.next;
            n.next = n1.next;
            //System.out.println("n1 " + n1.data);
            n1 = null;
        }
    }

    public void show()
    {
        Node node = head;

        while(node.next!=null)
        {
            System.out.println(node.data);
            node = node.next;
        }
        System.out.println(node.data);
    }

}
```

Runner class

```
-------------------------
package com.citi.linklist;

public class Runner {

    public static void main(String[] args) {

        LinkedList list = new LinkedList();
        list.insert(18);
        list.insert(45);
        list.insert(12);
        list.insertAtStart(25);
        list.insertAt(0, 55);
        list.deleteAt(2);
        list.show();
    }

}
-----------------------
```

Stack Demo

Stack class→

```
package com.citi.stack;

public class Stack {

    int stack[] = new int[5];
    int top=0;
    public void push(int data) {
        stack[top] =data;
        top++;
    }

    public int pop() {
        int data;
        top--;
        data =stack[top];
        stack[top]=0;
        return data;
    }

    public int peek() {
        int data;
        data =stack[top-1];
        return data;
    }

    public void show() {
        for(int n:stack) {
            System.out.println(n+ " ");
        }
```

```
    }
}
--------------------
package com.citi.stack;

public class Runner {

    public static void main(String[] args) {
        Stack stack = new Stack();
        stack.push(15);
        stack.push(24);
        stack.push(90);

        System.out.println("Peek "+stack.peek());

        System.out.println("Pop "+stack.pop());

        stack.show();

    }

}
--------------------
```

Queue Demo

Queue class→

```
package com.citi.queue;

public class Queue {

    int queue[]=new int[5];
    int size;
    int front;
    int rear;

    public void enQueue(int data) {
        queue[rear]=data;
        rear = rear+1;
        size=size+1;
    }

    public int deQueue() {
        int data = queue[front];
        front = front+1;
        size=size-1;
        return data;

    }
    public void show() {
        System.out.print("Elements :");
        for(int i=0;i<size;i++) {
            System.out.print(queue[front+i]+" ");
        }
    }
```

```
}
```

```
package com.citi.queue;

public class Runner {

    public static void main(String[] args) {
        Queue queue = new Queue();
        queue.enQueue(12);
        queue.enQueue(21);
        queue.enQueue(40);
        queue.enQueue(11);

        queue.deQueue();
        queue.deQueue();

        queue.show();

    }

}
```
