**Tutorial: Linked\_List**

**Question1: Solution**

public class ListNode {

public char value;

public ListNode next;

public ListNode (char v)

{

this.value=v;

}

}

public class LinkedList {

private ListNode head=null;

public void print(){

ListNode i = head;

while (i !=null)

{

System.out.println (i.value);

i=i.next;

}

}

public void add(char v, int index){

ListNode newNode = new ListNode (v); // create node.... we need to define constructor in ListNode

//newNode.value=v;

if (index==0)

{newNode.next=head;

head = newNode;

}

else {

ListNode i = head;

for(int count = 0; count <index-1; count++)

{

i= i.next;

}

newNode.next = i.next;

i.next = newNode;

}

}

public void remove(int index){

if (index==0){

head = head.next;

}

else {

ListNode i = head;

for (int count = 0; count <index-1; count++)

{

i = i.next;

}

ListNode j = i.next;

i.next = j.next;

}

}

public void isEmpty ()

{

if (head != null)

{

System.out.println ("LinkedList is not empty");

}

else System.out.println ("LinkedList is empty");

}

}

public class ListTest {

public static void main(String[] args) {

LinkedList L = new LinkedList();

L.add ('a', 0); //a

L.add ('b', 1); //a,b

L.add ('c', 2); //a,b,c

L.add ('h', 1); //a,h,b,c

L.add ('x', 0); // x,a,h,b,c

L.add ('z', 5); //x,a,h,b,c,z

L.remove (1); // x,h,b,c,z

L.remove (3); //x,h,b,z

L.remove (3); //x,h,b

L.remove (0); //h,b

L.remove (1); //h

L.remove (0);

L.isEmpty();

L.print(); //Linked\_List is empty

}

}

**Question 2 Solution**

**public class** Student {

**private** String id;

**private** String name;

**private double** mark;

**public** Student (String id, String name, **double** mark)

{

**this**.id=id;

**this**.name=name;

**this**.mark= mark;

}

**public** String getId() {

**return** id;

}

**public void** setId(String id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public void** setName(String name) {

**this**.name = name;

}

**public double** getMark() {

**return** mark;

}

**public void** setMark(**double** mark) {

**this**.mark = mark;

}

**public void** display(){

System.***out***.println("Id :" + id );

System.***out***.println("Name :" + name);

}

}

**/////////////////////////////////////////////////////**

**public class** Node {

**private** Student data;

**private** Node next;

**public** Node(){

**this**.data=**null**;

**this**.next=**null**;

}

**public** Node(Student data){

**this**.data=data;

**this**.next=**null**;

}

**public** Node(Student data, Node next){

**this**.data=data;

**this**.next=next;

}

**public** Student getData() {

**return** data;

}

**public void** setData(Student data) {

**this**.data = data;

}

**public** Node getNext() {

**return** next;

}

**public void** setNext(Node next) {

**this**.next = next;

}

}

**////////////////////////////////////////////////**

**public class** LinkedList {

**private** Node head;

**public** LinkedList() {

head = **null**;

}

**public boolean** isEmpty (){

**return** head==**null**;

}

**public void** insertAtFront (Student data) {

Node newNode = **new** Node(data);

newNode.setNext(head);

head = newNode;

}

**public void** insertAtBack (Student data) {

Node newNode = **new** Node(data);

**if**(isEmpty()) {

head = newNode;

}

**else** {

Node curr = head;

**while** (curr.getNext()!=**null**){

curr = curr.getNext();

}

curr.setNext(newNode);

}

}

**public void** removeFromFront (){

**if** ( !isEmpty()){

head= head.getNext();

}

}

**public** Student removeFromFront2(){

Student data = **null**;

**if**(!isEmpty()){

data = head.getData();

head = head.getNext();

}

**return** data;

}

**public void** removeFromBack(){

**if** (!isEmpty()){

**if**(head.getNext()==**null**){

head = **null**;

}

**else**

{

Node curr = head;

Node prev = **null**;

**while** (curr.getNext() !=**null**){

prev = curr;

curr = curr.getNext();

}

prev.setNext(**null**);

}

}

}

**public** Student removeFromBack2(){

Student data=**null**;

**if**(!isEmpty()){

**if**(head.getNext()==**null**){

data = head.getData();

head = **null**;

}

**else** {

Node curr = head;

Node prev = **null**;

**while**(curr.getNext()!=**null**){

prev = curr;

curr = curr.getNext();

}

data = curr.getData();

prev.setNext(**null**);

}

}

**return** data;

}

**public int** size(){

**int** count = 0;

Node curr = head;

**while** (curr !=**null**){

count++;

curr=curr.getNext();

}

**return** count;

}

**public void** display(){

Node curr = head;

**while** (curr!=**null**){

curr.getData().display();

curr= curr.getNext();

}

}

**public boolean** searchById(String id){

Node curr= head;

**while**(curr !=**null**){

**if** (curr.getData().getId().equals(id)){

**return true**;

}

curr=curr.getNext();

}

**return false**;

}

}