**6. Detailed code with relevant commands:->**

package Singlell;

import java.util.Scanner;

public class single {

// Node is created

public class node {

int data;

node next;

node(int data) {

this.data = data;

this.next = null;

}

}

public static node head = null;

public static int pos;

public static int size;

// Insert the element at first

public void insertAtBegin(int data) {

node temp = head;

node newnode = new node(data);

if (head == null) {

head = temp = newnode;

} else {

newnode.next = head;

head = newnode;

}

size++;

display();

}

// Insert the element at end

public void insertAtEnd(int data) {

node newnode = new node(data);

node temp = head;

if (head == null) {

temp = head = newnode;

} else {

while (temp.next != null) {

temp = temp.next;

}

temp.next = newnode;

temp = temp.next;

}

size++;

}

// Insert the element at perticular position

public void insertAtPos(int data, int pos1) {

node newnode = new node(data);

node temp = head;

int i = 1;

if (pos1 == 1) {

newnode.next = head;

head = temp = newnode;

} else {

if (pos1 <= size) {

for (i = 1; i < pos1 - 1; i++) {

temp = temp.next;

}

newnode.next = temp.next;

temp.next = newnode;

} else {

System.out.println("postion is more than size of linked list ");

}

}

size++;

display();

}

// delete the element which is present at begin

public void deleteFromBegin() {

node temp = head;

if (head == null) System.out.println("ll is empty");

else {

head = temp = temp.next;

size--;

}

display();

}

// delete the element which is present at end

public void deleteFromEnd() {

node temp = head;

node prevnode = head;

if (size == 1) {

head = null;

size--;

} else if (head == null) {

System.out.println("ll is emptyyy");

} else {

while (temp.next != null) {

prevnode = temp;

temp = temp.next;

}

prevnode.next = null;

size--;

}

display();

}

// delete the element which is present at perticular position

public void deleteFromPos(int pos1) {

if (pos1 <= size) {

node prevnode = head;

node temp = head;

if (pos1 == 1) {

head = head.next;

size--;

} else {

for (int i = 1; i < pos1; i++) {

prevnode = temp;

temp = temp.next;

}

prevnode.next = temp.next;

size--;

}

} else {

System.out.println("Invalid position");

}

display();

}

// printing elements if linked list with the help of temp

public void display() {

node temp = head;

System.out.println("linked list is ");

while (temp != null) {

System.out.print(temp.data + "->");

temp = temp.next;

}

System.out.println("Null");

}

// main function

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

single ll = new single();

int info;

int ch;

// for while loop

while (true) {

// List of choise

System.out.println("1.Insert at beginning");

System.out.println("2.Insert at particular position");

System.out.println("3.Insert at end");

System.out.println("4.Delete from begin");

System.out.println("5.Delete from perticular position");

System.out.println("6.Delete from end");

System.out.println("7.Print the linked list");

System.out.println("0.Exit");

System.out.println("Enter youre choise");

ch = sc.nextInt();

switch (ch) {

case 1:

// function call for insert at begin

System.out.println("Enter the number that you want to add at begining");

info = sc.nextInt();

ll.insertAtBegin(info);

System.out.println("Node is added at beginning");

break;

case 2:

// function call for insert at particular position

System.out.println("Enter the number that you want to add at perticular position");

info = sc.nextInt();

System.out.println("Enter position");

pos = sc.nextInt();

ll.insertAtPos(info, pos);

System.out.println("Node is added");

break;

case 3:

// function call for insert at end

System.out.println("Enter the number that you want to add at End");

info = sc.nextInt();

ll.insertAtEnd(info);

System.out.println("Node is added at end");

ll.display();

break;

case 4:

// function call for delete from begin

System.out.println("first node ie is head is deleted");

ll.deleteFromBegin();

break;

case 5:

// function call for delete from perticular position

System.out.println("Enter position");

pos = sc.nextInt();

System.out.println("Node is deleted");

ll.deleteFromPos(pos);

break;

case 6:

// function call for delete from end

ll.deleteFromEnd();

System.out.println("Last node is deleted");

break;

case 7:

// function call for displaying elements of linked list

ll.display();

break;

case 0:

System.exit(0);

break;

} } }

}

**7.Results :->**

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

2

Enter the number that you want to add at perticular position

1

Enter position

1

linked list is

1->10->Null

Node is added

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

1

Enter the number that you want to add at begining

10

linked list is

10->Null

Node is added at beginning

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

4

first node ie is head is deleted

linked list is

10->9->Null

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

3

Enter the number that you want to add at End

9

Node is added at end

linked list is

1->10->9->Null

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

6

linked list is

Null

Last node is deleted

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

5

Enter position

2

Node is deleted

linked list is

10->Null

1.Insert at beginning

2.Insert at particular position

3.Insert at end

4.Delete from begin

5.Delete from perticular position

6.Delete from end

7.Print the linked list

0.Exit

Enter youre choise

7

linked list is

25->14->Null