**DAY 3 ASSIGNMENT**

**Que 1:**

Write a function “insert\_any()” for inserting a node at any given position of the linked list. Assume position starts at 0.

**Function:**

// function to insert a Node at any given/required position

static Node insert\_any(Node headNode, int position, int data)

{

Node head = headNode;

if (position < 1)

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

// if position is 1 then new node is

// set infornt of head node

// head node is changing.

if (position == 1)

{

Node newNode = new Node(data);

newNode.nextNode = headNode;

head = newNode;

} else {

while (position-- != 0) {

if (position == 1) {

// adding Node at any given/required position

Node newNode = GetNode(data);

// Making the new Node to point to

// the old Node at the same position

newNode.nextNode = headNode.nextNode;

// Replacing current with new Node

// to the old Node to point to the new Node

headNode.nextNode = newNode;

break;

}

headNode = headNode.nextNode;

}

if (position != 1)

System.out.print("Position out of range");

}

return head;

}

static void PrintList(Node node) {

while (node != null) {

System.out.print(node.data);

node = node.nextNode;

if (node != null)

System.out.print(",");

}

System.out.println();

}

**QUE 2:**

Write a function “delete\_beg()” for deleting a node from the beginning of the linked list.

**Function:**

void delete\_beg()

{

struct node \*ptr;

if(head == NULL)

{

printf("\nList is empty");

}

else

{

ptr = head;

head = ptr->next;

free(ptr);

printf("\n Beginning node is deleted..");

}

}

**QUE 3:**

Write a function “delete\_end()” for deleting a node from the end of the linked list.

**Function:**

void delete\_end()

{

struct node \*toDelete, \*secondLastNode;

if(head == NULL)

{

printf("List is already empty.");

}

else

{

toDelete = head;

secondLastNode = head;

/\* Traverse to the last node of the list \*/

while(toDelete->next != NULL)

{

secondLastNode = toDelete;

toDelete = toDelete->next;

}

if(toDelete == head)

{

head = NULL;

}

else

{

/\* Disconnect link of second last node with last node \*/

secondLastNode->next = NULL;

}

/\* Delete the last node \*/

free(toDelete);

printf("SUCCESSFULLY DELETED LAST NODE OF LIST\n");

}

}