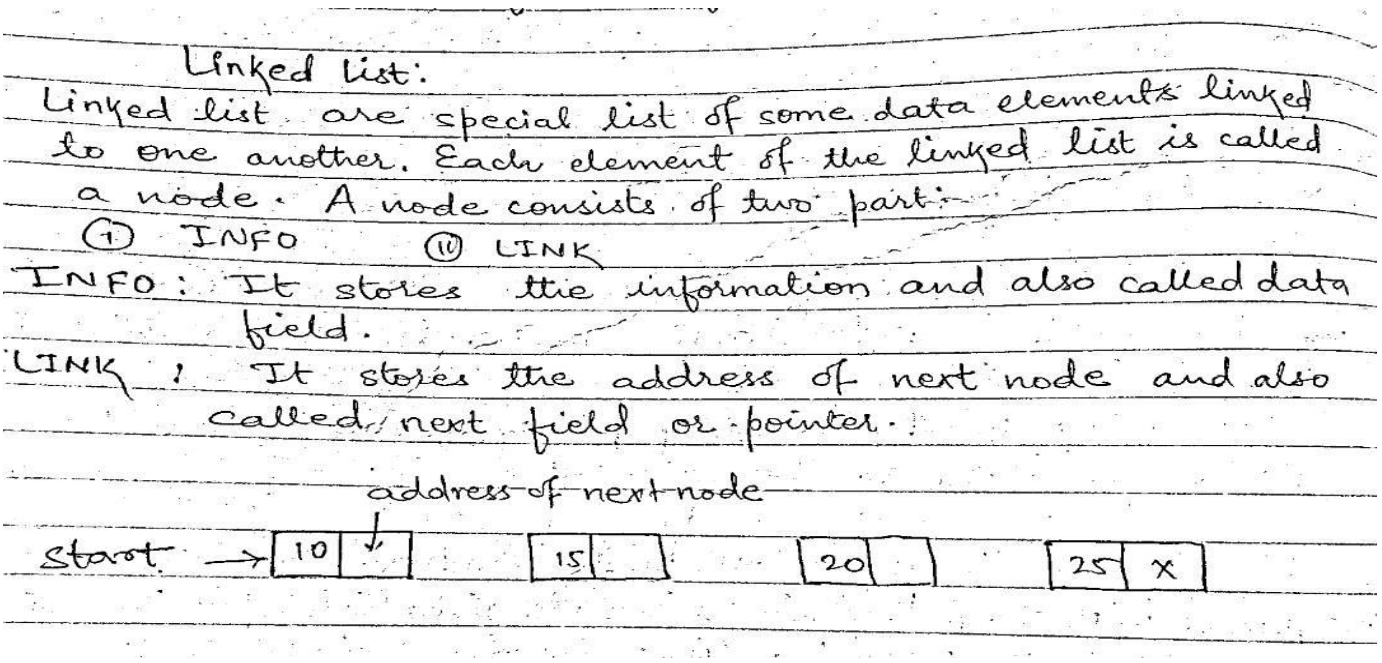
List



Linked List:

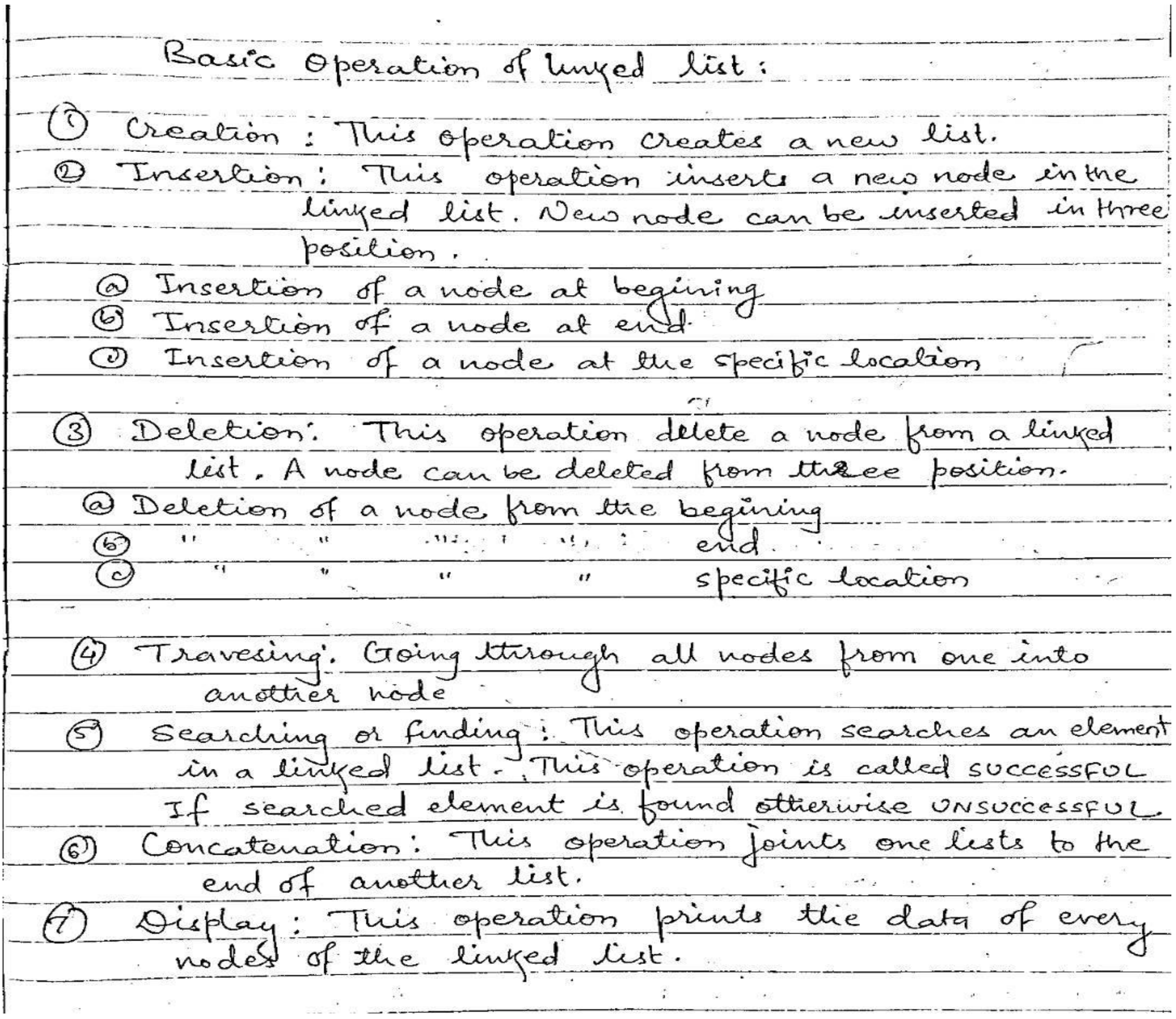
Linked list are special list of some data elements linked to one another. Each element of the linked list is called a node. A node consists of two part:-

(i) INFO (ii) LINK

INFO: It stores the information and also called data field.

LINK: It stores the address of next node and also called next field or pointer.

address of next node
Start
10
15
20
25
X



Basic Operation of linked list:

① Creation: This operation creates a new list.

② Insertion: This operation inserts a new node in the linked list. New node can be inserted in three position.

ⓐ Insertion of a node at begining
ⓑ Insertion of a node at end
ⓒ Insertion of a node at the specific location

③ Deletion: This operation delete a node from a linked list. A node can be deleted from three position.

ⓐ Deletion of a node from the begining
ⓑ " " " " end
ⓒ " " " " specific location

④ Travesing: Going through all nodes from one into another node

⑤ Searching or finding: This operation searches an element in a linked list. This operation is called SUCCESSFUL If searched element is found otherwise UNSUCCESSFUL.

⑥ Concatenation: This operation joints one lists to the end of another list.

⑦ Display: This operation prints the data of every nodes of the linked list.

Key Terms Used in Linked lists:

① Data — information hold by a node

② Link - Address of next node

③ NULL pointer - Link field of the last node contains NULL value. It is called NULL pointer. It indicates the end of the list.

④ External pointer - Pointer to very first linked list.

⑤ Empty list - If the nodes are not present in a linked list them it is called an empty list. The value of external pointer will be zero or an empty list ie. start = 0(NULL)

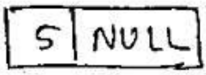
Representation of a Linked List:

```
Structure node
{
int a;
struct node *next;
};
typedef struct node NODE;
NODE * start;
```

Creating a node:

To create a new node malloc func. which dynamically allocate memory for the new node. After creating a node we can store new item in the node using a pointer to that node.

```
NodeType *P
P(NodeType *) Malloc( size of (nodeType));
P → info = 5
P → next = NULL
```



Inserting node:

To insert a node in a link list, the following three things are to be done:

- Allocating a node
- Assigning a data to info field of the node
- Adjusting a pointer

And, a new node may be inserted:

- At the begining of linked list
- At the specified location
- At the end of linked list

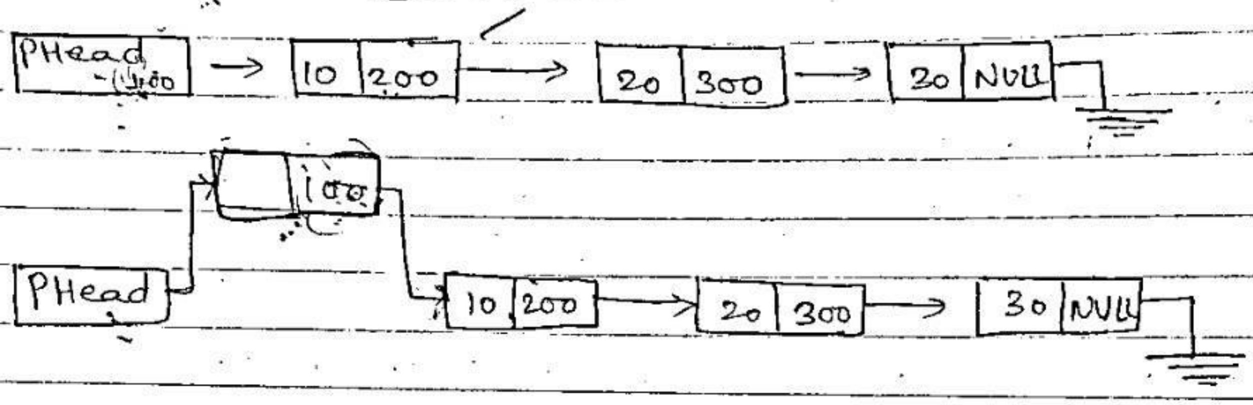
Algoritlim to insert a new node at the begining of linked list:

1. Create a NewNode using malloc func.
   pNewNode = (NodeType *) Malloc (Size of (NodeType)).

2. AssignData to info field of new node
   pNewNode → Info = newitem

3. Set next of new node to **next of Phead**
PNewNode → next = pHead

4. Set the head pointer to point to the new node
phead → pNewnode.



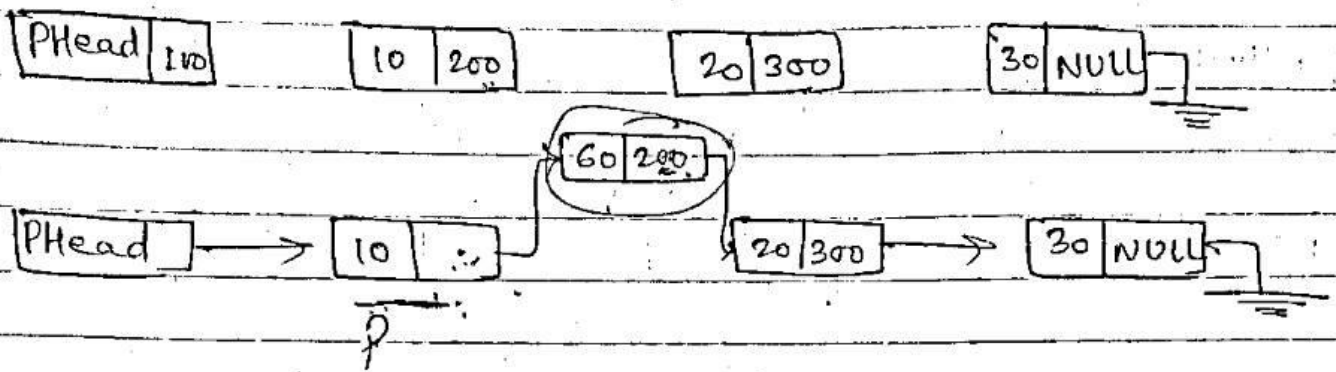
Algorithm to insert a new node at a specified location;
let P be the (Node pointer) after which we are going to add item:

1. Create a NewNode Using Malloc fucⁿ.
pNewNode = (NodeType *) Malloc(size of(NodeType));

2. Assign Data to info field of new node.
pNewNode → info = NewItem;

3. Set next of new node to next of P.
pNewnode → next = P → next;

4. Set next of P to point to the new node
P → next = PNewnode;

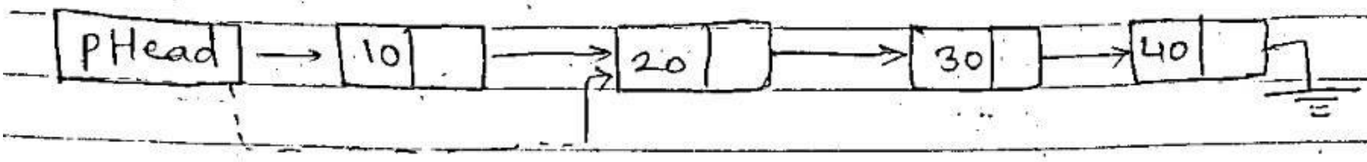


Algorithm to insert a new node at the end of the linked list:

```
1. Create a NewNode using Malloc function
   pNewNode = (Nodetype*) Malloc (Size of (NodeType));

2. Assign Data to info field of new node
      pNewNode → info = newitem

3. Set next of newnode to NULL:
      PNewNode → next = NULL

4.  If pHead = NULL Then set pHead = pNEW Node
       and Exit.
5.  Set loc = *phead
6.  While (loc → next != NULL)
           loc = loc → next
7.  Set loc → next = PNewnode
8.  End
```

Deleting nodes :

Algorithm: Deleting the first node of the linked list (deleting from begining).



~~Step-1~~

step-1: If the link list is empty printg "empty list". then exit.

Step-2 ; Store the address of first node in a temporary pointer ptemp.

step-3 : Set pHead to next ~~Atep~~ of pHead / Set pHead ~~set~~ (equal to) = next of the first node.

step-4 : Free the memory reversed by ptemp.

step-5 : Exit

Deleting the last node :

Step-1 : If the link list is empty print "empty list" then exit.

step-2 : If link list consist only one element set pHead = NULL and print item of first node.

Step-3 ; Search for the node, whose next node consist NULL in the link field.

step-4 ; Set the link field of the node as NULL.

Step-5 : Print the item of the last node.