SINGLY LINKED LISTS

A singly linked list is the simplest type of linked list in which every node contains some data and a pointer to the next node of the same data type. By saying that the node contains a pointer to the next node, we mean that the node stores the address of the next node in sequence. A singly linked list allows traversal of data only in one way. Figure 6.7 shows a singly linked list.

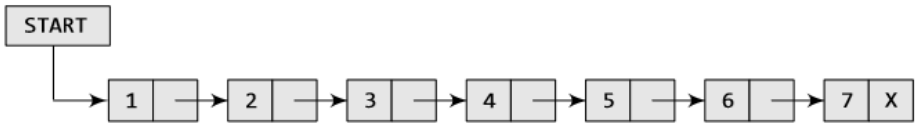

Figure 6.7 Singly linked list

6.2.1 Traversing a Linked List

Traversing a linked list means accessing the nodes of the list in order to perform some processing on them. Remember a linked list always contains a pointer variable START which stores the address of the first node of the list. End of the list is marked by storing NULL or –1 in the NEXT field of the last node. For traversing the linked list, we also make use of another pointer variable PTR which points to the node that is currently being accessed. The algorithm to traverse a linked list is shown in Fig. 6.8.

```
Step 1: [INITIALIZE] SET PTR = START
Step 2: Repeat Steps 3 and 4 while PTR != NULL
Step 3:           Apply Process to PTR -> DATA
Step 4:           SET PTR = PTR -> NEXT
       [END OF LOOP]
Step 5: EXIT
```

Figure 6.8 Algorithm for traversing a linked list

Count the number of nodes in a List

Let us now write an algorithm to count the number of nodes in a linked list. To do this, we will traverse each and every node of the list and while traversing every individual node, we will increment the counter by 1. Once we reach NULL, that is, when all the nodes of the linked list have been traversed, the final value of the counter will be displayed. Figure 6.9 shows the algorithm to print the number of nodes in a linked list.

```
Step 1: [INITIALIZE] SET COUNT = 0
Step 2: [INITIALIZE] SET PTR = START
Step 3: Repeat Steps 4 and 5 while PTR != NULL
Step 4:            SET COUNT = COUNT + 1
Step 5:            SET PTR = PTR -> NEXT
       [END OF LOOP]
Step 6: Write COUNT
Step 7: EXIT
```

Figure 6.9 Algorithm to print the number of nodes in a linked list

Searching for a value in a Linked List

Searching a linked list means to find a particular element in the linked list. As already discussed, a linked list consists of nodes which are divided into two parts, the information part and the next part. So searching means finding whether a given value is present in the information part of the node or not.

If it is present, the algorithm returns the address of the node that contains the value.

```
Step 1: [INITIALIZE] SET PTR = START
Step 2: Repeat Step 3 while PTR != NULL
Step 3:     IF VAL = PTR -> DATA
                SET POS = PTR
                Go To Step 5
            ELSE
                SET PTR = PTR -> NEXT
            [END OF IF]
        [END OF LOOP]
Step 4: SET POS = NULL
Step 5: EXIT
```

Figure 6.10 Algorithm to search a linked list

Consider the linked list shown in Fig. 6.11. If we have VAL = 4, then the flow of the algorithm can be explained as shown in the figure.

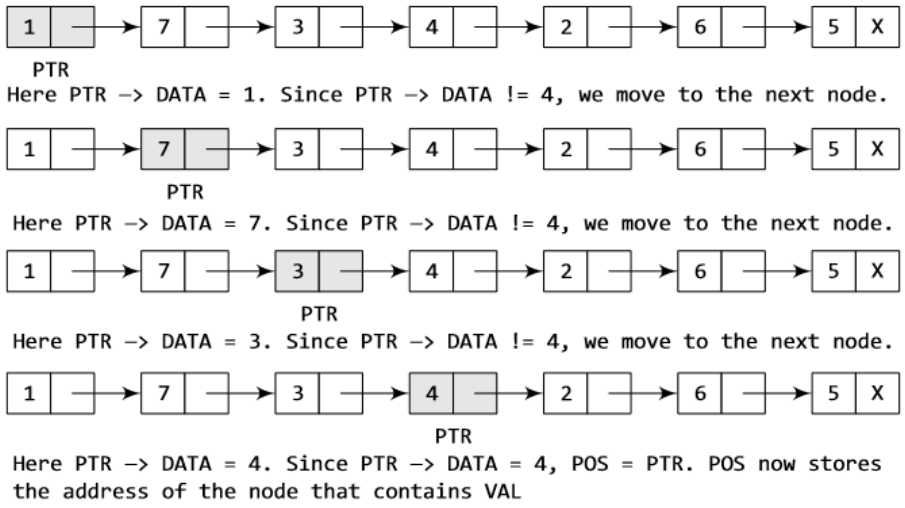

Figure 6.11 Searching a linked list