Linked Lists

A linked list is a collection of data elements called nodes in which the linear representation is given by links from one node to the next node. A linked list does not store its elements in consecutive memory locations and the user can add any number of elements to it. Unlike an array, a linked list does not allow random access of data. Elements in a linked list can be accessed only in a sequential manner. But like arrays, insertions and deletions can be done at any point in the list in a constant time.

Basic Terminologies

In C, we can implement a linked list using the following code:

```
struct node {
      int data;
      struct node *next;
};
```

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. A singly linked list allows traversal of data only in one way.

Circular Linked List

In a circular linked list, the last node contains a pointer to the first node of the list. We can have a circular singly linked list as well as a circular doubly linked list. While traversing a circular linked list, wde can begin at any node and traverse the list in any direction, forward or backward until we reach the same node where we started. Thus, a circular linked list has no beginning and no ending.

Doubly Linked Lists

A doubly linked list or a two way linked list is a more complex type of linked list which contains a pointer to the next node as well as the previous node in the sequence.

In C, We can implement the structure of a doubly linked list using the following code:

```
struct node {
      struct node *previous;
      int data;
      struct node *next;
};
```

The `previous` field of the first node and the `next` field of the last node will contain `NULL`.

Circular Doubly Linked List

A circular doubly linked list or a circular two-way linked list is a more complex type of linked list which contains a pointer to the next node as well as the previous node. The difference between a doubly linked list and a circular doubly linked list is the same as that exist between a singly linked list and a circular linked list.

The `previous` field of the first node store the address of the last node and the `next` field of the last node stores the address of the first node.

Header Linked Lists

A header linked list is a special type of linked list which contains a header node at the beginning of the list. So, in a header linked list, `START` will not point to the first node of the list but `START` will contain the address of the header node. The following are two variants of a header linked list:

- **Grounded header linked list** which stores `NULL` in the next field of the last node.

- **Circular header linked list** which stores the address of the header node in the next field of the last node.

In a Grounded header linked list, a node has two fields, `DATA` and `NEXT`. The `DATA` field will store the information part and `NEXT` field will store the address of the node in the sequence. `START` stores the address of the header node. The `NEXT` field of the header node stores the address of the first node of the list. The corresponding `NEXT` field stores the address of the next node. The Circular header linked list the last node stores the address of the header node.

Multi-Linked List

In a multi-linked list, each node can have n number of points to other nodes. A doubly linked list is a special case of multi-linked lists. However, unlike doubly linked list, nodes in a multi-linked list may or may not have inverses for each pointer. We can differentiate a doubly linked list from a multi-linked list in two ways:

(a) A doubly linked list has exactly two pointers. One pointer to the previous node and the other points to the next node. But a node in the multi-linked list can have any number of pointers.

(b) In a doubly linked list, pointers are exact inverses of each other, i.e., for every pointer which points to a previous node there is a pointer which points to the next node. This is not true for multi-linked list.

A new node can be inserted in multi-linked list in the same way as it is done for a doubly linked list.

Points to Remember

• A linked list is a linear collection of data elements called as nodes in which linear representation is given by links from one node to another.

• Linked list is a data structure which can be user to implement other data structures such as stacks, queues, and their variants.

• Before we insert a new node in linked list, we need to check for `OVERFLOW` condition, which occurs when no free memory cell is present in the system.

• Before we delete a node from a linked list, we must first check for `UNDERFLOW` condition which occurs when we try to delete a node from a linked list that is empty.

When we delete a node from a linked list, we have to actually free the memory occupied by that node. The memory is returned back to the free pool so that is can be used to store other programs and data.

• In a circular linked list, the last node contains a pointer to the first node of the list. While traversing a circular linked list, we can begin at any node and traverse the list in any direction forward or backward.

• A doubly linked list or a two-way linked list is a linked list which contains a pointer to the next as well as the previous node in the sequence. Therefore, it consists of three parts—data, a pointer to the next node, and a pointer to the previous node.

The `PREV` field ofthe first node and the `NEXT` field of the lat node contain `NULL`. This enables to traverse the list in the backward direction as well.

• A doubly linked list calls for more space per node and more expensive basic operations. However, a doubly linked list provides the ease to manipulate the elements of the list as it maintains pointers to nodes in both directions (forward or backward). The main advantage of using linked list is that it makes search operations twice as efficient.

• A circular doubly linked list or a circular two-way linked list is a more complex type of linked list which contains a pointer to the next node and previous node in the sequence. The difference between a doubly linked list and a circular linked list is the same as the circular doubly linked list does not contain `NULL` in the previous field of the first child and the next field of the last node. Rather, the next field of the last node stores the address of the first node of the list. Similarly, the previous field of the first field stores the address of the last node.

• A header linked list is a special type of linked list which contains a header node at the beginning of the list. So, in a header linked list `START` will not point to the first node of the list but `START` will contain the address of the header node.

• Multi-linked list are generally used to organize multiple orders of one set of elements. In a multi-linked list, each node can have *n* number of pointers to other nodes.