Aim

To implement Singly Linked List data structure with insertion, deletion, and display operations.

Objectives

1. To understand the concept of linked list data structure.
2. To implement insertion, deletion and display operation in a singly linked list.
3. To analyze the advantages of linked lists over arrays.

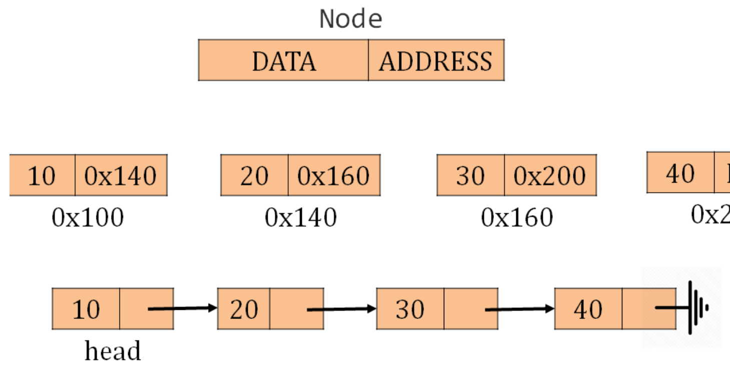
Theory

A linked list is a linear data structure where elements are stored at non-contiguous memory locations**.** Each element in a linked list is called a node.

Each node contains two parts:

- **Data** → stores the element value.
- **Pointer (next)** → stores the address of the next node.

A singly linked list is a type of linked list where each node points to the next node in the sequence, and the last node points to NULL.



The first node also known as HEAD is always used as a reference to traverse the list. Unlike arrays, linked lists provide dynamic memory allocation and allow efficient insertion and deletion**.**

Node Structure

```
class node {
public:
   int data;
   node* next;
};
```

Unidirectional Traversal: Nodes can only be traversed in one direction, from the beginning (head) to the end (tail) of the list, because each node only stores a pointer to its successor. Reverse traversal is not directly supported.

Dynamic Size: Unlike arrays, linked lists do not require contiguous memory allocation and can grow or shrink dynamically as elements are added or removed.

Algorithm for Insert Function

```
Step 1: Start

Step 2: Create a new node with given value

Step 3: If head is NULL

            Set head = new node

        Else

            Initialize q = head

            While q->next != NULL

                Move q to next node

            Set q->next = new node

Step 4: Stop
```

Algorithm for Display Function

```
Step 1: Start

Step 2: If head is NULL, print "List is empty" and return

Step 3: Initialize q = head

Step 4: While q != NULL

            Print q->data

            Move q to q->next

Step 5: Stop
```

Algorithm for Remove Function

```
Step 1: Start

Step 2: Initialize q = head, p = NULL

Step 3: If q != NULL and q->data == value
            Set head = q->next
            Delete q
            Return

Step 4: While q->next != NULL
            If q->next->data == value
                Set p = q->next
                Set q->next = p->next
                Delete p
                Return
            Move q to q->next

Step 5: If element not found, print "Value not found"

Step 6: Stop
```

Applications

- Dynamic memory allocation where size changes frequently.
- Implementation of stacks, queues, graphs, and hash tables.
- Efficient insertion and deletion operations compared to arrays.
- Used in real-time applications such as music playlists, image viewers, and undo functionality in text editors.

Questions:

1. **In a circular linked list**

   a) Components are all linked together in some sequential manner.
   b) There is no beginning and no end.
   c) Components are arranged hierarchically.
   d) Forward and backward traversal within the list is permitted.

2. **A linear collection of data elements where the linear node is given by means of pointer is called?**

   a) Linked list
   b) Node list
   c) Primitive list
   d) None

3. **In linked list each node contain minimum of two fields. One field is data field to store the data, second field is?**

   a) Pointer to character
   b) Pointer to integer
   c) Pointer to node
   d) Node

4. **In doubly linked lists, traversal can be performed?**

   a) Only in forward direction
   b) Only in reverse direction
   c) In both directions
   d) None

5. **A variation of linked list is circular linked list, in which the last node in the list points to first node of the list. One problem with this type of list is?**

   a) It waste memory space since the pointer head already points to the first node and thus the list node does not need to point to the first node.
   b) It is not possible to add a node at the end of the list.
   c) It is difficult to traverse the list as the pointer of the last node is now not NULL
   d) All of above

6. **A variant of the linked list in which none of the node contains NULL pointer is?**

   a) Singly linked list
   b) Doubly linked list
   c) Circular linked list
   d) None

7. **Which of the following statements about linked list data structure is/are TRUE?**

   a) Addition and deletion of an item to/ from the linked list require modification of the existing pointers
   b) The linked list pointers do not provide an efficient way to search an item in the linked list
   c) Linked list pointers always maintain the list in ascending order
   d) The linked list data structure provides an efficient way to find kth element in the list

8. **Linked lists are not suitable to for the implementation of?**

   a) Insertion sort
   b) Radix sort
   c) Polynomial manipulation
   d) Binary search

9. **In worst case, the number of comparison need to search a singly linked list of length n for a given element is**

   a) log n
   b) n/2
   c) $\log_2 n$-1
   d) n

10. **The situation when in a linked list head==NULL is ....**

    a) Underflow
    b) Overflow
    c) Houseful
    d) Saturated

11. **Each node in singly linked list has .... fields.**

    a) 2
    b) 3
    c) 1
    d) 4

12. **Linked list is generally considered as an example of _________ type of memory allocation.**

a) Static
b) Dynamic
c) Compile Time
d) None of these

References

1. Yedidyah Langsam, Moshe J Augenstein, Aaron M Tenenbaum – Data structures using C and C++ - PHI Publications (2nd Edition).
2. Ellis Horowitz, Sataraj Sahni- Fundamentals of Data Structures – Galgotia Books source.

Conclusion

A	C	O	T	Sign.