12/11/24 7:35 PM

* Objective:
Maintain club member information by performing different operations like add, delete, reverse concatenate on singly linked list

* Theory
* Linked list
A linked list is a linear data structure in which element called nodes are stored in a sequence each node contain two components data and a reference (or link) to the next node in the sequence This structure allows of efficient insertion and deletion operation as these can be performed without reorganizing the entire data structure

* Types of linked lists

1) Singly linked list
- each node contain data and a pointer to the next node
- Traversal is only possible in one direction (from head to tail)

2) Doubly linked list
- each node contains data. a pointer to the next node and a pointer to the previous node

3) Circular linked list
- The last node points back to the first node creating a circular structure
- useful for applicance that require a cyclic iterations that require a cyclic iteration through element

4) Cirdar Doubly linked list
- Combine the features of both circular and doubly linked list
- each node has pointers to both the next & previous nodes and the line is circular.

* Singly linked list
→ A singly linked ~~this~~ list is a linear data structure where each element called a node contains two parts called data and pointer to the next node.

→ Concept

- Node: The fundamental building block consisting of data and a next pointer.
- Head: The first node of the list. If the list is empty, the head is null
- Traversal: Accessing node in sequence from head to tail

2) Insert: Add a new element to the list
// Insert at the beginning
function Insert At beginning (data):
new Node ← New Node ()
New Node Data ← data
New Node Next ← Head
Head ← New node

3) Delete: Remove an element from the list
// Delete from the Beginning
function Delete from Beginning ()
If Head is null:
Return // list is empty
Head ← Head next

4) Search: find the element in the list
// search for an element
function search (data):
current ← Head
While current is not null:
If current . Data = data:
Return true
current ← current . Next
Return false

5) Display: print all element in the list
// Display all element
function Display ():

```
current ← Head
While current is not null:
    print current Data
    current ← current.Next
```

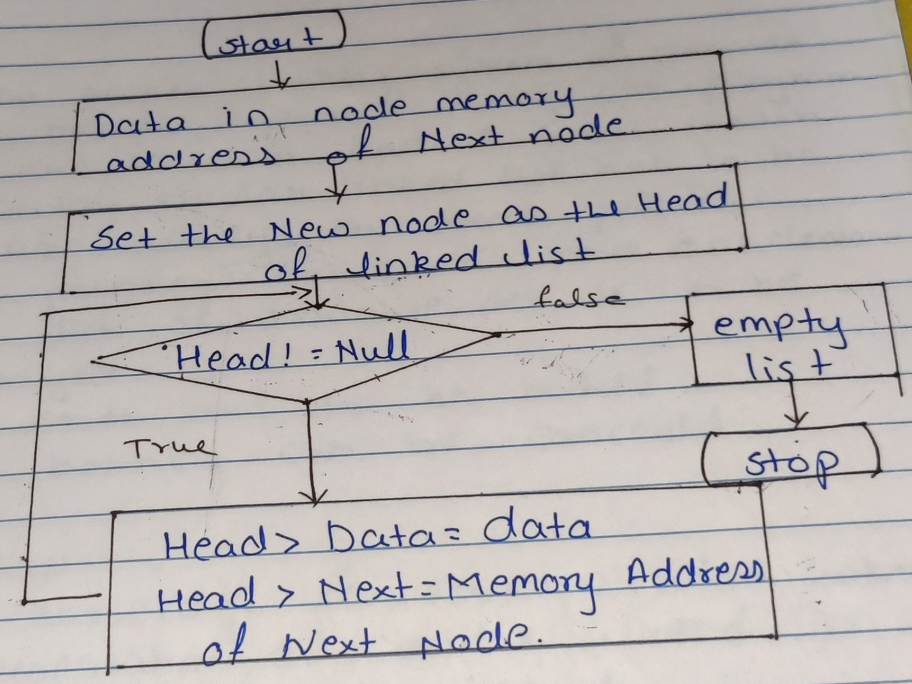
) Is empty: Check if the list is empty.

```
// check if the list is empty
function Is empty ():
Return Head is null
```

Algorithm

step 1) start
step 2) Define a Node structure
step 3) Intialize the linked list
step 4) Insert at the Beginning, end and middle
step 5) Delate Node By value
step 6) search for value
step 7) Display the list
step 8) end.

* Flowchart



* Conclusion

By this way we can maintain member information using singly list.

12/11/24
7:35 PM