Linked-List using C Part 3

Learning step 6 : Search element

Search is linear O(N) complexity starts from head node and then iterates till the pointer node data equals the target value.

```c
/*Take target value and list as parameters for the functioon
 * use pointer tnode and start from head of the list till tnode data is equal to value
 *once found print the index at which target is seen, keep track of it using i variable
*/
int search(linkedlist* list, int val){
        Node* tnode = list->head;
        int i = 0;
        while(tnode->data != val){
        tnode = tnode->next;
        i++;
        }
        printf("Exists at %d\n",i);
}
```

```c
        traverse(list);
        search(list, 10);
        return 0;
}
-- INSERT --
```

```
phenodiss1de@Phen0:~/ccode/DSA-DAY1/C-Project-based-Learning/DSA/LinkedList/Day3$ ./linkeds
0 -> 10 -> 15 -> 30 -> NULL
Exists at 1
phenodiss1de@Phen0:~/ccode/DSA-DAY1/C-Project-based-Learning/DSA/LinkedList/Day3$ vim linked_list.c
phenodiss1de@Phen0:~/ccode/DSA-DAY1/C-Project-based-Learning/DSA/LinkedList/Day3$
```

Learning step 7 : Reverse the List

Haveto say this was complicated and really interesting to understand. We need 3 pointers 1 tokeep track of next node and based on that we can just switch the link of next to previous and with the last iteration prev will be set to list head

```c
/*need 3 pointters for prev current and next
 *till current node is not null we iterate
 *in each iteration, first we temporarily store the next node in next pointer, then we use the curr pointer to point to previous
 *then just update prev with current and then current back to where the next was pointed
 * Lastly set the head to prev which was the last element
*/

void reverse(linkedlist* list){
        Node* prev = NULL;
        Node* curr = list->head;
        Node* next = NULL;
        while(curr!=NULL)
        {
                next = curr->next;
                curr->next = prev;
                prev = curr;
                curr = next;

        }
        list->head = prev;
}
```

```c
int main(){
        linkedlist* list = createlist();
        append(list, 10);
        append(list, 20);
        append(list, 30);
        prepend(list, 0);
        insertAt(list, 2, 15);
        deleteAt(list,3);
        traverse(list);
        search(list, 10);
        reverse(list);
        traverse(list);
        return 0;
}
-- INSERT --
```

```
phenodiss1de@Phen0:~/ccode/DSA-DAY1/C-Project-based-Learning/DSA/LinkedList/Day3$ ./linkeds
0 -> 10 -> 15 -> 30 -> NULL
Exists at 1
30 -> 15 -> 10 -> 0 -> NULL
```