**Data Structures**

Reversing of a linked list

**printing the data reversely:**

 data 1 add 2nd data2 add 3rd data3 NULL  

* Base condition is if head equal to null return nothing.
* Traverse through the end of a linked list recursively.
* print the data of the node after recursive step so that we traverse through

until the last node of linked list and then during return we print the data.

**void** PrintElements(lin\_list \*head){  
 //base condition  
 **if**(head==NULL){  
 **return**;  
 }  
 //recursive step  
 PrintElements(head->next);  
 printf("%d ",head->data);  
}

**Function for reversing a linked list:**

 data 1 add 2nd data2 add 3rd data3 NULL  

* create a new linked list(tail) and point it to NULL.
* Traverse through each node of linked list.
* Point the next pointer of head to tail.
* push each node to the newly created linked list(tail).

/\* Reverse the elements in the linked list \*/  
lin\_list \*ReverseList(lin\_list \*head) {  
 lin\_list \*next, \*tail = NULL;  
 **while** (head) {  
 //storing the linked list without first node into next.  
 next = head->next;  
 head->next = tail;  
 tail = head;  
 //restoring the linked list into head.  
 head = next;  
 }  
 **return** tail;  
}

**Whole program:**

#include<stdio.h>  
#include<stdlib.h>  
//creating a node.  
**typedef struct** lin\_list{  
 **int** data;  
 **struct** lin\_list \*next;  
}lin\_list;  
  
//inserting nodes  
lin\_list \*insertnode(lin\_list \*head,**int** data) {  
 lin\_list \*newnode=(lin\_list\*)malloc(**sizeof**(lin\_list));  
 newnode->data=data;  
 newnode->next=head;  
 head=newnode;  
 **return** head;  
}  
//printing the linked list.  
**void** PrintElements(lin\_list \*head){  
 //base condition  
 **if**(head==NULL){  
 **return**;  
 }  
 printf("%d ",head->data);  
 PrintElements(head->next);  
}  
  
**void** RecursiveReverse(lin\_list \*head){  
 //base condition  
 **if**(head==NULL){  
 **return**;  
 }  
 //recursive step  
 RecursiveReverse(head->next);  
 printf("%d ",head->data);  
}  
  
/\* Reverse the nodes in the linked list \*/  
lin\_list \*ReverseList(lin\_list \*head) {  
 lin\_list \*next, \*tail = NULL;  
 **while** (head) {  
 //storing the linked list without first node into next.  
 next = head->next;  
 head->next = tail;  
 tail = head;  
 //restoring the linked list into head.  
 head = next;  
 }  
 **return** tail;  
}  
**int** main(){  
 lin\_list \*head=NULL;  
 head=insertnode(head,4);  
 head=insertnode(head,3);  
 head=insertnode(head,2);  
 head=insertnode(head,1);  
 PrintElements(head);printf("\n");  
 //reversing of linked list  
 head=ReverseList(head);  
 PrintElements(head);printf("\n");  
 //printing elements reversely.  
 RecursiveReverse(head);  
 **return** 0;  
}

1 2 3 4

4 3 2 1

1 2 3 4