* reversal of a linked without recursion */

/*Algorithm:

1. The head nodeâ€™s next pointer should be set to NULL. This can be done outside the while loop. But, before we do this we will need a temp variable to point to the 2nd node

2. To reference the 2nd node is through the head nodeâ€™s next pointer.

2. The 2nd node should have itâ€™s own next pointer changed to point to the head node. This will reverse the order of the nodes. But, the 2nd nodeâ€™s next pointer will at first be pointing to the 3rd node. Before changing the 2nd nodeâ€™s next pointer, a reference to the 3rd node needs to be saved, otherwise there is no way of referencing the 3rd node.

3. Store a reference to the 3rd node in a variable before changing the 2nd nodeâ€™s next pointer.

3. The 3rd node then becomes the â€œfirstâ€ node in the while loop and the process of changing pointers continues as described in step 2.

4. Continue step 3 until, a node that has a next pointer set to NULL. Set the head node to point to the node that has the NULL next pointer. This node was previously the tail node, but is now the head node because of reversing the linked list.

*/

```
#include <iostream>

#include <cstdlib>

using namespace std;

class Node // class is made with name node

{
```

```
   public:
   int        data;
   Node*      next;


   Node(int x, Node* addr) // creation of a new node
{
      data = x;
      next = addr;
   }
};

class LinkedList // linked list class is created
{
   private:
   Node*      head;

   public:

   LinkedList() // the head of the linked list is initilized to null
 {
      head = NULL;
   }

   bool is_empty() // check if the list is empty
{
      if(head == NULL) return 1;
```

```
    else return 0;
  }

  // add an item to a linked list
  void node_add(int val) // element is added to the linked list
{
    if(head == NULL) head = new Node(val, NULL);
    else {
      Node* n = head;
      while(n->next != NULL) n = n->next;
      n->next = new Node(val, NULL);
    }
  }



  void print() // print() function prints the linked list onto the output.
{
    cout<<"[";
    if(head != NULL) {
      if(head->next == NULL) cout<<head->data;
      else {
        Node* n = head;
        while(n->next != NULL) // checking till the end of the list
{
          cout<<n->data<<",";
          n = n->next;
        }
        cout<<n->data;
```

```
        }
      }
      cout<<"]\n";
   }
   void reverse() // reverse() function to reverse the linked list
{
      if((head != NULL)*(head->next != NULL)) {
         Node* m = head;
         Node* n = head->next;
         Node* t = NULL;
         if(n->next != NULL) t = n->next;
         head->next = NULL;
         while(t->next != NULL) {
            n->next = m;
            m = n;
            n = t;
            t = t->next;
         }
         n->next = m;
         head = t;
         t->next = n;
      }
   }
};

int main(void) // main () function
{

   LinkedList     L;
```

```
    unsigned int    i;


    srand(time(NULL));

    for(i=0;i<8;i++) L.node_add(rand() % 100); // linked list taking 8 random values using randon no.
generator function


    L.print(); // calling the print () function


    L.reverse(); // calling the reverse() function to reverse the linked list


    cout << " The reversed link list is :\n\n";


    L.print(); //again calling the print()  function to output the linked list after reversal




    return(0);

}
```

/*Reverse linked list using recursion*/

```
#include<iostream>
  using namespace std;
  struct node
  {
     int data;
     struct node *next;
  };
  void insert(struct node **head,int x)
  {
     struct node *newnode= new node[sizeof(node)];
     newnode->data = x;
     newnode->next = *head;
          *head = newnode;
  }
  void display(struct node *head)
  {
     while(head!=0)
     {
        cout<<head->data<<"->";
        head=head->next;
     }
     cout<<endl;
  }
  struct node * reverse(struct node* newnode,struct node **head)
  {
     if(newnode->next==0)
```

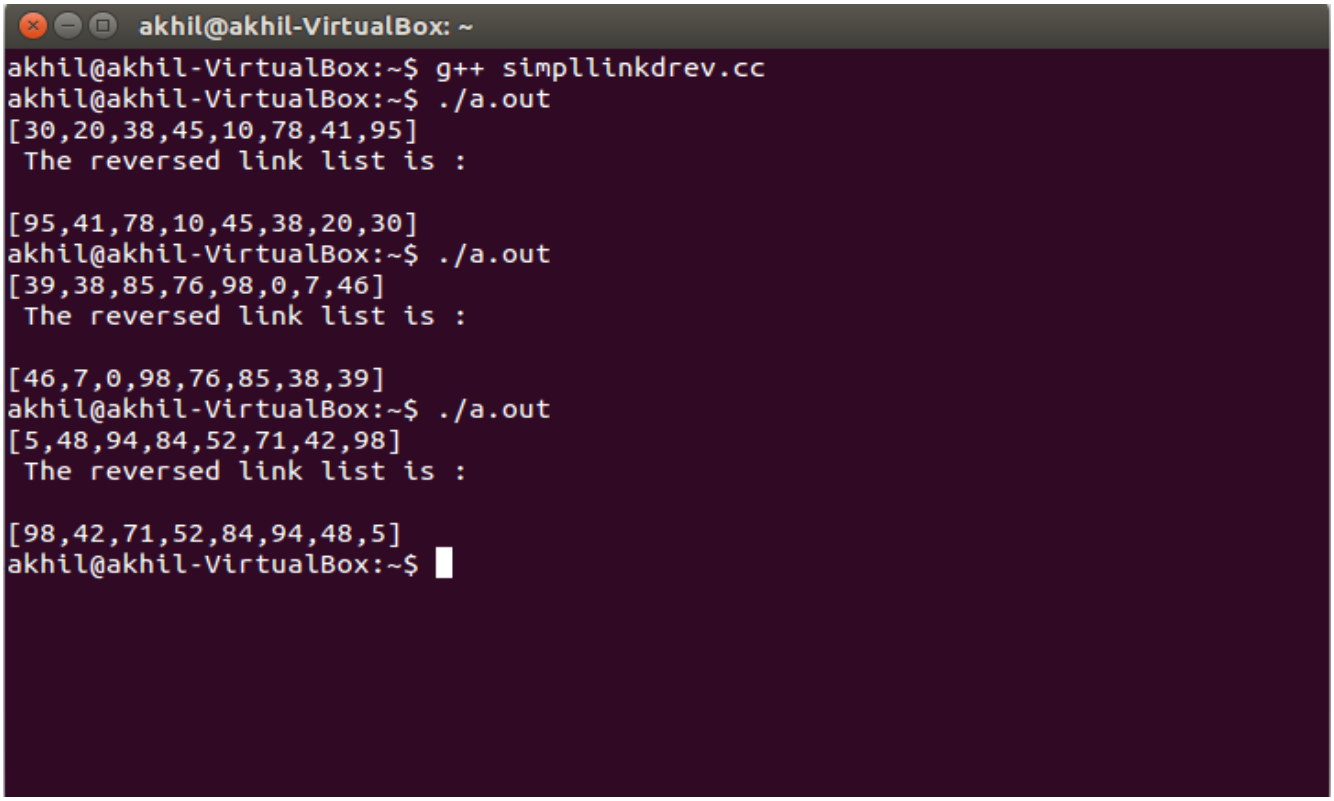
```
    {
         *head = newnode;
       return newnode;
    }
    else
    {
       struct node *prev = reverse(newnode->next,head);
                    prev -> next = newnode;
                    newnode -> next = 0;
       return newnode;
    }
}
int main()
{
        struct node *head = 0;
        int n=10;
        while(n!=0)
           insert(&head,n--);



    display(head);


    reverse(head,&head);
         display(head);
    return 0;
}
```

Reverse Linked List without recursion



Reverse Linked List using Recursion