```c
#include<stdio.h>
#include<stdlib.h>
struct node{
    int v;
    struct node *next;
};
// head points to first node of the linkedlist
struct node *head;
// inserts value at the front of LinkedList head
void insertAtBegin(int value){
    struct node *temp = (struct node*)malloc(sizeof(struct node));
    temp->v = value;
    if(head==NULL){
        temp->next = NULL;
        head = temp;
        return;
    }
    temp->next = head;
    head = temp;
}
// display linkedlist
void displayLL(struct node* head){
    struct node *temp = head;
    while(temp!=NULL){
        printf("%d->",temp->v);
        temp = temp->next;
    }
    printf("NULL \n\n");
}
struct node* reverseLLK(struct node* head, int k) {

    if(head==NULL||head->next==NULL)    return head;

    struct node* p;    //previous
    struct node* c;    //current
    struct node* n;    //next
    struct node *temp1, *temp2, *temp3;
    temp1 = temp2 = temp3 = NULL;
    int i;
```

```
40:     n = head;
41:
42:     while(n!=NULL){
43:
44:         // Reversing K-nodes from n (including node n)
45:         i = 1;
46:         temp1 = n;
47:         c = n;
48:         p = NULL;
49:         n = c->next;
50:         while (n != NULL) {
51:
52:             c->next = p;
53:             p = c;
54:             c = n;
55:             n = n->next;
56:             ++i;
57:
58:             if (i == k)
59:                 break;
60:
61:         }
62:         c->next = p;
63:
64:         // temp 3 store address of first node of K-group reversed l
65:         if(temp3==NULL)    temp3 = c;
66:
67:         // temp2 stores last node of previous reversed K-nodes
68:         if(temp2!=NULL)    temp2->next = c;
69:
70:         temp2 = temp1;
71:
72:     }
73:
74:     return temp3;
75: }
76: int main(){
77:
78:     for(int i=1;i<=12;++i){
```

```
79:         insertAtBegin(i);
80:     }
81:
82:     printf("LinkedList:\n");
83:     displayLL(head);
84:
85:     head = reverseLLK(head,4);
86:
87:     printf("LinkedList after Reverse:\n");
88:     displayLL(head);
89:
90: }
```