S.No: 13	Exp. Name: ***Write a C program to Insert an element at Begin and Count number of Nodes in Singly Linked List***	Date:2023-04-28

Aim:

The `insertAtBegin(NODE first, int x)` function inserts a new node at the beginning of the singly linked list.

The algorithm for `insertAtBegin(NODE first, int x)` is as follows:

```
Step-1: Allocate memory to the node temp.
Step-2: Store an integer value into data field of node temp.
Step-3: Assign the address contained in the first node to the next field of temp.
Step-4: Now treat the temp node as first node.
Step-5: Finally return the first node.
```

The `count(NODE first)` function counts the number of nodes linked in a singly linked list.

The algorithm for `count(node first)` is as follows:

```
Step-1: Assign the address contained in first node to temp node.
Step-2: Initialize a variable sum to 0 (zero).
Step-3: Repeat Step-4 and Step-5 until temp reaches the NULL.
Step-4: Increment the sum by 1.
Step-5: Move to the next node by placing the address of the next node in temp node.
Step-6: Finally return sum.
```

Source Code:

SingleLL2.c

```
#include<stdio.h>
#include<stdlib.h>

#include "InsAtBeginAndCount.c"

void main() {
   NODE first = NULL;
   int x, op;
   while(1) {
      printf("1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit\n");
      printf("Enter your option : ");
      scanf("%d", &op);
      switch(op) {
         case 1:  printf("Enter an element : ");
                  scanf("%d", &x);
                  first = insertAtBegin(first, x);
                  break;
         case 2:  printf("The number of nodes in a SLL are : %d\n", count(first));
                  break;
         case 3:  if (first == NULL) {
                     printf("Single Linked List is empty\n");
     }           else {
                     printf("The elements in SLL are : ");
                     traverseList(first);
```

```
    }
              break;
         case 4:  exit(0);
  }
 }
}
```

InsAtBeginAndCount.c

```
struct node {
   int data;
   struct node *next;
};
typedef struct node *NODE;
NODE createNode() {
NODE temp;
temp=(NODE)malloc(sizeof(struct node));
temp->next=NULL;
return temp;
}
NODE insertAtBegin(NODE first, int x) {
NODE temp;
temp=createNode();
temp->data=x;
temp->next=first;
first=temp;
return first;
}
int count(NODE first) {
   NODE temp=first;
   int sum=0;
   while(temp!=NULL)
   {
     sum++;
     temp=temp->next;
   }
   return sum;
}
void traverseList(NODE first) {
   NODE temp = first;
   while (temp != NULL) {
      printf("%d --> ",temp -> data);
      temp = temp -> next;
 }
   printf("NULL\n");
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 1
Enter your option : 1

```
Enter an element :  10
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 1
Enter your option :  1
Enter an element :  20
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 1
Enter your option :  1
Enter an element :  30
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 2
Enter your option :  2
The number of nodes in a SLL are : 3 3
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 3
Enter your option :  3
The elements in SLL are : 30 --> 20 --> 10 --> NULL 1
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 1
Enter your option :  1
Enter an element :  40
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 2
Enter your option :  2
The number of nodes in a SLL are : 4 3
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 3
Enter your option :  3
The elements in SLL are : 40 --> 30 --> 20 --> 10 --> NULL 4
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 4
Enter your option :  4
```

Test Case - 2

User Output

```
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 1
Enter your option :  1
Enter an element :  99
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 1
Enter your option :  1
Enter an element :  89
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 3
Enter your option :  3
The elements in SLL are : 89 --> 99 --> NULL 2
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 2
Enter your option :  2
The number of nodes in a SLL are : 2 4
1.Insert At Begin 2.Count Number of Nodes 3.Traverse the List 4.Exit 4
Enter your option :  4
```