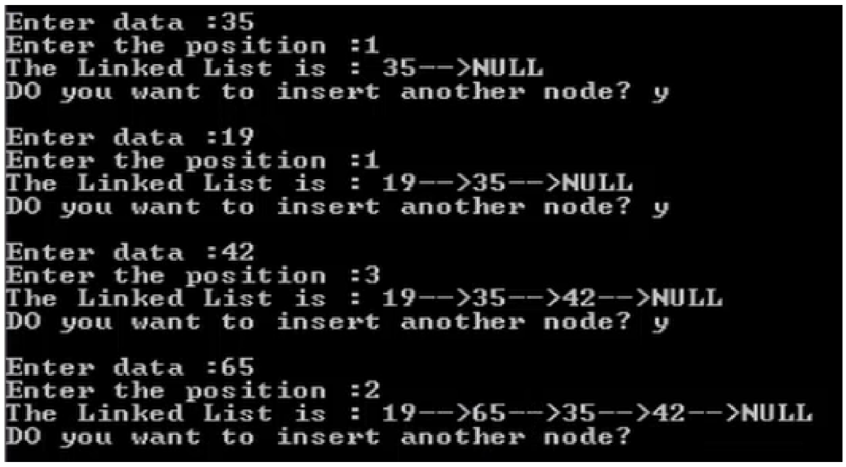
DAY-3 ASSIGNMENT | 26th December, 2020

1. Problem Statement:
Write a function “insert_any()” for inserting a node at any given position of the linked list. Assume position starts at 0.
Program in C:
```
#include <stdio.h>
#include <stdlib.h>
struct node
{
   int data;
   struct node *next;
};
struct node *head = NULL;
void insert_any()
{
   int n;
   struct node*new_node;
   new_node = (struct node*)malloc(sizeof(struct node*));
   printf("Enter the data: ");
   scanf("%d",&new_node -> data);
   new_node->next = NULL;
   printf("Enter the position: ");
   scanf("%d",&n);
   if(n==1)
   {
      new_node->next = head;
      head = new_node;
   }
   else
   {
      struct node*temp = head;
      for(int i=1;1<n-1;i++)
      {
         temp = temp->next;
      }
      new_node->next = temp->next;
      temp->next = new_node;
   }
}
void display()
{
   struct node *new_node;
   new_node = head;
   printf("The linked list is: ");
   while(new_node !=NULL)
   {
      printf("%d -->",new_node->data);
```

```
        new_node = new_node->next;
    }
    printf("NULL");
}
int main()
{
    char ch;
    do
    {
      insert_any();
      display();
      printf("\nDo you want to insert another node?");
      scanf("%c",&ch);
      printf("\n");
    }
    while(ch!='n');
    return 0;
}
```

Output:



```
Enter data :35
Enter the position :1
The Linked List is : 35-->NULL
DO you want to insert another node? y

Enter data :19
Enter the position :1
The Linked List is : 19-->35-->NULL
DO you want to insert another node? y

Enter data :42
Enter the position :3
The Linked List is : 19-->35-->42-->NULL
DO you want to insert another node? y

Enter data :65
Enter the position :2
The Linked List is : 19-->65-->35-->42-->NULL
DO you want to insert another node?
```

2. Problem Statement:

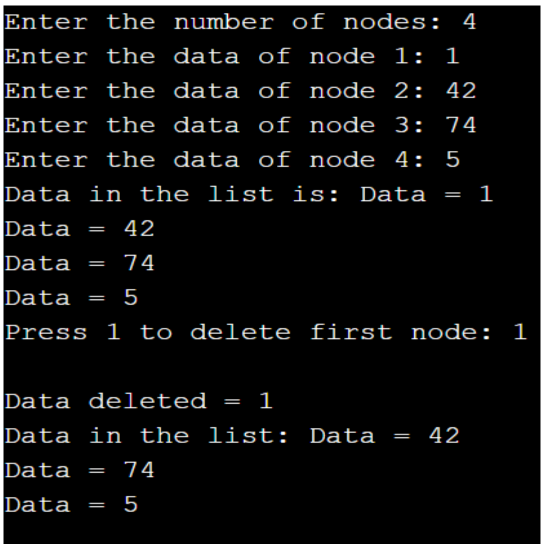
Write a function “delete_beg()” for deleting a node from the beginning of the linked list.

Program in C:

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int data;
    struct node *next;
}*head;
void createList(int n);
void delete_beg();
void displayList();
int main()
{
    int n, choice;
    printf("Enter the number of nodes: ");
    scanf("%d", &n);
    createList(n);
    printf("Data in the list is: ");
    displayList();
    printf("Press 1 to delete first node: ");
    scanf("%d", &choice);
    if(choice == 1)
        delete_beg();
    printf("Data in the list: ");
    displayList();
    return 0;
}
void createList(int n)
{
    struct node *newNode, *temp;
    int data, i;
    head = (struct node *)malloc(sizeof(struct node));
    if(head == NULL)
    {
        printf("Unable to allocate memory.");
    }
    else
    {
        printf("Enter the data of node 1: ");
        scanf("%d", &data);
        head->data = data; // Link the data field with data
        head->next = NULL; // Link the address field to NULL
        temp = head;
        for(i=2; i<=n; i++)
        {
            newNode = (struct node *)malloc(sizeof(struct node));
            if(newNode == NULL)
            {
                printf("Unable to allocate memory.");
                break;
            }
            else
            {
                printf("Enter the data of node %d: ", i);
                scanf("%d", &data);
                newNode->data = data;
                newNode->next = NULL;
                temp->next = newNode;
                temp = temp->next;
            }
        }
    }
}
```

```
void delete_beg()
{
    struct node *toDelete;
    if(head == NULL)
    {
        printf("List is already empty.");
    }
    else
    {
        toDelete = head;
        head = head->next;
        printf("\nData deleted = %d\n", toDelete->data);
        free(toDelete);
    }
}
void displayList()
{
    struct node *temp;
    if(head == NULL)
    {
        printf("List is empty.");
    }
    else
    {
        temp = head;
        while(temp != NULL)
        {
            printf("Data = %d\n", temp->data);
            temp = temp->next;
        }
    }
}
```

Output:



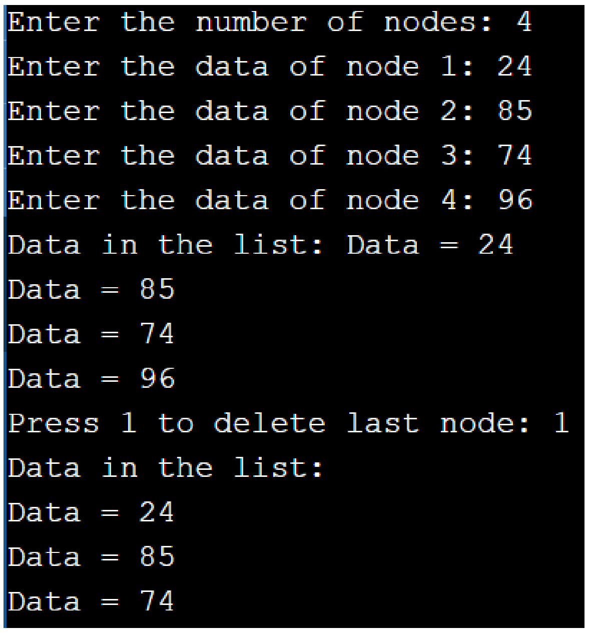
```
Enter the number of nodes: 4
Enter the data of node 1: 1
Enter the data of node 2: 42
Enter the data of node 3: 74
Enter the data of node 4: 5
Data in the list is: Data = 1
Data = 42
Data = 74
Data = 5
Press 1 to delete first node: 1

Data deleted = 1
Data in the list: Data = 42
Data = 74
Data = 5
```

3. Problem Statement:
Write a function “delete_end()” for deleting a node from the end of the linked list.
Program in C:
```
#include <stdio.h>
#include <stdlib.h>
struct node {
   int data;
   struct node *next;
}*head;
void createList(int n);
void delete_end();
void displayList();
int main()
{
   int n, choice;
   printf("Enter the number of nodes: ");
   scanf("%d", &n);
   createList(n);
   printf("Data in the list: ");
   displayList();
   printf("Press 1 to delete last node: ");
   scanf("%d", &choice);
   if(choice == 1)
      delete_end();
   printf("Data in the list:\n");
   displayList();
   return 0;
}
void createList(int n)
{
   struct node *newNode, *temp;
   int data, i;
   head = (struct node *)malloc(sizeof(struct node));
   if(head == NULL)
   {
      printf("Unable to allocate memory.");
   }
   else
   {
      printf("Enter the data of node 1: ");
      scanf("%d", &data);
      head->data = data;
      head->next = NULL;
      temp = head;
      for(i=2; i<=n; i++)
      {
         newNode = (struct node *)malloc(sizeof(struct node));
         if(newNode == NULL)
```

```
            {
                printf("Unable to allocate memory.");
                break;
            }
            else
            {
                printf("Enter the data of node %d: ", i);
                scanf("%d", &data);
                newNode->data = data;
                newNode->next = NULL;
                temp->next = newNode;
                temp = temp->next;
            }
        }
    }
}
void delete_end()
{
    struct node *toDelete, *secondLastNode;
    if(head == NULL)
    {
        printf("List is already empty.");
    }
    else
    {
        toDelete = head;
        secondLastNode = head;
        while(toDelete->next != NULL)
        {
            secondLastNode = toDelete;
            toDelete = toDelete->next;
        }
        if(toDelete == head)
        {
            head = NULL;
        }
        else
        {
            secondLastNode->next = NULL;
        }
        free(toDelete);
    }
}
void displayList()
{
    struct node *temp;
    if(head == NULL)
    {
        printf("List is empty.");
    }
    else
    {
        temp = head;
        while(temp != NULL)
        {
            printf("Data = %d\n", temp->data);
            temp = temp->next;
        }
    }
}
```

Output:



```
Enter the number of nodes: 4
Enter the data of node 1: 24
Enter the data of node 2: 85
Enter the data of node 3: 74
Enter the data of node 4: 96
Data in the list: Data = 24
Data = 85
Data = 74
Data = 96
Press 1 to delete last node: 1
Data in the list:
Data = 24
Data = 85
Data = 74
```