Code

```
#include <stdio.h>
#include <stdlib.h>
struct Node {
    int data;
    struct Node *next;
};
struct Node * head = NULL;
void createList(int n)
{
    struct Node *newNode, *temp;
    int data;
    if (n<=0)
    { printf("number of nodes should be greater than 0\n");
    return;
    }
    for(int i=1; i<=n; i++){
        newNode = (struct Node*) malloc(sizeof(struct Node));
        if(newNode==NULL){
            printf("memory allocation fail\n");
            return;
        }
        printf("enter data:\n");
        scanf("%d", &data);
        newNode -> data = data;
        newNode -> next = NULL;
        if(head==NULL)
            head = newNode;
        else
            temp -> next = newNode;
        temp = newNode;

    }
    printf("created linked list\n");
}
```

```
void insertAtBeginning(int data){
    struct Node *newNode = (struct Node*) malloc(sizeof(struct Node));
    newNode -> data = data;
    newNode -> next = head;
    if(head==NULL)
    head = newNode;
    printf("Inserted at Beginning\n");

}
void insertAtEnd(int data){
    struct Node *newNode = (struct Node*) malloc(sizeof(struct Node));
    newNode -> data = data;
    newNode -> next = NULL;
    if(head==NULL)
        head = newNode;
    else{
        struct Node *temp = head;
        while(temp -> next!=NULL)
            temp = temp -> next;
        temp -> next = newNode;
    }
    printf("Inserted at end\n");
}
void insertAtPosition(int data, int pos){
    struct Node *temp = head;
    if(pos<1)
    { printf("Invalid\n");
    return;
    }
    if(pos==1)
    { insertAtBeginning(data);
    return;
    }
    struct Node *newNode = (struct Node*) malloc(sizeof(struct Node));
    newNode -> data = data;
```

```
for( int i=1; i<pos-1 && temp
    temp = temp->next;
if (temp == NULL)
{ printf("position out of range\n");
  return;
}
else {
    newNode->next = temp->next;
    temp->next = newNode;
    printf("Inserted at position %d\n", pos);
}
}
void display(){
  struct Node *temp = head;
  if ( head == NULL)
  { printf("List is empty\n");
    return;
  }
  while (temp != NULL){
    printf("%d -> ", temp->data);
    temp = temp->next;
  }
}
int main(){
  int ch, n, data, pos;
  do{ printf("1. create Linked list, 2. Insert at begin,
        3. Insert at end, 4. Insert at position,
        5. display\n");
    printf("enter choice:\n");
    scanf("%d", &ch);
    switch(ch){
      case 1: {
        printf("enter number of nodes: ");
        scanf("%d", &n);
        createList(n);
        break; }
      case 2: {
        printf("enter data:\n");
        scanf("%d", &data);
        insertAtBeginning(data);
        break; }
      case 3: {
        printf("enter data:\n");
        scanf("%d", &data);
        insertAtEnd(data);
        break; }
      case 4: {
        printf("enter data and position:\n");
        scanf("%d %d", &data, &pos);
        insertAtPosition(data, pos);
        break; }
      case 5: {
        display();
        break; }
      default:
        printf("Invalid choice\n");
    }
  } while (ch != 0);
  return 0;
}
```

Output:

```
1. create Linked list, 2. Insert at begin, 3. Insert at end,
4. Insert at position, 5. display
enter choice:
1
enter number of nodes: 3
enter data:
10
enter data:
20
enter data:
30
created Linked list
```

1. Create linked list, 2. Insert at beg
4. Insert at position, 5. display.

enter choice:
2
enter data:
40
Inserted at beginning
1. create linked list, 2. Insert at begin, 3. Insert at end,
4. Insert at position, 5. display
enter choice:
3
enter data:
60
Inserted at end
1. create linked list, 2. Insert at begin, 3. Insert at end,
4. Insert at position, 5. display.
enter choice:
4
enter data and position:
2
70
position out of range
1. create linked list, 2. Insert at begin, 3. Insert at end,
4. Insert at position, 5. display.
enter choice:
5
40 -> 10 -> 20 -> 30 -> 60
1. create linked list, 2. Insert at begin, 3. Insert at end,
4. Insert at position, 5. display.
enter choice:
0
Invalid choice.

11/11/25